**Министерство образования и науки Российской Федерации**

Старооскольский технологический институт им. А.А. УГАРОВА

(филиал) федерального государственного автономного образовательного учреждения

высшего образования

«Национальный исследовательский технологический университет «МИСиС»

**ОСКОЛЬСКИЙ ПОЛИТЕХНИЧЕСКИЙ КОЛЛЕДЖ**

**Т.Н. Коренькова**

**Е.Н. Запасник**

**ОСНОВЫ АЛГОРИТМИЗАЦИИ И ПРОГРАММИРОВАНИЯ**

**ЧАСТЬ 1**

**Учебное пособие для студентов всех форм обучения специальности**

**09.02.04 Информационные системы (по отраслям)**

# **ОДОБРЕНО НАУЧНО-МЕТОДИЧЕСКИМ СОВЕТОМ ОПК**

# **В КАЧЕСТВЕ УЧЕБНОГО ПОСОБИЯ**

**Старый Оскол 2016**

**ББК**

**УДК**

**Рецензент:**

преподаватель ОПК СТИ НИТУ «МИСиС» *Горюнова М.В.*

**Коренькова Т.Н., Запасник Е.Н.**

**Основы алгоритмизации и программирования. Часть 1:** учебное пособие. – Старый Оскол: СТИ НИТУ «МИСиС», 2016. – 140 с.

Учебное пособие составлено в соответствии с ФГОС по специальности 03.09.02 Информационные системы. Способствует развитию навыка программирования в среде TPascal, Delphi. Рассмотрены основные понятия алгоритмизации и программирования, представлены все виды задач обработки данных, приводятся таблицы сравнительного анализа форматов, операторов, процедур, описания данных для языков программирования. Дается развернутое и практически полное описание языков и систем программирования Pascal и Delphi. Все разделы насыщены примерами и задачами.

Учебное издание предназначено для студентов системы СПО.

©Коренькова Т.Н., Запасник Е.Н., 2016

©СТИ НИТУ «МИСиС»

# **Содержание**

Введение …………………………………………………………………………………………..4

# **Практическое занятие №1.**

# Составление блок-схем алгоритмов. 5

# **Практическое занятие №2**

# Составление таблиц истинности. 9

# **Практическое занятие №3**

# Составление программ линейной структуры. 15

# **Практическое занятие №4**

# Составление программ разветвляющейся структуры. 20

# **Практическое занятие №5**

# Составление программ разветвляющейся усложненной структуры. 24

# **Практическое занятие №6**

# Составление программ циклической структуры. 32

# **Практическое занятие №7**

# Составление программ усложненной структуры. 38

# **Практическое занятие №8**

# Обработка одномерных массивов. 44

# **Практическое занятие №9.1**

# Обработка двухмерных массивов. 55

# **Практическое занятие №9.2**

# Обработка двухмерных массивов. 60

# **Практическое занятие №10**

# Использование стандартных функций для работы с массивами. 77

# **Практическое занятие №11**

# Работа со строковыми переменными. 82

# **Практическое занятие №12**

# Использование стандартных функций и процедур для работы со строками. 87

# **Практическое занятие №13**

# Работа с данными типа множество. 102

# **Практическое занятие №14**

# Организация процедур. 107

# **Практическое занятие №15**

# Использование процедур. 107

# **Практическое занятие №16**

# Организация функций. 118

# **Практическое занятие №17**

# Использование функций. 118

**Сообщения и коды ошибок**. 126

**Литература**. 139

Введение

Трудно представить себе мир без компьютеров. Человек, не умеющий обращаться с ним, находиться в положении неграмотного, попавшего в библиотеку: на полках много интересных книг, а как их прочтешь? Культура общения с компьютером стала частью общей культуры человека. Но чтобы пользоваться услугами компьютера, надо уметь правильно сформулировать ему задание – четко, однозначно и на языке который он понимает. В настоящее время благодаря своей четкости и логичности Паскаль занял достойное место среди других зыков программирования. Он прекрасно подходит как для обучения программированию, так и для создания серьезных программ-приложений.

Пособие включает в себя: задания, разобранные примеры, комментарии к выполнению, контрольные задания и вопросы, справочные материалы и литературу.

Задачи сгруппированы по темам и охватывают все разделы базового курса программирования: от объявления переменных и программ с линейной структурой до работы с объектно-ориентированным языком программирования.

В каждой практической работе имеется справочник по языку программирования Паскаль, который содержит описание наиболее часто используемых процедур и функций.

*Обращение к студентам.*

Напоминаем, что научиться программировать можно только, программируя, решая конкретные задачи. Поэтому, чтобы получить максимальную пользу, вы должны работать активно. Решайте задачи. Изучайте приведенные решения. Вводите их в компьютер. Не бойтесь экспериментировать — вносите изменения в программы. Чем больше вы сделаете самостоятельно, тем большему вы научитесь! Желаем удачи!

*Как работать с пособием?*

Группы задач следуют в том порядке, в котором изучаются соответствующие разделы в курсе программирования. Перед тем как приступить к решению задач, нужно изучить соответствующую тему — прочитать теоретический материал. Писать программу лучше сначала на бумаге (в виде блок – схем), а уже затем вводить в компьютер. Задача считается решенной, если написанная программа работает так, как сказано в условии задачи.

*Оформление решений:*

Важно, чтобы решенная задача была правильно оформлена. Это предполагает:

1. использование несущих смысловую нагрузку имен переменных, констант и функций;
2. применение отступов при записи инструкций программы;
3. использование комментариев.

Правильно оформленную программу легче отлаживать, кроме того, она производит хорошее впечатление.

**Правила выполнения практических работ**

В соответствии с общими задачами обучения и развития к уровню подготовки предъявляются следующие требования:

1. Строго выполнять весь объем домашней подготовки, указанный в описании соответствующей практической работы;
2. Знать, что при выполнении каждой работы предшествует проверка готовности студента, которая проводится преподавателем;
3. После выполнения работы студент, должен представить отчет о проделанной работе с обсуждением полученных результатов и выводов.

В конце каждого занятия преподаватель ставит зачет, который складывается из результатов наблюдения за выполнением практической части работы, проверки отчета, беседы в ходе работы или после неё.

# **Практическое занятие №1. Составление блок-схем алгоритмов.**

**Цель работы: Научиться читать листинг и составлять блок - схемы**

**Теоретическая часть:**

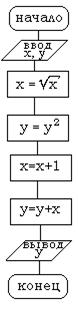
**Основные блоки**



**Примеры определения результата выполнения алгоритма по заданной блок-схеме.**

***Пример 1.***

Дана блок-схема алгоритма



Определить результат выполнения алгоритма при определённых значениях исходных данных

Например, при x=16 и y=2

Ввод: х=16 y=2

x=16=4

y=y2=4

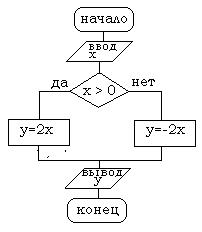
x=4+1=5

y=4+5=9

Вывод: y=9

***Пример 2.***

Дана блок-схема алгоритма



Определить результат выполнения алгоритма при определённых значениях исходных данных

Например, при x=-6 или x=0 или x=7

1)      Ввод:  х=-6

Проверка условия x>0  «Нет» y=-2\*(-6)=12

Вывод: y=12

2)      Ввод:  х=0

Проверка условия x>0  «Нет» y=-2\*0=0

Вывод: y=0

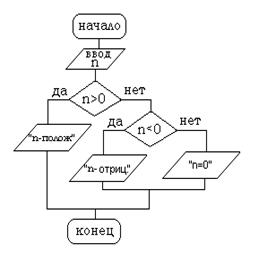
3)      Ввод:  х=7

Проверка условия x>0  «Да» y=2\*7=14

Вывод: y=14

***Пример 3.***

Дана блок-схема алгоритма



Определить результат выполнения алгоритма при определённых значениях исходных данных

Например, при n=15 или n=0 или n=-7

1)    Ввод:  n=15

Проверка условия n>0  «Да»  Вывод «n-положительное»

2)    Ввод:  n=0

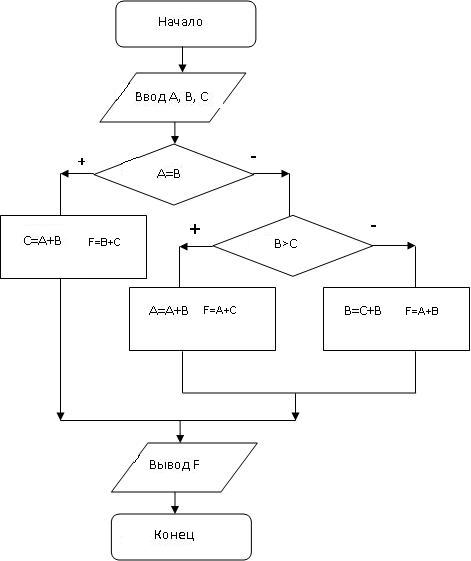
Проверка условия n>0  «Нет»  Проверка условия n<0 «Нет»  Вывод «n=0»

3)    Ввод:  n=-7

Проверка условия n>0  «Нет»  Проверка условия n<0 «Да»  Вывод «n-отрицательное»

***Пример 4.***

Дана блок-схема алгоритма



Определить результат выполнения алгоритма при определённых значениях исходных данных

Например, при A=7; B=8; C=9 или A=6; B=6; C=-10 или A=6; B=10; C=-10

1)      Ввод:  A=7; B=8; C=9

Проверка условия A=B «Нет»  Проверка условия B>C «Нет» B=9+8=17; F=7+17=24  Вывод F=24

2)      Ввод:  A=6; B=6; C=-10

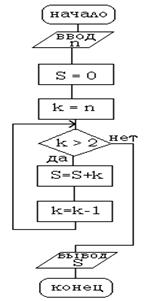
Проверка условия A=B «Да»  С=6+6=12; F=6+12=18  Вывод F=18

3)      Ввод:  A=6; B=10; C=-10

Проверка условия A=B «Нет»  Проверка условия B>C «Да» A=6+10=16; F=16+(-10)=6  Вывод F=6

***Пример 5.***

Дана блок-схема алгоритма



Определить результат выполнения алгоритма при определённых значениях исходных данных

Например, при n=4 или n=1

1)      Ввод:  n=4

S=0

k=4

Проверка условия k>2  «Да» S=0+4=4; n=4-1=3

Проверка условия k>2  «Да» S=4+3=7; n=3-1=2

Проверка условия k>2  «Нет»  Вывод S=7

2)      Ввод:  n=1

S=0

k=1

Проверка условия k>2  «Нет»  Вывод S=0

**Практическая часть**

**1. Составить блок-схемы:**

**Линейный алгоритм**

1. Вычисление площади прямоугольника.
2. Вычисление суммы четырех чисел.
3. Вычисление произведения трёх чисел.
4. Вычисление площади треугольника.
5. Вычисление частного двух чисел.
6. Вычисление длины окружности.
7. Вычисление площади круга.
8. Вычисление площади квадрата.
9. Вычисление площади параллелограмма.
10. Вычисление заданных двух чисел 5 и 18.

**Ветвление**

1. По форме фигуры определить, какая фигура: “квадрат”, “окружность”.
2. Определить виды предложений.
3. Найти значение функции Image7872.

Дополнительное задание (составить блок-схемы):

1. Пешеход шел по пересеченной местности. Его скорость движения по равнине v1 км/ч, в гору — v2 км/ч и под гору — v3 км/ч. Время движения соответственно t1, t2 и t3 ч. Какой полный путь прошел пешеход?
2. Решение квадратного уравнения.

### 2. Работа за компьютером

1) Составить алгоритм в программе MS WORD (напечатать).

Контрольные вопросы:

* + Понятие алгоритма.
  + Свойства алгоритмов.
  + Формы записей алгоритмов.
  + Общие принципы построения алгоритмов.
  + Основные алгоритмические конструкции: линейные, разветвляющиеся, циклические. Данные: понятие и типы.
  + Основные базовые типы данных и их характеристика.
  + Структурированные типы данных и их характеристика.
  + Методы сортировки данных.

Конец формы

# **Практическое занятие №2 Составление таблиц истинности.**

**Цель работы: Научиться читать таблицы истинности и решать алгебраические задачи.**

**Теоретическая часть**

**Основы логики**

***Константы*** алгебры логики (булевой алгебры) – логический 0 (ложь) и логическая 1 (истина).

Логические переменные принимают только два значения – логический 0 или логическая 1.

***Логической функции F*** от набора логических переменных х1, х2,…, хnназывается функцией, которая может принимать только два значения: логический 0 или логическую 1.

Любая логическая функция может быть задана с помощью таблицы истинности, в левой части которой записывается наборы аргументов, а в правой соответствующее значение функции.

**Элементы логических операций. Таблицы истинности. Логические схемы.**

1. Логическое умножение – *конъюнкция* – операции и – *AND*.

Обозначается: *&,^,●* или совсем опускается: *х●у*, или *х&y*, или *x^y*, или *xy*.

Логическая схема

Таблицы истинности функции *F(х,у)=ху*.

|  |  |  |
| --- | --- | --- |
| Х | У | ху |
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |



ху

и

у

х

2.Логическое сложение – *дизбюнкция* – операция *ИЛИ* - *OR.*

Обозначается: или *+:ху* или *х+у.*

Логическая схема

х

|  |  |  |
| --- | --- | --- |
| х | у | ху  у  ху  или |
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |

3.Отрицание – *инверсия* – операция *НЕ – NOT.*

Обозначается:- или:**или *х’*.

Логическая схема

Таблица истинности функции *F(x)= *.

|  |  |
| --- | --- |
| x | y |
| 0 | 1  х  не |
| 1 | 0 |

***Логические (булевы) выражения*** – это булевы константы переменные, связанные логическими операциями *И, ИЛИ, и НЕ* в единую формулу. При вычислении логического выражения учитывается следующее старшинство логических операций : инверсия (-); 2)конъюнкция (●); 3)дизъюнкция (). Для изменения порядка действий используются скобки.

Например: *F(х1, х2, х3)=(х1х2)(х1х3)х2х3*

***Теоремы алгебры логики***



4. х

5. 

6. 

7.  - *коммутативный закон*

8. 

9.

10. 

11. 

12. 

**Теоремы алгебры логики**

Конъюнкция Дизъюнкция Инверсия

Переместительный (коммуникативный) закон



Сочетательный (ассоциативный) закон



Распределительный закон



Формула де Моргена



Формула Склеивания



Формула поглощения





Рассмотрим на примере составление таблицы истинности логической функции  *F = (AB)&(AB).*

1.  Определим количество строк в таблице:

Количество строк равно количеству возможных комбинаций значений логических переменных, входящих в логическое выражение, плюс 1(заголовок столбцов). Вычисляется по формуле *2n + 1*, где n - количество переменных. В нашем случае две переменных: А, В, значит количество строк таблицы - 22 + 1 = 5.

2.  Определим количество столбцов в таблице:

Количество столбцов равно количеству логических переменных плюс количество логических операций. В нашем случае: две переменных (*А, В*) и пять логических операций (*АВ; А; В; АВ; (AB)&(AB)*), значит количество столбцов в таблице равно семи.

3.  Построим таблицу с 5-ю строками и 7-ю столбцами.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |

4.  Заполним таблицу по столбцам:

В первых столбцах записываем значения логических переменных, затем логические операции, учитывая порядок их выполнения (действия в скобках, инверсия, конъюнкция, дизъюнкция, импликация, эквивалентность) и значения таблиц истинности базовых логических операций.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **А** | **В** | **АВ** | **А** | **В** | **АВ** | **F** |
| 0 | 0 | 0 | 1 | 1 | 1 | 0 |
| 0 | 1 | 1 | 1 | 0 | 1 | 1 |
| 1 | 0 | 1 | 0 | 1 | 1 | 1 |
| 1 | 1 | 1 | 0 | 0 | 0 | 0 |

Итак мы определили значение логической функции для любого набора значений логических переменных.

Логические выражения называются равносильными, если последние столбцы их таблиц истинности совпадают. Для обозначения равносильных логических выражений используется знак «=» или «».

Пример 1: Доказать, что логические выражения *А&В* и *(АВ)* равносильны.

Построим таблицы истинности данных выражений, при одинаковых наборах значений, входящих в них переменных:

*А&В                                                                                                            (АВ)*

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **А** | **В** | **А** | **В** | **А&В** |  | **А** | **В** | **АВ** | **(АВ)** |
| 0 | 0 | 1 | 1 | 1 |  | 0 | 0 | 0 | 1 |
| 0 | 1 | 1 | 0 | 0 |  | 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 1 | 0 |  | 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 0 | 0 |  | 1 | 1 | 1 | 0 |

Значения в последних столбцах таблиц истинности совпадают, следовательно логические выражения равносильны: *А&В* = *(АВ).*

Пример 2: Построить таблицу истинности для логической формулы *А&B(AB)A*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **A** | **B** | **A** | **А&B** | **AB** | **(AB)** | **А&B(AB)** | **А&B(AB)A** |
| 0 | 0 | 1 | 0 | 0 | 1 | 1 | 1 |
| 0 | 1 | 1 | 1 | 1 | 0 | 1 | 1 |
| 1 | 0 | 0 | 0 | 1 | 0 | 0 | 1 |
| 1 | 1 | 0 | 0 | 1 | 0 | 0 | 1 |

Если при всех наборах значений переменных формула принимает значение 1, то формула является тождественно – истинной, следовательно формула *А&B(AB)A –* тождественно – истинная.

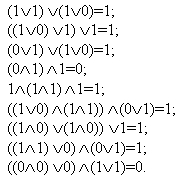
Пример 3: Построить таблицу истинности для логической формулы *(АВ)&(A&B)*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **A** | **B** | **AB** | **(AB)** | **B** | **A&B** | **(AВ)&(A&B)** |
| 0 | 0 | 0 | 1 | 1 | 0 | **0** |
| 0 | 1 | 1 | 0 | 0 | 0 | **0** |
| 1 | 0 | 1 | 0 | 1 | 1 | **0** |
| 1 | 1 | 1 | 0 | 0 | 0 | **0** |

Если при всех наборах значений переменных формула принимает значение 0, то формула является тождественно – ложной, следовательно

**Практическая часть**

1. Найдите значения логических выражений, проверите правильность решения:



2. Даны простые высказывания:

А = {Принтер - устройство ввода информации};

В = {Процессор - устройство обработки информации};

С = {Монитор - устройство хранения информации};

D = {Клавиатура - устройство ввода информации}.

Определите истинность составных высказываний:

img4

Решения (ответы).

А=0; В=1; C=0; D=1.

img5

3. Построение таблиц истинности для сложных высказываний

1. Дано составное высказываниеImage4999, где А и В - простые высказывания. В каком случае данное высказывание будет ложным?

Для ответа на поставленный вопрос требуется знать значение истинности каждого простого высказывания, входящего в сложное. Но поскольку такого знания нет, то придется рассмотреть все возможные варианты.

Строится таблица истинности:

ОбъясняютсяПравила построения таблицы истинности сложного высказывания

* определение количества строк в таблице истинности по формуле *2n*, где *n* - количество простых высказываний;
* определение порядка логических операций (приоритетов);
* заполнение исходных данных.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **А** | **В** | Image5000 | Image5001 | Image5002 |
| 0 | 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 | 1 |
| 1 | 1 | 0 | 0 | 1 |

4. С помощью таблиц истинности доказать тождества:

**Image5003**

**Image5004**

Решение (ответы)

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **А** | **В** | Image5000 | Image5005 | Image5006 | Image5007 | Image5008 | Image5009 | Image5010 | Image5011 |
| 0 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 1 |
| 0 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 |
| 1 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 |

5. Построив таблицу истинности сложного выражения Image5012, докажите, что оно является *тождественно-ложным*.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **А** | **В** | **С** | **Image5000** | **Image5005** | **Image5013** | **(1-я)** | **(2-я)** | **(3-я)** | **Итог** |
| 0 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| 1 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Выражение называется ***тождественно-ложным***, если оно ложно при любых значениях простых высказываний, в него входящих.

6. Доказать, что логическое выражение Image5014является тождественно-истинным или опровергнуть это утверждение.

Решение.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **А** | **В** | **С** | **Image5013** | **(1-я)** | **(2-я)** | **Image5008** | **Итог** | Приведенное сложное высказывание не является тождественно-истинным. |
| 0 | 0 | 0 | 1 | 0 | 0 | 1 | 1 |
| 0 | 0 | 1 | 0 | 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| 1 | 1 | 0 | 1 | 1 | 0 | 0 | 1 |
| 1 | 1 | 1 | 0 | 0 | 1 | 0 | 1 |

**Контрольные вопросы:**

1. Нарисуйте элементы И на два, четыре и пять входов, составьте для каждого из них таблицу истинности, напишите соответствующее каждому элементу логическое выражение.
2. Нарисуйте элементы ИЛИ на три, четыре и пять входов, составьте для каждого из них таблицу истинности, напишите соответствующее каждому элементу логическое выражение.
3. Нарисуйте элементы И-НЕ на два, четыре и пять входов, составьте для каждого из них таблицу истинности, напишите соответствующее каждому элементу логическое выражение.
4. Нарисуйте элементы ИЛИ-НЕ на три, четыре и пять входов, составьте для каждого из них таблицу истинности, напишите соответствующее каждому элементу логическое выражение.
5. Какой уровень сигнала является решающим для логического сложения? для логического умножения? для функции И-НЕ? для функции ИЛИ-НЕ?
6. Что такое таблица истинности?
7. Сколько строк в таблице истинности для 5-входовой логической схемы? для 4-входовой? для 2-входовой?
8. Функция скольких переменных описывается таблицей истинности длиной 4 строки? 64 строки? 512 строк?

# **Практическое занятие №3 Составление программ линейной структуры.**

**Цель работы: научиться решать задачи линейной структуры.**

## 

## **Теоретический часть**

Структура программы

Программа, написанная на языке Pascal, содержит следующие разделы:

* заголовок программы;
* раздел объявления используемых модулей;
* раздел объявления меток;
* раздел объявления констант;
* раздел объявления типов;
* раздел объявления переменных;
* раздел объявления процедур и функций;
* тело программы.

В принципе, указанный порядок следования разделов не обязателен (за исключением заголовка программы, если он есть, и раздел объявления используемых модулей), но всё-таки целесообразно придерживаться именно этого порядка, так как, во-первых, этот порядок выбран не случайно и можно обосновать именно такую последовательность разделов, а во-вторых, это приводит к единообразию в написании программ, в результате чего проще искать те или иные разделы.  
  
Вот стандартная схема программы на языке Pascal:

**program**<Имя\_программы>;

**uses**<Используемые\_модули>;

**label**

  <Используемые\_метки>;

**const**

  <Имя\_константы> = <Значение\_константы>;

**type**

  <Имя\_типа> = <Определение\_типа>

**var**

  <Имя\_переменной> : <Тип\_переменной>;

**begin**

  <Тело\_программы>

**end.**

Приведём небольшой пример - программу, вычисляющую сумму двух чисел:

**program** Example1;

**var**

  X, Y, Sum : Integer;

**begin**

  Write ('Введите числа X и Y : ');

  ReadLn (X, Y);

  Sum := X + Y;

  WriteLn ('Сумма чисел X и Y равна ', Sum);

**end.**

Здесь вводятся переменные X, Y и Sum. На экран выводится запрос на ввод X и Y, затем переменной Sum присваивается значение суммы X + Y, и результат выводится на экран. Обратите внимание на символ ';' в конце строк и на символ '.' в конце программы. Так же следует заметить, что в программе на языке Pascal можно использовать только определённые ранее переменные (в отличии от Basic, например). Так, переменные X, Y и Summ описываются в разделе **var** и имеют тип Integer.

## Процедуры ввода и вывода данных

Для ввода и вывода данных используются стандартные процедуры:

**Read, Readln, Write и Writeln**.

**Read (Список ввода); Readln(Список ввода);** - ввод данных;

**Write(Список вывода); Writeln(Список вывода); -** вывод данных.

Процедуры Readln и Writeln отличаются от Read и Write тем, что после ввода (вывода) указанного списка данных курсор перемещается на новую строку.

Список ввода содержит последовательность имён переменных, значения которых должны быть введены в процессе выполнения программы.

Список вывода может содержать значения выводимых на экран дисплея переменных, выражений или сообщения. Текст сообщений может содержать допустимые символы языка или русский текст и заключается в апострофы: 'текст'.

Например, в программе стоит оператор: Read(a,b,c);

Пользователь должен ввести значения переменных a, b и c, в соответствии с типом, указанным в подразделе Var. Курсор остается на той же строке, где располагаются значения переменных. В том случае если используется оператор Readln(a,b,c); - действия пользователя те же, но курсор после ввода перемещается на следующую строку экрана. Числовые данные при вводе отделяются друг от друга пробелами.

При выводе данных, как правило, выводятся как значения переменных,

так и пояснения. Например, необходимо вывести полученное значение Y:

Writeln(' Полученное значение Y = ',Y);

В результате выполнения на экране появится сообщение, заданное в апострофах, и значение переменной Y. После чего курсор будет установлен в начало новой строки. Процедуры Write и Writeln удобно использовать для вывода сообщений перед вводом данных с клавиатуры, что упрощает ввод данных. Например, перед вводом данных в предыдущем примере перед процедурой ввода Read нужно поставить:

Writeln(' Введите a, b и c');

**Практическая часть**

Вычислить значение выражения при вещественных типах данных. Вычисления следует выполнять с использованием промежуточных переменных. Сравнить и объяснить полученные результаты.

|  |  |  |  |
| --- | --- | --- | --- |
| № | Задание | № | Задание |
| 1 | ,  при а=1000, b=0.0001 | 14 | ,  при а=1000, b=0.0001 |
| 2 | ,  при а=1000, b=0.0001 | 15 | ,  при а=1000, b=0.0001 |
| 3 | ,  при а=1000, b=0.0001 | 16 | ,  при а=1000, b=0.0001 |
| 4 | ,  при а=1000, b=0.0001 | 17 | ,  при а=1000, b=0.0001 |
| 5 | ,  при а=1000, b=0.0001 | 18 | ,  при а=1000, b=0.0001 |
| 6 | ,  при а=1000, b=0.0001 | 19 | ,  при а=100, b=0.001 |
| 7 | ,  при а=1000, b=0.0001 | 20 | ,  при а=100, b=0.001 |
| 8 | ,  при а=100, b=0.001 | 21 | ,  при а=100, b=0.001 |
| 9 | ,  при а=100, b=0.001 | 22 | ,  при а=100, b=0.001 |
| 10 | ,  при а=100, b=0.001 | 23 | ,  при а=1000, b=0.0001 |
| 11 | ,  при а=100, b=0.001 | 24 | ,  при а=1000, b=0.0001 |
| 12 | ,  при а=1000, b=0.0001 | 25 | ,  при а=1000, b=0.0001 |
| 13 | ,  при а=1000, b=0.0001 |

**Контрольные вопросы:**

1. Что будет напечатано после выполнения программы?

program print;

begin write(1);write(2,3);writeln(4);

write(5);writeln(6,7);writeln;

write(8)

end.

1. Каков будет результат работы программы?

program wr(input,output);

var i:integer;

begin i:=maxint;

writeln(i,' ',i+1,' ',i+2);

readln;

end.

1. Даны значения переменных: *a*=1, *b*=5. Какими будут их значения после выполнения последовательности операторов? a:=b; b:=a
2. Найдите ошибки в каждой из следующих программ:

* program B;

const k=true;

var x:real;

begin read(x);

writeln(odd(x)=k)

end.

* program D;

var x:real;

begin read(x);

y:=sqr(x)+1;

writeln(y)

end.

* program E;

const B:=2.5;

var a,b,c:real;

begin read(a,c);

writeln(a\*c>b)

end.

* program F;

var a,b:integer;

begin read(A);

d:=odd(pi\*0) and b>a;

writeln(d)

end.

* program G;

var a,b:integer;

r:integer;

begin readln(a,b);

r:=a/b;

write(r)

end.

* program H;

var a,b:real;

r,p:integer;

begin readln(a,b);

r:=a div b;

p:=a mod b;

write(r,' ',p)

end.

* program I;

var a,b:integer;

r:real;

begin readln(a,b);

r:=a div b;

write(r)

end.

1. Укажите ошибки в записи оператора ввода:

read(a,b,ab);

rEaD(i);

read(begin);

read{возможно ли такое?}(a,b);

READ(a{а такое возможно?},b);

read({может быть можно так?});

re{или так?}ad(a,b);

read((a,b));

read((a),b);

read(a,a,a);

read(a,1,b);

read('Введите число ',a);

read( a, b, c);

read(a,b).

# **Практическое занятие №4 Составление программ разветвляющейся структуры.**

## **Цель работы:** научиться решать программы разветвляющей структуры.

## **Теоретическая часть**

## Оператор условного перехода

***Условный оператор*** дает возможность, в зависимости от заданного в нём условия, выполнить то или иное действие, что позволяет разветвлять

вычислительный процесс. Он имеет несколько форм записи:

1. **IF**<условие>**Then** <оператор >;

Если ***условие*** принимает значение ***TRUE (истина),*** тогда ***(THEN)*** выполняется оператор, если же FALSE, то ***оператор не выполняется*** и управление в этом случае передается оператору, следующему за IF. В качестве ***оператора*** может использоваться любой ***простой*** или ***сложный (структурный) оператор.***

Примеры записи оператора по этой форме:

IF P>0.2 then go to 13;

Z:=X;

13: P1:=P+3;

…

IF y>x THEN Z:=q+q1 ит. д.

1. **IF**<условие>**Then** <оператор 1>**Else**<оператор 2>;

где ***условие - логическое выражение;***

***оператор 1, оператор 2 -*** любые операторы языка ТР.

Здесь ключевые слова имеют смысл:

IF — если;

THEN — тогда;

ELSE — иначе.

Условный оператор работает следующим образом: если ***<условие>*** принимает значение ***TRUE*** (истина), то выполняется ***оператор 1***, а ***оператор 2*** пропускается; если ***- FALSE*** (ложь), то ***оператор 1*** пропускается, а выполняется ***оператор 2***.

На месте оператора 1 или оператора 2 может стоять группа операторов, заключенных в операторные скобки (Begin ... end).

Часть Else (оператор 2) может быть опущена. Тогда при значении TRUE

условного выражения выполняется оператор 1, в противном случае выполняется оператор, стоящий за оператором IF.

Пример: Вычислить значение функции Y, в зависимости от значения x.



Условный оператор будет иметь вид:

If x>0 Then Y:=a\*x+b Else Y:=a\*x\*x+b;

**Задачи(примеры решения)**

1. Если точка М(х,у) принадлежит внутренней области треугольника с вершинами А(а,0), В(0,b), О(0,0), то напечатать его площадь, в противном случае его периметр (a>0,b>0).

program zadacha\_3;   
  uses crt;   
    var   
     a,b,y,x: integer;   
     p,s: real;   
    begin   
     clrscr;   
       write ('Введите a ');   
       readln (a);   
       write ('Введите b ');   
       readln (b);   
       writeln ('введите координату точки x');   
       readln (x);   
       writeln ('введите y ');   
       readln (y);   
        if (x>=0) and (y>=0) and (y<(-(b/a)\*x)+b) then   
           begin   
             writeln ('точка с координатами (',x,',',y,') принадлежит области треугольника');   
             s:=a\*b;   
             write ('s=',s:5:2);   
             readln;   
          end   
        else   
           begin   
              writeln ('тоска с координатами (',x,',',y,') не принадлежит области треугольника');   
              p:=a+b+(sqrt(sqr(a)+sqr(b)));   
              write ('p=',p:5:2);   
           end;   
       readln;   
   end.

1. Составить программу, заменяющую меньшее из трех данных чисел произведением этих чисел

program zadacha\_8;

uses crt;

var

min,a,b,c:integer;

begin

clrscr;

writeln('введитечисло №1 ');

readln(a);

writeln('введитечисло №2 ');

readln(b);

writeln('введитечисло №3 ');

readln(c);

if (a < b) and (b < c) then

min:=a;

if (a > b) and (b < c ) then

min:=b;

if (a > b) and (b > c) then

min:=c;

writeln('min=',min);

readln;

min:=a\*b\*c; {присваиваю min произведение трех чисел}

writeln('min=',min);

readln;

end.

1. Составить программу, определяющую, проходит ли график функции y=x2+2x+2 через точку А(x,y).

program zadacha\_10;

uses crt;

var

x,y,x1,y1:real;

begin

clrscr;

writeln ('введите x');

readln (x);

writeln ('введите y');

readln (y);

y1:=x\*x+2\*x+2;

writeln ('y1= ',y1:2:0);

readln;

If (y=y1) then

writeln (' график проходит через точку А')

else

writeln('график не проходит через точку А');

readln;

end.

1. Вывести номер координатной плоскости, которой принадлежит точка с заданными координатами (x,y) при условии, что х<>0,y<>0.

program zadacha\_18;

uses crt;

Var

x,max,min:integer;

y:integer;

begin

clrscr;

write('Введите x - ');

readln(x);

write('Введите y - ');

readln(y);

begin

if (x<>0) and (y<>0) then

if (x>0) and (y>0) then

begin

writeln('Точка находится в I четверти');

end;

if (x<0) and (y>0) then

begin

writeln('Точка находится во II четверти');

end;

if (x<0) and (y<0) then

begin

writeln('Точка находится в III четверти');

end;

if (x>0) and (y<0) then

begin

writeln('Точка находится в IV четверти');

end;end;

readln;

end.

**Практическая часть**

Варианты самостоятельных заданий:

1. Проверить истинность высказывания: «Расстояние между точками *x* и *y* меньше 0,1».
2. Проверить истинность высказывания: «Значение переменной *х* лежит на отрезке [*a;b*]».
3. Провериь истинность высказывания: «Значение переменной *х* лежит вне отрезка [*a;b*]».
4. Проверить истинность высказывания: «Значение переменной х лежит вне отрезков [*a;b*] и [*c;d*]».
5. Провериь истинность высказывания: «Точка *М* с координатами (*x,y*) лежит внутри параболы: *ax2+bx+c*».
6. Проверить истинность высказывания: «Точка *М* с координатами (*x,y*) лежит под параболой: *ax2+bx+c*».
7. Проверить истинность высказывания: «Точка *М* с координатами (*x,y*) лежит внутри квадрата, заданного координатами левого верхнего - (*x1,y1*) и правого нижнего угла - (*x2,y2*)».
8. Проверить истинность высказывания: «Целое число *а* делится на целое число *b*».
9. Проверить истинность высказывания: «Целое число *а* является полным квадратом».
10. Проверить истинность высказывания: «Квадратное уравнение *ax2+bx+c* имеет действительные корни».
11. Проверить истинность высказывания: «Каждая из трех целых переменных *x, y, z* положительна».
12. Проверить истинность высказывания: «Хотя бы одна из трех целых переменных *x, y, z* положительна».
13. Проверить истинность высказывания: «Ни одна из трех целых переменных *x, y, z* не положительна».
14. Проверить истинность высказывания: «Только одна из трех целых переменных *x, y, z* положительна».
15. Проверить истинность высказывания «Не больше двух из трех целых переменных *x, y, z* положительны».
16. Проверить истинность высказывания: «Только две из трех целых переменных *x, y, z* положительны».
17. Проверить истинность высказывания: «Целые переменные *x, y, z* равны между собой».
18. Проверить истинность высказывания: «Среди трех целых переменных *x, y, z* имеются две равных».
19. Проверить истинность высказывания: «Среди трех целых переменных *x, y, z* имеется не более двух равных».
20. Проверить истинность высказывания: «Среди трех целых переменных *x, y, z* имеется не менее двух равных».
21. Каждое из четырех целых чисел равно 5.
22. Даны три числа a, b, c. Определить, какое из них равно некоторому числу d. Если среди данных чисел таких нет, то найти max(d-a,d-b,d-c).
23. Заданы площадь круга и квадрата. Определите, поместится ли квадрат в круг.
24. Написать программу, определяющую, равны ли два треугольника, которые заданы длинами сторон.
25. Составить программу для решения квадратного уравнения.
26. Проверить превышает ли 200 сумма квадратов двух натуральных чисел.

**Контрольные вопросы:**

1. Определите значение *а* после выполнения фрагмента программ, если в одном случае *a*=0, а в другом *a*=6:

1) if a<>0 then ; a:=2;

2) if a<4 then begin a:=a+2; a:=a+3 end;

3) if a<4 then a:=a+2; a:=a+3;

4) if a<4 then a:=a+2 else a:=a+3;

5) if a<4 then else begin a:=a+2; a:=a+3 end.

1. Может ли в записи программы на Паскале количество ключевых слов else быть больше количества ключевых слов then? А меньше?
2. Являются ли условными операторами следующие последовательности символов:

1) if x<y then x:=0 else y:=0;

2) if x>y then x:=0 else read(y);

3) if x>=y then x:=0;y:=0 else write(y);

4) if a<b then 100 else z:=25;

5) if a=b then begin; c:=3 end else c:=1;

6) if f or false then a:=2\*f else a:=c-2;

7) if c<2 or a>2 then read(a) else read(c);

8) if a+b then write('a') else write('b');

9) if a and true then begin c:=1 else c:=2 end;

10) if (1<=x) and (x<=9) then c:=2; else c:=3;

11) if 2\*a-3>0 then (a:=0) and (b:=0) and (c:=0);

12) if t<z else a:=4;

13) a:=if c>d then d else c;

# **Практическое занятие №5 Составление программ разветвляющейся усложненной структуры**.

# **Цель работы: научиться решать задачи разветвляющейся усложненной структуры.**

**Теоретическая часть**

***Оператор IF является сложным, так как в него входят другие операторы.***

**IF**<условие1>**Then** <оператор 1>

**Else IF**<условие2>**Then**<оператор 2>

**Else**<оператор 2>;

В этой форме возможности оператора IF расширены за счет вложенности новых условий.

Пример:

IF T<=Y Then Z:=cos(X)

Else IF T>Q Then Z:=sqrt(T)

ElseZ:=sin(X);

Порядок выполнения оператора таков: при T≤Y (TRUE) вычисляется Z=COS(X); если это условие FALSE, проверяется значение T>Q. Если оно TRUE, то вычисляется , если это условие FALSE, то просто вычисляем Z=Sin X.

## ***Оператор безусловного перехода***

Кроме передачи управления, в зависимости от некоторого условия, порядок выполнения операторов можно изменить с помощью оператора безусловной передачи управления вида:

**GOTO метка;**

Действие оператора Goto состоит в передаче управления в программе

соотвествующему помеченному оператору.

При использовании меток необходимо помнить:

* метка, на которую ссылается оператор ***Goto***, должна быть описана в подразделе ***Label*** и она обязательно должна встретиться в теле программы;
* метка может ставиться перед ключевым словом End, что означает переход на конец текущего блока ***Begin … End***;
* следует избегать переходов (и расстановки меток), передающих управление внутрь операторов циклов; зато выход из циклов с помощью ***Goto***может быть использован для их досрочного завершения;
* область действия оператора ***goto*** и связанных с ним меток строго локализована. Метки, описанные в основной части программы, действуют только в ней. Метки, описанные в подпрограммах, определенных программистом, действуют только внутри этих подпрограмм и поэтому передача управления извне подпрограммы на метку внутри нее невозможна.

Кроме того, ТР обладает средствами безусловного выхода из программных блоков (процедур, функций или основной программы), что позволяет завершать программу или подпрограммы без предварительных переходов по меткам. Для этого используются системные процедуры **EXIT**и **HALT**.

Вызов Exit вызывает завершение работы только того программного блока, в котором он используется.

Процедура Halt завершает выполнение всей программы.

## ***Условный оператор Case***

## Форма записи оператора выбора Case

Для разветвления в программе более чем в 2-х направлениях, можно использовать оператор выбора:

**Case<Выражение - ключ выбора>OF<Список выбора>End**

Или

**Case<Выражение - ключ выбора>OF<Список выбора>**

**Else<Операторы>End**

***Ключ выбора*** - выражение любого порядкового типа;

***<Список выбора>* -** одна или более конструкций вида:

***<Константа выбора>: <Оператор>;***

***<Константа выбора>*** - выражение того же типа, что и ключ выбора;

***Оператор выбора*** работает следующим образом: вычисляется значение выражения ***<ключ выбора>***, а затем в последовательности операторов ***<список выбора>***отыскивается такой, которому предшествует константа, равная вычисленному значению. Найденный оператор выполняется, после чего оператор выбора завершает работу. Если в списке выбора нет такой константы, то управление передается оператору, стоящему за ключеваым словом ***Else***. Часть ***Else***может отсутствовать и в этом случае оператор выбора завершает свою работу.

Решение подобных задач:

* 1. Найти сумму корней квадратного уравнения. Если эта сумма положительна и кратна трём, то увеличить коэффициенты a и b в 0.5 раза. Иначе проверить, кратна ли данная сумма произведению коэффициентов этого уравнения. Если уравнение не имеет корней, то вывести соответствующие сообщение.

program zadacha\_6;

uses crt;

var

s1,p: integer;

x1,x2,d,s,a,b,c: real;

begin

clrscr;

writeln ('Введите кооффициенты квадратного уравнения');

readln (a,b,c);

d:=sqr(b)-4\*a\*c;

if d>0 then

begin

x1:=(-b+sqrt(d))/2\*a;

x2:=(-b-sqrt(d))/2\*a;

s:=x1+x2;

s1:=round(s);

if (s>0) and ((s1 mod 2)= 0) then

begin

a:=a\*0.5;

b:=b\*0.5;

writeln ('Новоеуравнение: ',a:5:2,'x\*x+',b:5:2,'x+',c:5:2,'=0');

end

else

begin

p:=round(a\*b);

if s1 mod p=0 then

begin

writeln ('Сумма корней кратна произведению коэффициентов') ;

end

else

begin

writeln ('Сумма корней некратная произведению коэффициентов');

end;

end;

end;

if d<0 then

begin

writeln ('Уравнеине не имеет корней');

end;

readln;

end.

* 1. Даны два числа. Если эти числа кратны некоторому заданному числу, то большее из этих чисел уменьшить в два раза. В противном случае каждое из этих чисел увеличить на сумму их квадратов.

programzadacha\_11;

usescrt;

var

a:integer; {первое число}

b:integer; {второе число}

c:integer; {заданное число}

d,d1:real; {уменьшенные числа, если исходные числа кратны зад. числу}

f,f1:integer; {увеличенные числа, если исх. числа не кратны зад. числу}

begin

clrscr;

writeln('введите первое число ');

readln(a);

writeln('введите второе число ');

readln(b);

writeln('введите третье число ');

readln(c);

if (a mod c=0) and (b mod c=0) then {проверяетсякратностьчисел}

begin

if a > b then

begin

d:=a/2;

writeln('уменьшенное число, если первое число больше второго=',d:2:0);

readln;

end;

if a < b then

begin

d1:=b/2;

writeln('уменьшенное число, если второе число больше первого=',d1:2:0);

readln;

end;

end

else

begin

f:=a+sqr(a+b);

writeln('увеличенное первое число на квадрат чисел=',f:2);

readln;

f1:=b+sqr(a+b);

writeln('увеличенное второе число на квадрат числа=',f1:2);

readln;

end;

end.

* 1. Дано действительное число. Определить что больше sin(x) или cos(x). Если больше cos(x), то данное число увеличить на его квадрат, в противном случаи извлечь квадратный корень из заданного числа (если это возможно, если нет , то вывести на экран соответствующее сообщение). Если величины равны, то найти их произведение и разделить его на абсолютную величину данного числа.

program zadacha\_14;

uses crt;

var

x: real;

k: real;

begin

clrscr;

write ('Введитечисло');

readln (x);

if cos(x)>sin(x) then

begin

k:=x+sqr(x);

end;

if cos(x)0 then

begin

k:=sqrt(x);

end

else

begin

writeln ('Из введённого числа невозможно извлечь квадратный корень');

end;

end;

if cos(x)=sin(x) then

begin

k:=(cos(x)\*sin(x))/abs(x);

end;

if k<>0 then

begin

write ('Новоезначениевеличины x=',k:5:2);

end;

readln;

end.

* 1. Точки на плоскости заданы своими координатами. Определить какая из них находится ближе к началу координат.

program zadacha\_15;

uses crt;

var

x1,y1: real; {координаты первой точки}

x2,y2: real; {координаты второй точки}

x3,y3: real; {координаты третей точки}

d1,d2,d3: real; {расстояние от начала координат до каждой точки}

begin

clrscr;

writeln('Введите координаты x1 и y1 первой точки');

readln(x1,y1);

writeln('Введите координаты x2 и y2 первой точки');

readln(x2,y2);

writeln('Введите координаты x3 и y3 первой точки');

readln(x3,y3);

d1:=sqrt(sqr(x1)+sqr(y1));

d2:=sqrt(sqr(x2)+sqr(y2));

d3:=sqrt(sqr(x3)+sqr(y3));

writeln('d1=',d1);

writeln('d2=',d2);

writeln('d3=',d3);

if (d1 < d2) and (d1< d3) then

writeln('Точка с координатами: A(',x1:2:2,';',y1:2:2,') ближе к началу координат');

if (d2 < d1) and (d2 < d3) then

writeln('Точка с координатами: A(',x2:2:2,';',y2:2:2,') ближе к началу координат');

if (d3 < d1) and (d3 < d2) then

writeln('Точка с координатами: A(',x3:2:2,';',y3:2:2,') ближе к началу координат');

readln;

end.

1. Даны вещественные положительные числа a, b, c. Если существует треугольник с такими сторонами, то определить его вид (прямоугольный, остроугольный, тупоугольный).

programzadacha\_17;

usescrt;

var

a,b,c:real; {стороны треугольника}

cos1,cos2,cos3:real; {косинусыуглов}

begin

clrscr;

writeln('введите первую сторону ');

readln(a);

writeln('введите вторую сторону ');

readln(b);

writeln('введите третью сторону ');

readln(c);

begin

if (a<(b+c)) and (b<(a+c)) and (c<(b+a)) then

begin

writeln('такой треугольник существует ');

readln;

end

else

begin

writeln('такого треугольника не существует ');

readln;

end;

end;

begin

if (a<(b+c)) and (b<(a+c)) and (c<(b+a)) then begin

cos1:=((sqr(a)+sqr(b)-sqr(c))/2\*a\*b);

writeln('cos1 = ',cos1:2);

readln;

end;

end;

begin

if (a<(b+c)) and (b<(a+c)) and (c<(b+a)) then begin

cos2:=((sqr(c)+sqr(b)-sqr(a))/2\*c\*b);

writeln('cos2 = ',cos2:2);

readln;

end;

end;

begin

if (a<(b+c)) and (b<(a+c)) and (c<(b+a)) then

begin

cos3:=((sqr(a)+sqr(c)-sqr(b))/2\*a\*c);

writeln('cos3 = ',cos3:2);

readln;

end;

end;

if ((cos1<0) or (cos2<0) or (cos3<0)) and ((a<(b+c)) and (b<(a+c)) and (c<(b+а)) then

writeln('треугольник тупоугольный ');

readln;

end;

if ((cos1=0) or (cos2=0) or (cos3=0)) and ((a<(b+c)) and (b<(a+c)) and (c<(b+а)) then

writeln('треугольник прямоугольный ');

readln;

end;

if ((cos1>0) and (cos2>0) and (cos3>0)) and ((a<(b+c)) and (b<(a+c)) and (c<(b+a)) then

writeln('треугольник остроугольный ');

readln;

end;

end.

**Практическая часть**

Варианты самостоятельных заданий

1. Заданы три числа. Определить есть ли среди них хотя бы одна пара равных по величине.
2. Две прямые описываются уравнениями: a1x+b1y+c1=0; a2x+b2y+c2=0. Напечатать координаты точки пересечения этих прямых, либо сообщить, что эти прямые совпадают, не пересекаются или вовсе не существуют.
3. Написать программу, которая позволяет по данным длинам сторон треугольника выяснить, является ли он прямоугольным.
4. Определить одного ли знака два числа, вводимые с клавиатуры.
5. Определить является ли число А делителем числа В?
6. Для заданных вещественных чисел а, b и с определите, имеет ли уравнение ах + bх + с = 0 хотя бы одно вещественное решение.
7. Заданы радиус круга R и сторона квадрата А. Определить какая фигура имеет большую площадь.
8. Проверить является ли введенное с клавиатуры число кратное двум и пяти.
9. Проверить принадлежит ли точка с заданными координатами третей координатной четверти.
10. Написать программу, которая возводит данное число в куб, если оно является четным, увеличивает на 3 - в противном случае.
11. Даны числа nl, n2, n3, n4. Те числа, которые являются положительными, заменить нулями
12. Даны три числа. Выберите те из них, которые принадлежат заданному отрезку [e,f] .
13. Составить программу для решения биквадратного уравнения.
14. Дано трехзначное число. Проверить делится ли данное число на три и на девять.
15. Дано два целых числа. Если эти числа кратны некоторому числу, то большее из этих чисел уменьшить в два раза. В противном случае каждое из этих чисел увеличить на сумму их квадратов.
16. Дано трехзначное число. Проверить являются ли четными все цифры этого числа.
17. Найти количество положительных чисел среди четырех целых чисел. Если количество положительных чисел больше, чем отрицательных, то найти наибольшее среди отрицательных, иначе наибольшее среди положительных. Посчитать количество нулевых чисел.
18. Даны два числа. Проверить будут ли они кратны некоторому заданному числу n и не превышает ли их сумма квадрата числа n
19. Определить делителем каких чисел a, b, c является число k.
20. Написать программу, определяющую, равны ли два треугольника, которые заданы длинами сторон.
21. Определите, есть ли среди цифр заданного целого трёхзначного числа одинаковые.
22. Даны два действительных числа, не равные друг другу. Меньшее из них заменить половиной их суммы, а большее – удвоенным произведением.
23. Написать программу, которая позволяет по данным длинам сторон треугольника выяснить, является ли он прямоугольным.
24. Даны два угла треугольника (в градусах). Определить, существует ли такой треугольник. Определить его вид (остроугольный, прямоугольный, тупоугольный).
25. Составить программу, которая определяет, является ли счастливым трамвайный билет (т.е. равна ли сумма первых трех цифр числа, из которых оно состоит, сумме последних трех - число шестизначное).
26. Написать программу вычисления площади кольца.
27. Запишите заданное смешанное число в виде неправильной дроби.

**Контрольные вопросы:**

1. Объясните, как будут выполняться фрагменты программ:

1) ifa<bthena:=belsea:=b;

2) iffalsethenwrite('a') elsewrite('b');

3) iftruethenwrite('a') elsewrite('b');

4) if 2\*a<=3\*b then a:=a+b-14 else a:=b-14+a;

5) if a<b then a:=b else a:=a;

6) if a>b then a:=a else a:=b;

7) if a mod 2<>3 then a:=a+1;

8) if (a mod 2=1) or (a mod 2 =0) then if a<0 then a:=-a;

9) if a<0 then b:=a else b:=-a.

Замените приведенные фрагменты, где возможно, более короткими операторами Паскаля.

1. Запишите одним условным оператором:

1) ifk>5 thenc:=1;

    if k>5 then d:=2;

    if k>5 then e:=3;

    if k<=5 then write(c+d);

    if k<=5 then write(d+e);

2) if a>b then c:=1;

    if a>b then d:=2;

    if a<=b then c:=3;

    if a<=b then d:=4;

3) if x<y then x:=1 else y:=2;

    if x<y then y:=0.5 else x:=1.5;

    if x>=y then z:=0.25 else z:=2.25;

    if x>=y then d:=5;

4) if a-b<0 then else c:=1;

    if a-b<0 then else d:=2;

    if a>=b then else c:=3;

    if a>=b then else d:=4.

1. Определите, какими будут значения переменных *p* и *d* после выполнения операторов:

p:=true; d:=1;

case k mod 10 of

3,2,7,5: d:=k;

1: ;

4,8: begin p:=false; d:=2 end;

9,6: begin p:=false; d:=3 end;

end {ofcase}

если целая переменная *k* имеет значение: 1) 6; 2) 235; 3) 71; 4) 100.

# **Практическое занятие №6 Составление программ циклической структуры.**

**Цель работы: научиться решать задачи циклической структуры.**

**Теоретическая часть**

В языке ТР имеются три различных оператора, с помощью которых организуются циклические (повторяющиеся) фрагменты программ.

## ***Оператор цикла с предусловием (While)***

Оператор цикла с предусловием имеет вид:

**While <условие> Do <оператор>;**

**где <условие> - логическое выражение.**

Если логическое выражение принимает значение TRUE (истина), то выполняется оператор, стоящий после ключевого слова Do, в противном случае оператор ***While*** прекращает свою работу. На месте одного оператора может располагаться группа операторов, заключённых в операторные скобки (Begin ... end)

*ПРИМЕРЫ ПРОСТЕЙШИХ ЦИКЛИЧЕСКИХ ПРОЦЕССОВ.*

Пример 1. Вывести 20 одинаковых символов на экран.

**Program Simwol; Uses crt;**

**var i,h,n: integer**;

**Begin** {Действительно ли на экране 20 символов?}

**clrscr;** {пронумеруем символы:}

**i:=1; h:=1; n:=20;**

**while i<=n do while i<=n do**

**begin begin**

**write('\* '); writeln(i,'- \* ');**

**i:=i+h; i:=i+h;**

**end; end;**

**readkey;**

**End.**

Пример 2. Вывести на экран Пример 3. Вывести на экран

числа от 1 до 20. числаот 20 до 1.

**Program Ch; Uses crt; Program Ch; Uses crt;**

**var i,h,n: integer; var i,h,n: integer;**

**Begin Begin**

**Clrscr; clrscr;**

**i:=1; h:=1; n:=20; i:=20; h:=-1; n:=1;**

**while i<=n do while i>=n do**

**begin begin**

**write(i,' '); write(i,' ');**

**i:=i+h; i:=i+h;**

**end; end;**

**readkey; readkey;**

**End. End.**

Пример 4. Вывести на экран числа, оканчивающиеся на 7 на интервале от -100 до 100.

**Program Ch; Uses crt;**

**var i,h,n: integer;**

**Begin**

**clrscr;**

**i:=-100; h:=1; n:=100;**

**while i<=n do**

**begin**

**if i mod 10=7 then write(i,' ');**

**i:=i+h;**

**end;**

**readkey;**

**End.**

**нечетные:** **if odd(i) then write(i,' ');**

**четные:** **if not(odd(i)) then write(i,' ');**

**делящиесяна 7:** **if i mod 7=0 then write(i,' ')**

**нечетныеиделящиесяна 7:** **if (odd(i)) and (i mod 7=0) then write(i,' ');**

Пример5. Найти сумму ряда: **1, 2, ..20.**

Тип переменной S может быть integer, если сумма чисел не более 32767.

**Program Summa; Uses crt;**

**var i,n,h,s: integer;**

**Begin**

**clrscr;**

**i:=1; n:=20; h:=1; s:=0; { s:=0 -обнулениесуммы }**

**while i<=n do**

**begin**

**s:=s+i; { процесссуммирования }**

**i:=i+h;**

**end;**

**write('Сумма чисел натурального ряда от 1 до 20 равна ',s);**

**readkey;**

**End.**

Пример 6. Найти сумму ряда: **0.1, 0.3, 0.5, .. , 32.5**

**Program Summa; Uses crt;** { s - сумма }

**var x,h,n,s: real;** { h - шаг }

**Begin** { x – начальное значение параметра цикла}

**ClrScr;** { n - конечное значение параметра цикла.}

**s:=0; x:=0.1; h:=0.2; n:=32.5;**

**while (x<=n) do**

**begin**

**s:=s+x;** { -суммирование очередного элемента }

**x:=x+h;** { -изменение параметра цикла на шаг }

**end;**

**writeln('Суммаряда: 0.1, 0.3,.., 32.5 S=',s); readkey;**

**End.**

Пример 7. Вывести на экран таблицу функции **Y=sin(х),**

где х изменяется от 10 до 90 с шагом 5 градусов.

**Program Tabl; Uses crt;**

**var x, h, n, y: real;**

**Begin**

**ClrScr; writeln(' Таблицафункции y=sin(х)');**

**x:=10; h:=5; n:=90;**

**while x<=n do**

**begin**

**y:=sin(x\*pi/180);**

**writeln('X=',x,' Y=',y);**

**x:=x+h;**

**end;**

**readkey;**

**End.**

## ***Оператор цикла с постусловием (Repeat…Until)***

Оператор цикла с постусловием имеет вид:

**Repeat <Тело\_цикла> Until <условие>**

где <Тело цикла> - последовательность операторов ТР;

<условие> - логическое выражение.

Операторы "тела цикла" выполняются хотя бы один раз, после чего

вычисляется значение логического выражения и если это значение FALSE,

то операторы "тела цикла" повторяются, в противном случае оператор

цикла завершает свою работу.

Например:

Вычислить  для любого Х, с проверкой корректности ввода.

**Program Func**; **Uses crt;**

**var x,y: real;**

**Begin**

**repeat**

**ClrScr;**

**write('X='); readln(x);**

**until (x<>0);**

**y:=1/x;**

**writeln('x=',x,' y=',y);**

**readkey;**

**End.**

## ***Оператор цикла с параметром (For)***

Оператор цикла с параметром имеет вид:

**For <параметр\_цикла> := <нач.знач.> To < конеч.знач.> Do <оператор>;**

где ***<параметр\_цикла>*** - переменная любого порядкового типа;

***<нач.знач.>*** - начальное значение - выражение того же типа;

***<конеч.знач.>*** - конечное значение - выражение того же типа;

for - (фо) - для; to - (ту) - до; { шаг=1 };

downto - (даунту) - вниз до; { шаг=-1 }; do - (ду) - выполнить;

параметр цикла - переменная любого скалярного типа, кроме вещественного

(управляющая переменная) ;

выражение1 - скалярное выражение - начальное значение параметра цикла;

выражение2 - скалярное выражение - конечное значение параметра цикла;

оператор - тело цикла;

На месте одного оператора может располагаться группа операторов, заключённых в операторные скобки (Begin ... end).

Для более гибкого управления операторами циклов ***For, While, Repeat*** в ТР введены две процедуры:

***BREAK***- реализует немедленный выход из цикла; действие процедуры заключается в передаче управления оператору, стоящему сразу за после последнего оператора цикла.

***CONTINUE***- обеспечивает досрочное завершение очередного прохода цикла, что эквивалентно передаче управления в самый конец циклического оператора.

Напечатать числа от 1 до 20.

**Program Chisla; Usescrt;**

**var i:integer;** { управляющая (параметр) цикла }

**Begin**

**clrscr;**

**for i:=1 to 20 do write(i,' ');** { 1 -начальноезначениепараметра }

**End.** { 20 -конечное значение параметра }

{ write(i,' ') -тело цикла }

После каждого выполнения тела цикла параметр **i** увеличивается **на 1**.

Напечатать числа от 20 до 1.

**ProgramChisla; Usescrt;**

**var i:integer;**

**Begin**

**clrscr;**

**for i:=20 downto 1 do write(i,' ')**

**End.**

После каждого выполнения тела цикла параметр **i** увеличивается на **-1**.

Найти произведение чисел делящихся на 2 от 10 до 20.

**Program Mult; Uses crt;**

**var x:integer; P:real;**

**Begin**

**clrscr;**

**p:=1;** { начальное значение произведения }

**for x:=10 to 20 do**

**if x mod 2=0 then p:=p\*x;**

**write('Произведение чисел, делящихся на 2 от 10 до 40 равно P=',p)**

**End.**

**Практическая часть**

**Задания для самостоятельной работы:**

1. Найти сумму ряда: **5.6, 5.9, .. ,9.7**
2. Получить таблицу функции **y=cos(5∙х)**, где х изменяется от 1 до 10 с шагом 0.5
3. Найти произведение чисел от **6.7** до **7.8** с шагом **0.4**
4. Сравнить, что больше сумма ряда **8.5, 8.6,...,40.9** или произведение чисел от **1** до **3** с шагом **0.25**
5. Вычислить сумму ряда: **0.6, 0.7, 0.8, 0.9**.
6. Вывести таблицу функции **y=cos(x)** , где x изменяется в градусах от 10 до 45 с шагом 5.
7. Определить, что больше сумма ряда: **1.1 , 1.3 , ... , 93.1**или произведение: **5\*7\*...\*11**.
8. Вычислить значение функции **y=1/(x-5)**, на отрезке от 0 до 10с шагом 1 и с проверкой отсутствия деления на 0.
9. Вычислить сумму чисел из промежутка [1;1000].
10. Вывести на экран 56 символов &.
11. Получить на экране значения функции Y=SIN X,где x берется из промежутка [25;45] и измеряется в градусах.
12. Вывести на экран все числа, делящиеся на 13, от 1 до 100.
13. Вывести на экран все числа от 500 до 300.
14. Найти значение дроби: в числителе - сумма чисел от 1 до 100 делящихся на 7,а в знаменателе - сумма чисел от 100 до 500, делящихся на 17.
15. 5+6+7+…+(*n*+5);
16. 1+3+5+…+(2*n*-1);
17. sin(1)+sin(2)+sin(3)+…+sin(*n*);
18. сумму натуральных чисел отрезка [*a;b*];
19. сумму четных чисел отрезка [*a;b*];
20. 2+22+23+…+2*n*;
21. ;
22. 
23. *k*⋅(*k*+1)⋅(*k*+2)⋅…⋅2*k*;
24. (*n*+*m*)⋅(*n*+2*m*)⋅(*n*+3*m*)⋅…⋅(*n*+*m*2);
25. *a*/(*a*+1)/(*a*+2)/(*a*+3)/…/(*a*+*n*).
26. Поменять местами первую и последнюю цифры числа. Напри­мер, из числа 8547 должно быть получено число 7548.
27. Приписать к исходному числу n такое же число. Например,  
    из числа 1903 должно быть получено число 19031903.
28. Найти минимальное число, большее 300, которое нацело де­лится на 19.
29. Приписать по 1 в начало и в конец записи числа n. Напри­мер, было n=3456, стало n=134561.
30. Найти сумму целых положительных чисел, больших 20, ме­ньших 100, кратных 3 и заканчивающихся на 2, 4 или 8.
31. Сумма цифр трехзначного числа кратна 7, само число также делится на 7. Найти все такие числа.
32. Среди двузначных чисел найти те, сумма цифр которых равна n (0n18) и число делится без остатка на числа q.

**Контрольные вопросы:**

1. Определите, сколько раз будет выполнено тело цикла в приведенном фрагменте программы:

s:=0; i:=1; read(n);

while i<=n do;

begin s:=s+i;

i:=i+2;

end;

1. Нарисуйте блок-схему алгоритма, содержащего в цикле два идущих подряд условных блока.
2. Нарисуйте блок-схему алгоритма, содержащего в цикле два вложенных друг в друга условных блока.
3. Установите, какую задачу решают предложенные фрагменты программ (везде *n* – натуральное число). Ответьтенавопросы.

1) s:=0;

   i:=n;

   while i>0 do

   begin s:=s+i;

   i:=i-1

   end.

Запишите данный фрагмент с помощью операторов for… и repeat … until.

2) s:=0;

   i:=1;

   j:=n;

   while i<=j do

   begin s:=s+i+j;

   i:=i+1;

   j:=j-1

   end;

   if n mod 2=1

   then s:=s-n mod 2+1.

В каких случаях выполнится условный оператор, записанный после цикла?

3) s:=0; i:=56; j:=1;

   while i<=56+2\*(n-1) do

   begin s:=s+j;

   j:=j+1;

   i:=i+2

   end.

Можно ли в последнем фрагменте переменную *j* выразить через переменную *i* и убрать *j* из программы?

1. Правильно ли записаны на Паскале операторы цикла? Если «Да», то подсчитайте, сколько раз выполняется тело цикла. Если «Нет», объяснитеошибки.

1) while true do i:=i+1;

2) for z:=1 to 2\*z do i:=i+1;

3) while false do i:=i-1;

4) for u:=2\*a downto –1 do i:=i-1;

5) for i:=j to j+1 do s:=s+i;

6) k:=0;

for i:=2 downto k do k:=k\*2;

7) j:=56;

for i:=-4 to j div 7 do j:=j-2;

8) j:=56;

i:=-4;

while i<=j div 7 do

begin j:=j-2;

i:=i-1

end;

9) for i:=1 to n do a[i]:=2\*i+6;

for a[1]:=1 to n do write(a[a[1]],' ');

10) i:=5;while i>0 do i:=i+1;

11) i:=5;while i<=0 do i:=i-1;

12) i:=5;while i>=0 do i:=i-1;

13) i:=5;while i<=0 do i:=i+1;

14) i:=90;repeat i:=i-1 until i>0;

15) i:=90;repeat i:=i-1 until i<=0;

16) i:=9;repeat i:=i+1 until i>10;

17) i:=9;repeat i:=i+1 until i<=10;

18) i:=9;repeat i:=i-1 until false;

19) i:=9;repeat i:=i+1 until true;

20) i:=9;repeat i:=i\*2 until false and not false;

21) i:=9;repeat i:=i/2 until true or false;

22) i:=9;repeat i:=i/2 until true and false;

23) i:=9;repeat i:=i\*2 until not (i<9) and (i>=9);

24) i:=9;while (i<=100) and (i>100) do i:=i+1;

25) i:=9;while (i<=100) do i:=i div 2;

26) for i:=9 downto i div 2 do i:=i div 2;

27) while read(x) do write(x);

28) for i:=-1 to –10 do write(i+1);

29) i:=1; while i<2 do i:=j;

30) while x<1 do x:=x>0;

31) k:=0; for i:=1 to 8 do; k:=k+i;

32) g:=9; for i:=1.0 downto –1.0 do g:=g\*2;

33) g:=1; for i:=1 downto –16 d0 g:=g\*2;

35) for i+j:=1 to n do read(a,b,c);

38) for if:=1 to 10 do write('then 1 else 2');

39) for x>0 to 35 do readln;

41) x:=1; repeat x:=x+1 until 100;

42) repeat true until true;

43) x:=0; repeat x:=x+1 until x<0;

# **Практическое занятие №7 Составление программ усложненной структуры.**

**Цель работы:** научиться решать задачи усложненной структуры.

**Теоретическая часть**

*ТАБЛИЦА П И Ф А Г О Р А*

Если телом цикла является циклическая конструкция, то такие циклы называют **ВЛОЖЕННЫМИ.**

**Program Pifagor; Uses crt;**

**var i,j: integer;**

**Begin**

**clrscr;**

**gotoxy(20,1);writeln('ТаблицаПифагора');**

**FOR I:=1 TO 9 DO**

**BEGIN**

**FOR J:=1 TO 9 DO**

**BEGIN**

**gotoxy(i\*4,j\*2);**

**write(i\*j:4);**

**END;**

**WRITELN;**

**END;**

**for i:=1 to 41 do**

**begin**

**gotoxy(i,3);write('-');**

**if i<=19 then begin**

**gotoxy(9,i);**

**write('!');**

**end;**

**end;**

**gotoxy(5,22);**

**readkey;**

**End.**

Рассмотрим суммирование числового ряда, если элемент ряда является

алгебраическим выражением с помощью циклов ДЛЯ, ПОКА, ДО:

Найти сумму ряда для i от 1 до 15, каждый элемент которого выражается

по формуле , то есть найти сумму: 6+9+14+...

**Program Summa; Uses crt; Program Summa; Uses crt;**

**var s,i,n: integer; var s,i,n,h: integer;**

**Begin Begin**

**clrscr; clrscr;**

**s:=0; n:=15; s:=0; i:=1; n:=15; h:=1;**

**FOR i:=1 TO n DO s:=s+(i\*i+5); WHILE i<=n DO**

**write('Суммачисловогоряда S=',s); begin**

**End. s:=s+(i\*i+5);**

**i:=i+h;**

**Program Summa; Uses crt; end;**

**var s,i,n,h: integer; write('Суммаряда S=',s);**

**Begin End.**

**clrscr;**

**s:=0; i:=1; n:=15; h:=1;**

**REPEAT**

**s:=s+(i\*i+5);**

**i:=i+h;**

**UNTIL i>n;**

**write('Сумма числового ряда S=',s);**

**End.**

Найти произведение чисел ряда, каждый элемент которого определяется по формуле **(i+1)/i**  и i изменяется от 1 до 20 с шагом 0.2 , т.е. найтипроизведениечисел: P=2\*1.8333\*1.714\*1.625\*...

**Program Mult; Uses crt;**

**var i,p,n,h: real;**

**Begin**

**clrscr;**

**p:=1; i:=1; n:=20; h:=0.2;**

**while i<=n do begin p:=p\*(i+1)/I; i:=i+h; end;**

**write('Произведение ряда чисел P=',p);**

**End.**

Для получения качественного вывода таблицы применяется указание ШИРИНЫ

ПОЛЯ ВЫВОДА, которая пишется после переменной через двоеточие.

Например, **а:5** или **в:6** .Для числа <а> выделяется поле из пяти символов.

Но следует иметь ввиду, что целые и вещественные числа при таком выводе

в таблице будут расположены нестандартно.

Для получения более качественного вывода таблицы применяют **GOTOXY(C,R)**.

**C** - номер символа в строке, изменяется от 1 до 79 , { Column }

**R** - номер строки, изменяется от 1 до 24. { Row }

При выводе таблицы значение C не изменяется, а значение R должно, после каждой выведенной строки, увеличиваться на 1.

Рассмотрим программу решения задания #2 практического занятия #14.

**Program Tab; Uses crt;**

**var x,h,n: real; r:byte;**

**Begin**

**clrscr;**

**x:=7; h:=0.4; n:=15; r:=1;** { r - номерстроки }

**repeat**

**gotoxy(1,r); write('X=',x:2,' Y=',(x\*x+5\*x+4)/(x-6));**

**x:=x+h; r:=r+1;** ПРИМЕЧАНИЯ: 1. Выравнивание разрядов

**untilx>n;** производится if-then-else

**End**. 2. Если строк в таблице больше, чем строк на

экране, то применяется поэкранный вывод с

очисткой экрана (см. WHEREY занятие № 17).

Встречаются таблицы, в которых при некоторых значениях аргументов значения функции не определяются. В этом случае необходимо исключить прерывание программы и выдать на экран сообщение об отсутствии функции.

Вывести таблицу значений функции  на интервале 1,2..15.

**Program Tab; Uses crt;**

**var x, n: integer;**

**Begin**

**clrscr; n:=15;**

**for x:=1 to n do if x=5 then writeln('X=',x:2,' Y-неопределяется')**

**else writeln('X=',x:2,' Y=',1/(x-5) );**

**End.**

**Практическая часть**

**Задания для самостоятельного решения:**

1. Найти сумму чисел ряда, каждый элемент которого определяется поформуле (i+5)/i и i изменяется от 1 до 20 с шагом 0.4.
2. Получить таблицу для построения графика функции  на отрезке [7;15] с шагом 0.4.
3. Найти сумму ряда чисел, каждый элемент которого определяется поформуле (i+5)/(i\*i) и i=1,2,3,4,... . Суммировать до элемента равного 6E-3.
4. Получить таблицу для построения графика функции **y=sin х,** используя указание ширины поля вывода.
5. На воду опущен шар радиуса r, изготовленный из вещества плотности ρ (ρ<1). Найти расстояние от центра шара до поверхности воды. *Указание.* При составлении программы воспользоваться формулой объема шарового сегмента высоты h: V = πh2 (3r - h)/3, где r - радиус шара, h - высота шарового сегмента.
6. Составить программу, определяющую, является ли данное число N простым.
7. Составьте программу вывода на экран всех простых чисел, не превосходящих заданного N.
8. Найти все двузначные числа, сумма квадратов цифр которых делится на 13.
9. Составить программу вывода всех трехзначных чисел, сумма цифр которого равна данному натуральному числу.
10. Дано натуральное число n. Составить программу, подсчитывающую количество цифр числа n, сумму его цифр и определяющую его первую цифру.
11. В 1202 году итальянский математик Леонард Пизанский (Фибоначчи) предложил такую задачу: пара кроликов каждый месяц дает приплод – двух кроликов (самца и самку), от которых уже через два месяца уже получается новый приплод. Сколько кроликов будет через год, если в начале года имелась одна пара? Согласно условию задачи числа, соответствующие количеству кроликов, которые появляются через каждый месяц, составляют последовательность 1, 1, 2, 3, 5, 8, 13, 21, 34,… Составьте программу, позволяющие найти все числа Фибоначчи, меньше заданного числа N.
12. Найти первые сорок чисел Фибоначчи.
13. Найти первое число Фибоначчи, большее заданного числа М (М>1)
14. Вычислить сумму всех чисел Фибоначчи, которые не превосходят 1000.
15. Напишите программу, которая выведет на экран 10 строк по 5 случайных чисел в диапазоне 0…36.
16. Каждая бактерия делится на две в течение одной минуты. В начальный момент времени имеется одна бактерия. Составьте программу, которая подсчитывает количество бактерий на заданное вами целое значение момента времени.
17. Составьте программу получения в порядке убывания всех делителей данного числа.
18. Разложить число на простые множители.
19. Составьте программу определения наибольшего общего делителя двух натуральных чисел. (Реализовать в виде программы следующий вариант алгоритма Эвклида нахождения наибольшего общего делителя двух натуральных чисел: пока числа не станут равными, большее заменяется на разность большего и меньшего чисел. Когда числа станут равными одному и тому же числу d, то вычисления прекращаются: d равно наибольшему общему делителю исходных чисел).
20. Составьте программу определения наименьшего общего кратного двух натуральных чисел (использовать формулу: НОК\*НОД=a\*b)..
21. Составьте программу, определяющую максимальное из всех введенных вами чисел. (Пусть признаком конца ввода чисел является введенное число 0.)
22. Первоначальное напряжение в цепи равно 200 вольт. Известно, что оно меняется через каждую секунду следующим образом: через одну секунду оно уменьшается на М%, а еще через секунду увеличивается на N%, затем снова уменьшается на М%, потом уменьшается на N% и т.д. Составить программу для определения, через сколько секунд напряжение в цепи превысит 220 вольт.
23. Напишите программу вычисления среднего геометрического модулей двух введенных с клавиатуры целых чисел. Программа должна использовать цикл whiledo. Условие выхода из цикла – значение числа, равное 999.
24. Напишите программу, которая определит первое отрицательное число последовательности: y = sin(i/100), i = 1,2,3…
25. Дано действительное число А, большее 1. Найти среди чисел 1, 1+1/2, 1+1/2+1/3, ... первое, большее А и его номер.

**Контрольные вопросы:**

1.Установите, чему равен *х*:

1) x:=0;

   for i:=-5 to 0 do

   n:=i\*i;

   x:=x+n;

2) n:=5;

   x:=0;

   for i:=1 to n do x:=i;

   writeln(x);

3) for i:=1 to 5 do

for j:=1 to 5 do write('x');

2. Исправьте этот фрагмент так, чтобы вывод составил квадрат размером 5\*5;

4) n:=0;

   while n<=10 do

   begin x:=n\*n;

   writeln(n:6,' ',x:6)

   end;

3. Как нужно изменить этот фрагмент, чтобы были выведены числа от 1 до 10 и их квадраты. Как ту же задачу можно решить с помощью оператора for?

5) x:=0;

   whilex<5 dox:=x+1;

6) x:=0;i:=0;

   whilei<5 doi:=i+1; s:=s+1/i;

7) x:=0;i:=1;

   while i>1 do begin x:=x+1/i; i:=i-1 end;

8) x:=0 i:=1;

   repeat x:=x+1/i; i:=i-1 until i<=1;

9) x:=1; n:=1;

   for i:=2 to n do x:=x+1/i.

1. Определите значение переменной *s* после выполнения следующих операторов:

s:=0; n:=10;

for i:=2 to n do s:=s+100 div i.

1. Установите, какую задачу решает каждый из предложенных фрагментов программ:

1) readln(n);

   s:=0;

   while n>0 do

   begin s:=s\*10+n mod 10;

   n:=n div 10

   end;

2) s:=0;

   for i:=1 to n do

   for j:=i+1 to n do s:=s-a[j,i];

3) s:=0;

   for i:=3 downto –2 do

if i mod 2=0

then s:=s-i

else s:=s+i;

4) i:=p; j:=q;

   while i<j do

   begin r:=a[i]; a[i]:=a[j]; a[j]:=r;

   i:=i+1;

   j:=j-1

   end;

5) p:=1; i:=1; y:=x;

   while i<=n do

   begin y:=sin(y);

   p:=p\*y;

   i:=i+2

   end;

6) s:=0;

   for i:=n downto 1 do

   begin p:=1;

   for j:=1 to i do p:=p\*j;

   s:=s+p

   end;

7) i:=1; j:=n;

   while i<=j do

   begin read(a[i],a[j]);

   i:=i+1;

   j:=j-1

   end.

1. Студент решал задачу: «Натуральные числа вводятся в порядке возрастания до появления нуля (признак окончания ввода). Нужно напечатать факториалы всех введенных чисел. Программа должна быть эффективной, т.е. не повторять вычислений». Ниже приводятся два решения этой задачи – студента и преподавателя. Определите, какое решение удовлетворяет всем условиям задачи.

{Решениестудента}

read(a);

while a<>0 do

begin p:=1;

for i:=1 to a do p:=p\*i;

writeln(p);

read(a)

end;

{Решение преподавателя}

read(a);

p:=1; k:=1;

while a<>0 do

begin for i:=k to a do p:=p\*i;

write(p);

k:=a+1;

read(a)

end.

1. Установите, какую задачу решает приведённый фрагмент программы. Перепишите его с использованием оператора while:

s:=1; max:=a[1];

for i:=n downto 2 do

if max<a[i]

then begin max:=a[i]; s:=1 end

else if max=a[i] then s:=s+1.

1. Установите, что делает фрагмент программы. Перепишите его, используя оператор цикла с постусловием:

s:=0;

readln(a);

while a<>0 do

begin if a mod b=0

then s:=s-a;

read(a)

end.

# **Практическое занятие №8 Обработка одномерных массивов.**

**Цель работы: научиться решать задачи векторного типа.**

#### Теоретическая часть.

ПОСТАНОВКА ЗАДАЧИ - существует большое число задач, требующих много

однотипных переменных. Например, присвоить переменным значения среднесуточных температур в течение года.

Работать с таким количеством переменных очень трудно: процесс описания, присваивания, способы обработки переменных и т.д.

РЕШЕНИЕ ЗАДАЧИ - введем новое понятие - массив.

**МАССИВ - это упорядоченный по номерам набор значений, объединенных общимтипом и именем.** Упорядоченность в том, что элементы массива располагаются в последовательных, пронумерованных ячейках памяти. Массив имеет фиксированные: имя, тип и количество значений. Массиву выделяется место в памяти.

Элементы массива - это переменные с индексами. Индексы можно вычислять.

ОПРЕДЕЛЕНИЕ МАССИВА - массивы, как и простые переменные, подлежат описанию в разделе переменных:

**VAR <имя>: ARRAY[<min индекс..max индекс>] OF <тип элементов>;**

array (эррэй) - массив. of (оф) - из.

Тип индексов - любой скалярный тип кроме real. Индексы можно вычислять.

Тип элементов - любой для данной версии языка.

[min..max] - размер массива. min - нижняя, max –верхняя границы индексов. min<=max. Границы можно задать с помощью переменных.

Можно задавать постоянный массив:

**const x: array [1..10] of integer=(4,6,3,5,2,1,7,8,9,13);**

**const x: array [1..8] of char=('a','b','c','d','e','f','g','h');**

Элементы таких массивов являются константами.

Если массивы одинаковы и отличаются лишь именами, то их можно описать в одной строке:

**var a,b,c :array[1..n] of real;** , где n -верхняя граница индекса и n описывается в разделе констант.

ВИДЫ МАССИВОВ :

**Одномерные** - ряд элементов. B[5] - 5-ый элемент массива.

**Двумерные**  - таблицы. B[3,5] - элемент таблицы B ,стоящий в 3 строке и в 5 столбце.

**Многомерные** - ряд таблиц (изучаются в Вузе).

ЗАПОЛНЕНИЕ МАССИВА:

1. С помощью CONST.

2. С помощью READ и циклической конструкции.

3. Часто требуется обнуление ячеек массива. **For I:=1 to n do A[I]:=0;**

4. Для отладки программ и для программирования игр, тестов применяют заполнение массива случайными числами**. randomize**;**ForI:=1 tondoA[I]:=random(q-p+1)+p; q≥p**

5. Присваивание значений. A[1]:=7; A[2]:=77; A[3}:=33; и т. д.

#### ЗАПОЛНЕНИЕ И ВЫВОД МАССИВОВ

Заполнить и вывести на экран одномерный массив целых чисел.

**Program Primer1; uses crt;**

**var a: array[1..5] of integer;**

**i: integer;**

**Begin**

**clrscr;**

**{\*\*\*\*\*\*\* заполнениемассива c клавиатуры \*\*\*\*\*\*\*}**

**for i:=1 to 5 do**

**begin**

**write('a[',i,']='); readln(a[i]);**

**end;**

**clrscr;**

**{\*\*\*\*\* вывод массива на экран \*\*\*\*\*}**

**Write(‘МассивА: ’); for i:=1 to 5 do write(a[i],’ ‘);**

**readkey;**

**End.**

Заполнить и вывести на экран одномерный массив символов.

**Program Primer2; uses crt;**

**const a: array[1..5] of char=('a','b','c','d','e');**

**procedure Print(n: integer);**

**var i: integer;**

**begin**

**for i:=1 to n do write (a[i]:2)**

**end;**

**Begin**

**clrscr;**

**write('МассивА: '); Print(5);**

**readkey;**

**End.**

#### *СУММИРОВАНИЕ ЭЛЕМЕНТОВ МАССИВА*

ПОСТАНОВКА ЗАДАЧИ: Найти сумму элементов одномерного массива. Найти среднее арифметическое.

**ProgramSumma; usescrt;**

**constn=5;**

**vara: array[1..n] ofinteger;**

**m,i,s: integer;** Автоматическое заполнение

случайнымичисламиот1 до20

**procedure Input;** Procedure Input;

**begin** begin

**for i:=1 to n do** p:=1; q:=20; randomize;

**begin** for i:=1 to n do a[i]:=random(q-p+1)+p;

**write('a[',i,']='); readln(a[i]);** end;

**end**

**end;**

**procedure Print;**

**begin**

**write('МассивцелыхчиселА: ');**

**for i:=1 to n do write(' ',a[i])**

**writeln; writeln;** { пропускстроки }

**end;**

**procedure Sum( m: integer);**

**begin**

**s:=0;**

**for i:=1 to m do s:=s+a[i];**

**end;**

**Begin**

**ClrScr;**

**Input;** { заполнение массива из 5 элементов }

**ClrScr;**

**Print;**  { печать заполненного массива }

**write('Сколько элементов суммировать: '); readln(m);**

**Sum(m); writeln;** { суммирование m элементов массива }

**writeln('Сумма ',m,' элементов массива равна: ', s);**

{ суммирование всех элементов массива }

**Sum(n); writeln;**

**writeln('Сумма всех ',n,' элементов массива равна: ', s);**

**Write ('Среднее арифметическое всех элементов равно ', s/n);**

**Readkey;**

**End.**

#### *СУММА ЭЛЕМЕНТОВ МАССИВА ПО УСЛОВИЮ*

ПОСТАНОВКА ЗАДАЧИ: Заполнить и вывести массив на экран. Ввести условие суммирования ( а[i]<u ).

Найти сумму и вывести на экран результат.

**Program SumUsl; uses crt;**

**const a: array[1..10] of integer=(3,5,2,1,6,3,8,7,5,9);**

**var i, s, u: integer;**

**procedure Su;**

**begin**

**s:=0;**

**for i:=1 to 10 do if a[i]<u then s:=s+a[i]**

**write('Сумма элементов меньших ',u,' равна ',s);**

**end;**

**Begin**

**ClrScr;**

**Print; { смотризанятие 32 }**

**writeln; writeln;**

**write('Введитеусловиесуммирования : '); readln(u);**

**Su;**

**Readkey;**

**End.**

## *ОПРЕДЕЛЕНИЕ ЧИСЛА ЭЛЕМЕНТОВ МАССИВА ПО УСЛОВИЮ*

ПОСТАНОВКА ЗАДАЧИ:Заполнить массив и вывести его на экран.Ввести с клавиатуры условие.

Подсчитать количество элементов удовлетворяющих условию.

**Program KolUsl; uses crt;**

**const m=1; { min индекс }**

**n=10; { max индекс }**

**var i, k, usl: integer;**

**a: array[m..n] of integer;**

**... { процедуры Input и Print см. занятие 32 }**

**procedure Kol\_po\_Usl;**

**begin**

**write('Введитеограничение :'); readln(usl);**

**k:=0;**

**for i:=m to n do if a[i]>usl then k:=k+1**

**write('Кoличество элементов больших ',usl, ' равно ', k);**

**end;**

**Begin**

**ClrScr;**

**Input;**

**ClrScr;**

**Print;**

**writeln; writeln;**

**Kol\_po\_Usl;**

**Readkey;**

**End.**

# *СУММА ДВУХ ОДИНАКОВЫХ ПО РАЗМЕРУ МАССИВОВ*

В языке Pascal программист может описывать свои собственные типы и объекты, относящиеся к этому типу. Тип массив относится к *структурированным* типам.

Описание типа помещают в раздел определения типов после раздела описания констант и до раздела описания переменных (var).

Например: **TYPE massiv=array[1..100] of integer;**

**var a, b, c: massiv;**

Описание типа позволяет применить формальный параметр типа massiv.

ПОСТАНОВКА ЗАДАЧИ: Заполнить и вывести на экран два одинаковых по размеру массива: А и B.Найти массив C ,каждый элемент которого равен суммесоответствующих элементов массивов А и В: C[i]=A[i]+B[i].

**Program Summa\_Mass; uses crt;**

**Type mas= array[1..100] of integer;**

**var a, b, c: mas;**

**I, n: integer;**

**procedure Input(var x: mas);**

**begin**

**for i:=1 to n do begin write('Элемент [',i,']='); readln(x[i]); end**

**end;**

**procedure Print(x: mas); begin for i:=1 to n do write(' ',x[i]); end;**

**procedure SumMass; begin for i:=1 to n do c[i]:=a[i]+b[i]; end;**

**Begin**

**ClrScr; n:=20;**

**writeln('ВвoдмассиваА:); Input(a);**

**writeln('Ввод массива В:); Input(b);**

**ClrScr;**

**write(' A: '); Print(a);**

**write(' B: '); Print(b);**

**SumMass;**

**write(' C: '); Print(c);**

**readkey;**

**End.**

# *ОБЪЕДИНЕНИЕ ДВУХ МАССИВОВ С ЧЕРЕДОВАНИЕМ ЭЛЕМЕНТОВ*

ПОСТАНОВКА ЗАДАЧИ:

Объединить два массива A и B ,содержащих по N элементов в один массив c, который будет содержать 2\*N элементов, т.е. получить массив C=(а1,b1,a2,b2,..,an,bn).

**Program Ob\_Mass; uses crt;**

**type massiv=array[1..20] of integer;**

**var a,b,c: massiv;**

**I, n: integer;**

**procedure Input(var x: massiv; n: integer);**

**begin**

**for i:=1 to n do**

**begin**

**write('Элемент [',i,']='); readln(x[i]);**

**end**

**end;**

**procedure Print(x: massiv; n: integer);**

**...**

**procedure Ob(n: integer);**

**begin**

**for i:=1 to n do**

**begin**

**c[2\*i-1]:=a[i];**

**c[2\*i]:=b[i]**

**end**

**end;**

**Begin**

**ClrScr; n:=10;**

**Input(a,n); writeln;**

**Input(b,n);**

**ClrScr;**

**write('МассивА: '); Print(a, n); writeln;**

**write('Массив B: '); Print(b, n); writeln;**

**Ob(n);**

**write('Массив C: '); Print(c, 2\*n);**

**readkey;**

**End.**

#### *И Н В Е Р Т И Р О В А Н И Е МАССИВА*

ПОСТАНОВКА ЗАДАЧИ:

Требуется изменить порядок следования элементов массива C, состоящего из n элементов, на обратный, используя вспомогательную переменную p.

Исходный массив C: 1 2 3 4 5

Конечный массив C: 5 4 3 2 1

Перестановка производится по одному элементу через промежуточную переменную:

1) 1-ый элемент передается в ячейку p;

2) последний элемент ставится на место 1-го;

3) элемент из ячейки p ставится на последнее место.

Количество таких перестановок равно trunc(n/2) или (ndiv 2) – деление нацело.

**Program Inwert; usescrt;**

**{ описание переменных и массива }**

**{ описание процедуры ввода массива }**

**{ описание процедуры вывода массива }**

**procedure Inw;**

**var m: integer; p: real;**

**begin**

**m:=trunc(n/2);** { определение числа циклов }

**for i:=1 to m do**

**begin**

**p:=c[i];** { перестановка i элемента в p }

**c[i]:=c[n-i+1];** { перестановка на i место элемента n-i+1 }

**c[n-i+1]:=p;** { перестановка элемента из p нa n-i+1 место }

**end**

**end;**

**Begin**

**ClrScr; n:=20;**

**Input;**

**ClrScr;**

**Print; writeln;**

**Inw;**

**Print; writeln;**

**Inw;**

**Print; writeln;**

**Readkey;**

**End.**

*ФОРМИРОВАНИЕ МАССИВА ИЗ ЭЛЕМЕНТОВ ДРУГОГО*

ПОСТАНОВКА ЗАДАЧИ: Требуется из данного массива А, состоящего из N элементов, выбрать элементы, удовлетворяющие заданному ( пусть А(i)>T ), и сформировать из них массив B.

Исходный массив А: 4 6 8 2 23 1 9 Условие: A(i)>=8

Новый массив B: 8 23 9

Особенность решения: Индексы элементов массивов A и B не совпадают,

так как не все элементы массива A включаются в массив B.

Введем переменную, формирующую индексы и являющуюся счетчиком

элементовновогомассива.

**Program Massiv\_po\_Uslowiu; Uses crt;**

. . .

**procedure Mass\_Usl;**

**begin**

**write('Введите параметр условия: '); readln(t);**

**k:=0;** {k- счетчик найденых элементов }

**for i:=1 to n do if a[i]>=t then begin k:=k+1; b[k]:=a[i]; end;**

**end;**

**Begin**

**ClrScr; n:=20;**

**Input;** { заполнение исходного массива A }

**ClrScr;**

**Print(а, n); writeln;** { печать исходного массива A }

**Mass\_Usl;** { формирование массива по условию }

**Print(b, k); writeln;** { вывод на экран нового массива B }

**Readkey;**

**End.**

*ПОИСК МАКСИМАЛЬНОГО ЭЛЕМЕНТА В МАССИВЕ*

*С ЗАПОМИНАНИЕМ ЕГО ПОЛОЖЕНИЯ (ИНДЕКСА)*

ПОСТАНОВКА ЗАДАЧИ: Найти в массиве A, состоящем из n элементов, наибольший

элемент и запомнить его индекс (K).

**ProgramMaxElMassiva; usescrt;**

{ описание массива и переменных }

{ процедура ввода массива }

{ процедура вывода массива }

**procedure MaxEl;**

**begin**

**max:=a[1]; k:=1;**

**for i:=1 to n do if max<a[i] then begin max:=a[i]; k:=i; end;**

**writeln('Максимальный элемент массива А равен ',max);**

**writeln('Максимальный элемент массива стоит на ' ,k, ' месте);**

**end;**

**Begin**

**ClrScr; n:=20;**

**Input;**

**ClrScr;**

**Print; writeln;**

**MaxEl;**

**Readkey;;**

**End.**

#### *ПОИСК ЗАДАННОГО ЭЛЕМЕНТА В МАССИВЕ*

ПОСТАНОВКА ЗАДАЧИ: Определить, есть ли в заданном массиве элемент равный L.

Имеется массив А: 3 1 2 5 3 4 8 L=5

В массиве А имеется элемент равный 5

В данной программе можно использовать "флажок" - специальную переменную, которая принимает значения "истина" или "ложь" в зависимости от того найден элемент или нет.

**Program Poisk\_El\_Massiva; uses crt;**

. . .

**procedure Poisk(L: integer);**

**var fl: boolean;**

**begin**

**fl:=false;**

**for i:=1 to n do if a[i]=L then fl:=true;**

**if fl then write(' Элемент равный ',L,' имеется ')**

**else write(' Элемент равный ',L,' отсутствует');**

**end;**

**Begin**

**ClrScr; n:=20;**

**Input;**

**ClrScr;**

**Print; writeln; writeln;**

**write('Введитезначениеискомогоэлемента: '); readln(L);**

**Poisk(L);**

**Readkey;**

**End.**

#### *У П О Р Я Д О Ч Е Н И Е М А С С И В А*

ПОСТАНОВКА ЗАДАЧИ: Расположить элементы массива в порядке возрастания.

Из множества сортировок рассмотрим поиск MIN и перестановку.

Исходный массив: 3 2 -5 2 0 8 4

Конечный массив: -5 0 2 2 3 4 8

**ProgramSortirowkaMassiva; usescrt;**

. . .

**procedure Sort;**

**var min: real;**

**begin**

**for i:=1 to n do**

**begin**

**min:=a[i]; k:=i;**

**for j:=i to n do**

**if a[j]<min then begin min:=a[j]; k:=j; end;**

**a[k]:=a[i];**

**a[i]:=min**

**end**

**end;**

**Begin**

**ClrScr;**

**Input;**

**ClrScr;**

**write('Исходный массив: '); Print; writeln;**

**Sort;**

**write('Конечный массив: '); Print;**

**readkey;**

**End.**

**Практическая часть**

Задачи для самостоятельного решения:

1. Введите с клавиатуры в массив пять целочисленных значений. Выведите их в одну строку через запятую. Получите для массива среднее арифметическое.
2. Создайте массив А[1..7] с помощью генератора случайных чисел и выведите его на экран. Увеличьте все его элементы в 2 раза.
3. Создайте массив А[1..8] с помощью генератора случайных чисел с элементами от –10 до 10 и выведите его на экран. Подсчитайте количество отрицательных элементов массива.
4. Создайте массив А[1..12] с помощью генератора случайных чисел с элементами от –20 до 10 и выведите его на экран. Замените все отрицательные элементы массива числом 0.
5. Создайте целочисленный массив А[1..15] с помощью генератора случайных чисел с элементами от –15 до 30 и выведите его на экран. Определите самый большой элемент массива и его индекс.
6. Создайте массив из 15 целочисленных элементов и определите среди них минимальное значение.
7. Сформируйте линейный массив вещественных чисел, элементы которого являются расстояниями, пройденными телом при свободном падении на землю за 1, 2, … , 10 с.
8. Дан линейный массив целых чисел. Проверьте, является ли он упорядоченным по убыванию.
9. Найти сумму положительных элементов линейного массива целых чисел. Размерность массива – 10. Заполнение массива осуществить с клавиатуры.
10. Найти сумму четных элементов массива целых чисел. Размерность массива – 20. Заполнение массива осуществить случайными числами от 100 до 200.
11. Найти произведение элементов массива целых чисел, которые кратны 7. Размерность массива – 15. Заполнение массива осуществить случайными числами от 10 до 50.
12. Найти сумму элементов массива вещественных чисел, имеющих нечетные номера. Размерность массива – 20. Заполнение массива осуществить случайными числами от 100 до 200.
13. Найти произведение всех элементов массива целых чисел, меньших 0. Размерность массива – 10. Заполнение массива осуществить с клавиатуры.
14. Найти сумму всех элементов массива вещественных чисел, больших заданного числа. Размерность массива – 20. Заполнение массива осуществить случайными числами от 50 до 100.
15. Найти произведение всех элементов массива вещественных чисел, меньших заданного числа. Размерность массива –10. Заполнение массива осуществить случайными числами от 50 до 100.
16. Найти произведение элементов массива, кратных 3 и 9. Размерность массива – 10. Заполнение массива осуществить случайными числами от 5 до 500.
17. Найти сумму всех элементов массива целых чисел, которые меньше среднего арифметического элементов массива. Размерность массива –20. Заполнение массива осуществить случайными числами от 150 до 300.
18. Найти сумму элементов массива целых чисел, которые делятся на 5 и на 8 одновременно. Размерность массива –30. Заполнение массива осуществить случайными числами от 500 до 1000.
19. Найти произведение элементов линейного массива целых чисел, которые кратны 5. Размерность массива –10. Заполнение массива осуществить случайными числами от 10 до 100.
20. Рассортируйте заданный линейный массив по возрастанию.
21. Найти количество четных элементов одномерного массива.
22. Найти количество четных элементов одномерного массива до первого встреченного числа равного наперед заданному числу а.
23. Вычислить среднее арифметическое значение тех элементов одномерного массива, которые расположены за первым по порядку минимальным элементом.
24. В доме, состоящем из 30 квартир, переселить жильцов так, чтобы жильцы первой квартиры переехали в тридцатую, из тридцатой - в первую, из второй - в 29 и т.д., найдите количество квартир, в которых проживает более 5 человек.
25. Найти наибольший элемент из элементов одномерного массива, имеющих четный номер. Определить, является ли он единственным.

Задачи повышенного уровня:

* В сказочной стране Лукоморье в целях экономии топлива все грузы перевозят на Змей Горыночах, имеющих свой бортовой номер от 1 до 100. Количество голов Змея определяет его грузоподъемность. В архиве К. Бессмертного заведено личное дело на каждого Змея Горыноча с информацией о количестве голов и бортовом номере Однажды на Калиновом мосту грузовым Змеем Горыночем была сбита гражданка Баба Яга. Нарушитель скрылся, но свидетели показали, что число голов Змея кратно его бортовому номеру. Помогите Д. Никитичу следователю по ДТП выяснить нарушителя.
* В одном районе расположен тридцать населенных пунктов. По территории проходит железная дорога. По просьбе жителей района планируется построить железнодорожную станцию и проложить дороги так, чтобы сумма расстояний от всех населенных пунктов до станции была минимальной, если участок дороги, проходящей по территории района прямолинеен. Результат представить графически.
* Задан список фамилий брокеров товарной биржи из N человек. Обменяйте местами фамилии брокеров: первого и последнего, второго и предпоследнего, третьего от начала и третьего от конца и т.д.

**Контрольные вопросы:**

1. Какие из приведенных описаний одномерных массивов являются неправильными и почему:

а) var dim:array[-1..1] of real;

б) type mas=array[char] of char;

var beta:mas;

в) type massiv=array['A'..'D'];

var a,b:massiv;

г) var vector:array[integer] of char;

д) var mm:array[false..true] of char;

е) type ss=array[-20..0] of integer;

var mas=ss;

ж) type tr=array[1..n,1..m] of real;

var x,y,z:tr;

з) var k:integer;

pak:array[1..k] of integer;

1. Может ли типом индекса массива быть тип integer или real?
2. Дано:

const n=101;

var a:array[1..n] of real;

a) напишите операторы для ввода элементов в массив *а*;

б) напишите операторы для заполнения массива *а* случайными числами в интервале от -16 до 73;

в) напишите операторы для вывода элементов массива *а* в строку, разделяя их двумя пробелами;

г) напишите операторы для вывода элементов массива *а* в строку, разделяя их таким количеством пробелов, каков порядковый номер выводимого элемента;

д) напишите операторы вывода элементов массива в обратном порядке по *n* чисел в строке.

1. Чем одномерный массив отличается от двумерного?
2. Почему массивы называются структурами данных с прямым доступом?
3. Приведите примеры программ, где прямой доступ необходим.
4. Установите, какая задача решается в предложенном фрагменте программы. Назовите все операторы и типы данных, использованные в нем, расскажите как они работают:

const n=10;

var u,v,w:array [1..n] of integer;

j,k,t:integer;

begin k:=0;

for j:=n downto 1 do

begin t:=u[j]-v[j]+k;

w[j]:=t mod 10;

k:=t div 10;

end;end.

# **Практическое занятие №9.1 Обработка двухмерных массивов.**

**Теоретическая часть**

*О Б Р А Б О Т К А М А Т Р И Ц*

***1. СУММА ЭЛЕМЕНТОВ МАТРИЦЫ***

ПОСТАНОВКА ЗАДАЧИ: Для двумерного массива B размером N х M необходимо

вычислить сумму всех элементов матрицы.

**Program Summa\_Mat; uses crt;**

. . .

**procedure Sum;**

**begin**

**s:=0;**

**for i:=1 to n do**

**for j:=1 to m do**

**s:=s+b[i,j]**

**end;**

**Begin**

**ClrScr;**

**Input\_Mat;** { если массив вводится с клавиатуры }

**ClrScr;**

**Print\_Mat; writeln;**

**Sum;**

**write('Сумма элементов матрицы равна ',s);**

**readkey;**

**End.**

***2. ВЫЧИСЛЕНИЕ СЛЕДА МАТРИЦЫ***

СЛЕД МАТРИЦЫ - сумма диагональных элементов матрицы N х N.

ПОСТАНОВКА ЗАДАЧИ: Для матрицы В размером N х N надо вычислить

сумму диагональных элементов B[i,i].

. . .

**procedure Sled\_Mat;** { главнаядиагональ }

**begin**

**s:=0;**

**for i:=1 to n do s:=s+b[i,i]**

**end;**

. . .

**3. ВЫЧИСЛЕНИЕ СУММЫ ЭЛЕМЕНТОВ СТРОК МАТРИЦЫ**

ПОСТАНОВКА ЗАДАЧИ: Вычислить сумму элементов каждой строки матрицы В

размером N х M. Результат получить в виде вектора D.

. . .

**procedure Sum\_Strok\_Mat;**

**begin**

**for i:=1 to n do**

**begin**

**s:=0;**

**for j:=1 to m do**

**s:=s+b[i,j];**

**d[i]:=s**

**end**

**end;**

. . .

. . .

**Процедуры обработки матриц:**

**Сумма указанной строки:**

S:=0; i:=3;

For J:=1 to m do S:=S+b[i,j];

**Транспонирование квадратной матрицы:**

Fori:=1 ton-1 do

For j:=1 to n do

Begin

P:= a[i,j];

a[i,j]:=a[j,i];

a[j,i]:=p;

end;

**Удаление строки из матрицы:**

n:=n-1;

For i:=1 to n do

For j:=1 to m do b[i,j]:=b[i+1,j];

**Включение строки в матицу:**

i:=n;

while i>=k do

begin

for j:=1 to m do b[i+1,j]:=b[i,j];

i:=i-1;

end;

for j:=1 to m do b[k,j]:=c[j];

n:=n+1;

**Перестановка строк матрицы:**

For k:=1 to k do c[k]:=a[i,k];

For k:=1 to k do a[i,k]:=a[j,k];

For k:=1 to k do a[j,k]:=c[k];

**Поиск минимального элемента матрицы:**

Min:= a[1,1]; k;=1; L:=1;

For i:=1 to n do

For j:=1 to m do

IF min>a[i,j] then begin min:=a[i,j]; k:=i; l:=j; end;

**Сложение строк матрицы:**

Forj:=1 tomdo

a[k,j]:=a[k,j]+a[l,j]\*b;

**Практическая часть**

**Задачи для самостоятельного решения:**

1. Посчитать сумму некоторой строки транспонированной матрицы.
2. Транспонировать матрицу и вывести её на печать
3. Дана квадратная матрица. Найти: сумму элементов этой матрицы; сумму элементов главной диагонали; сумму элементов побочной диагонали; сумму элементов некоторой строки.
4. В двумерном массиве посчитать суммы отрицательных, положительных и количество нулевых элементов.
5. Поменять местами первую строку матрицы с последней.
6. Найти сумму положительных элементов указанного столбца двумерного массива. Транспонировать данную матрицу. Элементы главной диагонали заменить нулями.
7. Найти сумму элементов квадратной матрицы, расположенных ниже побочной диагонали и произведение элементов, расположенных выше главной диагонали. Транспонировать исходную матрицу.
8. Дана квадратная матрица размера n. Найти произведение элементов побочной диагонали.
9. Дана квадратная вещественная матрица размерности n. Найти количество нулевых элементов, стоящих выше главной диагонали.
10. Найти сумму положительных элементов указанного столбца двумерного массива. Транспонировать данную матрицу. Элементы главной диагонали заменить нулями.
11. Дана квадратная матрица. Найти сумму элементов, расположенных выше побочной диагонали транспонированной матрицы.
12. Даны две матрицы. Получить произведение всех элементов этих матриц.
13. Дана квадратная матрица размера n. Транспонировать ее. В полученной матрице найти сумму элементов заданной строки.
14. Найти сумму положительных элементов указанного столбца двумерного массива. Транспонировать данную матрицу. Элементы главной диагонали заменить нулями.
15. Найти сумму элементов из области, помеченной звездочками.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | \* |  |  |
|  |  | \* |  |  |
| \* | \* | \* | \* | \* |
|  |  | \* |  |  |
|  |  | \* |  |  |

1. Посчитать в двумерном массиве количество нечетных чисел. Если этот массив яв-ляется квадратной матрицей, то транспонировать его.
2. Дана матрица размерности N на M. Найти максимальный элемент и строку, содержащую этот элемент, поменять с первой строкой.
3. Составить таблицу сложения натуральных чисел от 1 до 9.
4. В квадратной матрице найти сумму элементов побочной диагонали и разделить на полученную сумму все элементы последнего столбца.
5. Дана квадратная матрица порядка N. В матрице вычислить среднее арифметическое положительных элементов, стоящих на главной диагонали.
6. Дана матрица размерности N на M. Положительные элементы матрицы переписать подряд в одномерный массив В.
7. Дана квадратная вещественная матрица размерности n. Найти количество нулевых элементов, стоящих выше и ниже побочной диагонали.
8. Найти сумму квадратов элементов главной диагонали.
9. Дана матрица размерности N на M. Вычислить количество строк матрицы, в которых есть хоть один отрицательный элемент.
10. Даны натуральное число n, действительная квадратная матрица порядка n, действительные числа a1,...,an+5. Элементы последовательности a1,...,an+5 домножить на 10, если наибольший элемент матрицы (в предположении, что такой элемент единственный) находится на главной диагонали, и на 0.5 в противном случае.

**Контрольные вопросы:**

1.Задан массив . Установите, каким будет значение *С* после выполнения операторов:

for i:=1 to 3 do

for j:=1 to 3 do C[i,j] := A[A[i,j],A[j,i]].

Существенен ли порядок, в котором выбираются значения индексов *i* и *j*?

1. Установите, какая задача решается в предложенном фрагменте программы. Назовите все операторы и типы данных, использованные в нем, расскажите как они работают:

for i:=p to q-(p-q+1) div 2 dobegin r:=a[i];

a[i]:=a[q-i+p];

a[q-i+p]:=r

end.

1. Установите, какая задача решается в предложенном фрагменте программы. Назовите все операторы и типы данных, использованные в нем, расскажите как они работают:

for i:=1 to n do for j:=1 to n do a[i,j]:=(i div j) \* (j div i).

1. Установите, какая задача решается в предложенном фрагменте программы. Назовите все операторы и типы данных, использованные в нем, расскажите как они работают:

for i:=1 to n-1 do for j:=i+1 to n do if a[i]>a[j]

then begin r:=a[i];

a[i]:=a[j];

a[j]:=r end.

1. Какую задачу решает предложенный фрагмент алгоритма? Определите значение массива *А* после выполнения следующих операторов при *N*=3 и :

for i:=1 to n-1 do

for j:=i+1 to n do

begin r:=a[i,j];

a[i,j]:=a[j,i];

a[j,i]:=r

end.

1. Какую задачу решает предложенный фрагмент программы? Определите значение массива *C* после выполнения следующих операторов при *N*=3, *M*=2 и :

for j:=1 to m do

begin s:=0;

for i:=1 to n do s:=s+a[i,j];

c[j]:=s

end.

1. Какую задачу решает предложенный фрагмент программы? Определите значение массива *C* после выполнения следующих операторов для заданных массивов  и :

for i:=1 to n do

for j:=1 to n do

if a[i,j]=b[i,j]

then c[i,j]:=1

else c[i,j]:=0.

1. Какую задачу решает предложенный фрагмент программы? Определите значение массива *A* после выполнения следующих операторов при *N*=8, *K*=2 и *А*=(4, -3, 5, -2, 3, 10, 9, 0):

n:=n-1;

for i:=k to n do a[i]:=a[i+1].

1. Какую задачу решает предложенный фрагмент программы? Определите значение массива *A* после выполнения следующих операторов при *M*=3, *N*=5, *D*=6 и *А*=(5, 2, -8, 1, -3):

i:=m;

while i<=n-1 do

begin a[i]:=a[i+1];

i:=i+1

end;

a[n]:=d.

# **Практическое занятие №9.2 Обработка двухмерных массивов.**

**Цель работы: научиться работать с двумерными массивами, выводить данные в виде таблицы.**

**Теоретическая часть**

Графическое представление матриц:

***Квадратнойназывается такая матрица***, в которой количество строк равно количеству столбцов. Выделяют следующие элементы квадратной матрицы:

* главная диагональ;
* побочная диагональ;
* элементы, расположенные выше главной диагонали;
* элементы, расположенные ниже главной диагонали;
* элементы, расположенные выше побочной диагонали;
* элементы, расположенные ниже побочной диагонали;

**Главная диагональ.** Если значения индексов (i, j) элемента равны, то элементы расположены на главной диагонали.

|  |  |  |  |
| --- | --- | --- | --- |
| А11 | А12 | А13 | А14 |
| A21 | A22 | A23 | А24 |
| A31 | A32 | A33 | А34 |
| A41 | A42 | A43 | А44 |

if i=j then <инструкции>

**Побочная диагональ.** Если для значений индексов (i, j) элементов выполняется равенство: i+j=n+1, то элементы расположены на побочной диагонали.

|  |  |  |  |
| --- | --- | --- | --- |
| А11 | А12 | А13 | А14 |
| A21 | A22 | A23 | А24 |
| A31 | A32 | A33 | А34 |
| A41 | A42 | A43 | А44 |

 if i+j=n+1 then <инструкции>

**Для элементов, расположенных выше главной диагонали** необходимо использовать один из следующих фрагментов программы:

|  |  |  |  |
| --- | --- | --- | --- |
| А11 | А12 | А13 | А14 |
| A21 | A22 | A23 | А24 |
| A31 | A32 | A33 | А34 |
| A41 | A42 | A43 | А44 |

for i:=1 to n do

for j:=1 to n do

if i < j then <инструкции>

for i:=1 to n-1 do

for j:=i+1 to n do

<инструкции>

**Если элементы расположены на главной диагонали и выше её** необходимо использовать следующий фрагмент программы:

|  |  |  |  |
| --- | --- | --- | --- |
| А11 | А12 | А13 | А14 |
| A21 | A22 | A23 | А24 |
| A31 | A32 | A33 | А34 |
| A41 | A42 | A43 | А44 |

for i:=1 to n do

for j:=1 to n do

if i<=j then <инструкции>

**Для элементов, расположенных ниже главной диагонали** необходимо использовать следующий фрагмент программы:

|  |  |  |  |
| --- | --- | --- | --- |
| А11 | А12 | А13 | А14 |
| A21 | A22 | A23 | А24 |
| A31 | A32 | A33 | А34 |
| A41 | A42 | A43 | А44 |

for i:=1 to n do

for j:=1 to n do

if i>j then <инструкции>

**Для элементов, расположенных ниже главной диагонали и не ней** необходимо использовать следующий фрагмент программы:

|  |  |  |  |
| --- | --- | --- | --- |
| А11 | А12 | А13 | А14 |
| A21 | A22 | A23 | А24 |
| A31 | A32 | A33 | А34 |
| A41 | A42 | A43 | А44 |

for i:=1 to n do

for j:=1 to n do

if i>=j then <инструкции>

**Если элементы, расположены выше побочной диагонали,** то необходимо использовать следующий фрагмент программы:

|  |  |  |  |
| --- | --- | --- | --- |
| А11 | А12 | А13 | А14 |
| A21 | A22 | A23 | А24 |
| A31 | A32 | A33 | А34 |
| A41 | A42 | A43 | А44 |

for i:=1 to n-1 do

for j:=1 to n-1 do

if i+j<=n then <инструкции>

**Если элементы, расположены ниже побочной диагонали,** то необходимо использовать следующий фрагмент программы:

|  |  |  |  |
| --- | --- | --- | --- |
| А11 | А12 | А13 | А14 |
| A21 | A22 | A23 | А24 |
| A31 | A32 | A33 | А34 |
| A41 | A42 | A43 | А44 |

for i:=2 to n do

for j:=2 to n-1 do

if i+j>n+1 then <инструкции>

### Транспонирование матрицы

***Транспонированной матрицей*** называется матрица, у которой столбцы соответствуют строкам исходной квадратной матрицы. При этом элементы главной диагонали исходной и транспонированной матриц, одни и те же.

Операция транспонирования сводится к обмену элементов матрицы, расположенных симметрично главной диагонали.

|  |  |
| --- | --- |
| **Исходная матрица** | **Транспонированная матрица** |
| |  |  |  |  | | --- | --- | --- | --- | | 1 | 2 | 3 | 4 | | 5 | 6 | 7 | 8 | | 9 | 10 | 11 | 12 | | 13 | 14 | 15 | 16 | | |  |  |  |  | | --- | --- | --- | --- | | 1 | 5 | 9 | 13 | | 2 | 6 | 10 | 14 | | 3 | 7 | 11 | 15 | | 4 | 8 | 12 | 16 | |

Фрагмент программы транспонирования матрицы:

for i:=1 to n do {Просмотр всех строк матрицы}

for j:=i+1 to n do {Просмотр всех элементов в строке, расположенных выше главной диагонали}

begin

k:=a[i,j];

a[i,j]:= a[j,i];

a[j,i]:= k;

end;

Примеры решения:

***Задачи***

1. Найти сумму всех элементов некоторого двумерного массива и сравнить их с произведением элементов некоторой строки.

program zadacha\_1;

uses crt;

var

a: array[1..50,1..50] of integer; {массив}

i,j: integer; {переменные счетчики}

n,m: integer; {количество строк и столбцов массива}

s: integer; {сумма элементов массива}

p: integer; {произведение элементов некоторой строки}

q: integer; {некоторая строка}

begin

clrscr;

write('Введите количество строк: ');

readln(n);

write('Введите количество столбцов: ');

readln(m);

for i:=1 to n do

for j:=1 to m do

begin

write('a[',i,',',j,']=');

readln(a[i,j]);

end;

writeln('Матрица:');

for i:=1 to n do

begin

for j:=1 to m do

begin

write(a[i,j]:3);

end;

readln;

end;

for i:=1 to n do

for j:=1 to m do

begin

s:=s+a[i,j];

end;

write('Введите номер строки для работы: ');

readln(q);

p:=1;

for j:=1 to m do

begin

p:=p\*a[q,j];

end;

writeln('Сумма элементов матрицы: ',s);

writeln('Произведение элементов строки ',q,' равна ',p);

if s>p then

begin

writeln('Суммабольшепроизведения');

end

else

begin

writeln('Произведение больше произведения');

end;

readln;

end.

2. Дана квадратная матрица. Транспонировать её. Посчитать сумму всех нечётных элементов транспонированной матрицы.

program zadacha\_4;

uses crt;

var

a:array [1..5,1..5] of integer;

S,i,j,n,c:integer;

begin

clrscr;

write ('введите кол-во строк и столбцов ');

readln (n);

for i:=1 to n do

for j:=1 to n do

begin

write ('a[',i,',',j,']=');

readln (a[i,j]);

end;

for i:=1 to n do

for j:=i+1 to n do

begin

c:=a[i,j];

a[i,j]:=a[j,i];

a[j,i]:=c;

end;

S:=0;

for j:=1 to n do

for i:=1 to n do

begin

if a[j,i] mod 2 <>0 then

begin

S:=S+a[j,i];

end;

end;

writeln ('S=',S);

readln;

end.

3.Дан двумерный массив. Посчитать сумму его двух столбцов, вывести большую сумму.

program zadacha\_5

uses crt;

var

a:array[1..3,1..3] of integer;

i,j,m,n,s,s1,p,p1,max,p3:integer;

begin

clrscr;

write('введите количество строк');

readln(n);

write('введите количество столбцов');

readln(m);

for i:=1 to n do

for j:=1 to m do

begin

write('a[',i,',',j,']=');

readln(a[i,j]);

end;

begin

write('введите номер столбца');

readln(p);

for i:=1 to n do

for j:=1 to m do

if j=p then s:=s+a[i,p];

write('s=',s);

readln;

end;

s1:=0;

begin

write('введите номер другого столбца');

readln(p1);

for i:=1 to n do

for j:=1 to m do

if j=p1 then s1:=s1+a[i,p1];

write ('s1=', s1);

readln;

end;

begin

max:=s;

if s < s1 then

begin

max:=s1;

write('максимальнаясумма',max);

readln;

write('вывести на экран номер столбца большей суммы', p3);

readln(p3);

end;

end;

end.

end.

1. Дана вещественная матрица размерности n\*m. Вывести номера столбцов, содержащих только отрицательные элементы.

program zadacha\_10;

uses crt;

var

a: array[1..50,1..50] of real;

i,j: integer;

n,m: integer;

begin

clrscr;

write('введите кол-во строк ');

readln(n);

write('введите кол-во столбцов ');

readln(m);

for i:=1 to n do

for j:=1 to m do

begin

write('a[',i,',',j,']= ');

readln(a[i,j]);

end;

for j:=1 to m do

begin

if a[i,j]<0 then

begin

writeln('номер столбца, в котором все элементы отрицательные= ',J);

readln;

end

else

begin

writeln ('в столбце ',J,' нет отрицательных или не

все отрицательные элементы ');

end;

end;

readln;

end.

**5.**В двумерном массиве найти минимальное число и определить в какой строке и каком столбце он находится.

program zadacha\_11;

uses crt;

var

a:array [1..50,1..50] of integer; {описаниемасива}

i,j:integer; {переменные-счетчики}

min: real; { минимальное число}

n,m: integer; { кол-во строк, кол-во столбцов}

begin {начало программы}

clrscr;

write('введите кол-во строк '); {ввод кол-ва строк}

readln(n);

write('введите кол-во столбцов '); {ввод кол-ва столбцов}

readln(m);

for i:=1 to n do

for j:=1 to m do

begin

write('a[',i,',',j,']= '); { ввод элементов массива }

readln(a[i,j]);

end;

min:=a[1,1];

for i:=1 to n do

for j:=1 to m do

begin

if a[i,j] < min then {поиск минимального числа}

min:=a[i,j];

end;

for i:=1 to n do

for j:=1 to m do

begin

if a[i,j]=min then

begin

writeln('минимальное число = ',min:5);

{вывод минимального числа}

writeln('номер элемента i=',i,',','j=',j);

{вывод номера мин. чис-ла}

end;

end;

readln;

end.

**6.**Развернуть квадратную матрицу на 90 градусов по часовой стрелке.

program zadacha\_12;

uses crt;

var

a: array [1..30,1..30] of integer; {исходнаяматрица}

b: array [1..30,1..30] of integer; {промежуточнаяматрица}

c: array [1..30,1..30] of integer; {Матрицаразвёрнутана 90?}

i,j: integer; {переменные счётчики}

m,n: integer; {количество строк и столбцов}

begin

clrscr;

write ('введите количество строк и столбцов ');

readln (n);

for i:=1 to n do {вводэлементовмассива}

for j:=1 to n do

begin

write ('a[',i,',',j,']=');

readln (a[i,j]);

end;

for i:=1 to n do

for j:=1 to n do

begin

b[i,j]:=a[n+1-i,j]; {промежуточной матрицы присваиваем

элементы первоначальной матрицы по закону: первому элементу

присваиваем строки последний, последнему первый, второму

элементу предпоследний, предпоследнему второй и тд.}

end;

writeln ('Матрица развёрнута на 90?.');

for i:=1 to n do

for j:=1 to n do

begin

c[i,j]:=b[j,i]; {третьей матрице присваиваем

элементы промежуточной по закону: первая строка

становится первым столбцом и тд. }

writeln ('c[',i,',',j,']=',c[i,j]); {печать массива развёрнутого на 90?}

readln;

end;

end.

**7.**Заменить положительные элементы прямоугольной вещественной таблицы их целыми частями, а отрицательные - их модулями.

program zadacha\_13;

uses crt;

var

a:array[1..50,1..100] of real;

i,j,n,m:integer;

begin

clrscr;

write('Введите кол-во строк');

readln(n);

write('Введите кол-во столбцов');

readln(m);

for i:=1 to n do

for j:=1 to m do

begin

write('a[',i,',',j,']=');

readln(a[i,j]);

end;

for i:=1 to n do

for j:=1 to m do

begin

if a[i,j]>0 then a[i,j]:=int(a[i,j]);

if a[i,j]<0 then a[i,j]:=abs(a[i,j]);

writeln ('a[',i,j,']=',a[i,j]:3:2);

readln;

end;

end.

**8.**Дана вещественная квадратная матрица размерности n. Поменять местами элементы главной и побочной диагоналей матрицы по столбцам

program zadacha\_14;

uses crt;

var

a:array [1..50,1..100] of integer;

i,j,n,t,k:integer;

begin

clrscr;

write('введите кол-во строк и столбцов');

readln(n);

for i:=1 to n do

for j:=1 to n do

begin

write('a[',i,',',j,']=');

readln(a[i,j]);

end;

for i:=1 to n do

for j:=1 to n do

begin

if i=j then

begin

k:=a[i,j];

a[i,j]:=a[i,n+1-j];

a[i,n+1-j]:=k;

end;

end;

for i:=1 to n do

begin

for j:=1 to n do

writeln(a[i,j]:4);

writeln;

end;

readln;

end.

**9.**Дана вещественная матрица размерности m\*n. Определить индексы первого нулевого элемента матрицы. Обходделатьпостолбцам.

program zadacha\_15;

uses crt;

var

a: array [1..50,1..50] of real;

m,n: integer; {количество строк и столбцов}

i,j: integer; {переменные счётчики}

k: integer; {переменная счётчик }

begin

clrscr;

write ('Введите количество строк в массиве ');

readln (n);

write ('введите количество столбцов в массиве');

readln (m);

for i:=1 to n do {вводмассива}

for j:=1 to m do

begin

write ('a[',i,',',j,']=');

readln (a[i,j]);

end;

for j:=1 to m do

for i:=1 to n do

begin

if a[i,j]=0 then{проверка условия равен ли проверяемый элемент 0}

begin

k:=k+1; {если да, то увеличиваем переменную k на1}

if k=1 then {проверка условия k=1}

begin

writeln ('i=',i,',j=',j); {если да, то печатать № строки и № столбца }

end;

end;

end;

readln;

end.

**10.**Заполните квадратную матрицу по образцу:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 25 | 24 | 23 | 22 | 21 |
| 16 | 17 | 18 | 19 | 20 |
| 15 | 14 | 13 | 12 | 11 |
| 6 | 7 | 8 | 9 | 10 |
| 5 | 4 | 3 | 2 | 1 |

program zadacha\_16;

uses crt;

var

a: array [1..30,1..30] of integer;

i,j,n,p,s,k: integer;

begin

clrscr;

write ('Введите количество строк и столбцов в матрице.');

readln (n);

k:=n\*n;

for i:=1 to n do

begin

if (i mod 2)<>0 then

begin

for j:=1 to n do

begin

a[i,j]:=k;

k:=k-1;

end;

end

else

begin

for j:=n downto 1 do

begin

a[i,j]:=k;

k:=k-1;

end;

end;

end;

for i:=1 to n do

begin

for j:=1 to n do

writeln (a[i,j]:5);

readln;

end;

end.

**11.**Сформировать матрицу по образцу:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 |
| 2 | 1 | 2 | 3 | 4 |
| 3 | 2 | 1 | 2 | 3 |
| 4 | 3 | 2 | 1 | 2 |

program zadacha\_17;

var n,i,j:integer; a:array [1..100,1..100] of integer;

begin write('введите n:'); readln(n);

for i:=1 to n do

for j:=1 to n do

begin

a[i,j]:=j-i+1;

if a[i,j]<=0 then a[i,j]:=abs(a[i,j]-2);

end;

for i:=1 to n do

begin for j:=1 to n do

write(a[i,j]:3);

writeln

end

end.

**12.**Сформировать матрицу по образцу:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1 | 1 | 1 | 1 | 1 |
| 0 | 1 | 1 | 1 | 0 |
| 0 | 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 1 | 0 |
| 1 | 1 | 1 | 1 | 1 |

program zadacha\_18;

var a:array[1..100,1..100] of integer;

i,j,n:integer;

begin write('Введите количество строк и столбцов');

readln(n);

for i:=1 to n div 2 do

for j:=i to n do

begin a[i,j]:=0;

if ((j>=i) and (j<=(n-i+1))) and (i<=(n div 2)+1)

then begin a[i,j]:=1;

a[n-i+1,j]:=1;

end;

end;

if n mod 2=1 then a[n div 2+1,j div 2+1]:=1;

for i:=1 to n do

begin for j:=1 to n do write(a[i,j]:2);

writeln

end;

readln

end.

**13.**Дана матрица размеренности m\*n, в которой не все элементы равны нулю. Получить новую матрицу путем деления всех элементов данной матрицы на ее наибольший по модулю элемент.

program zadacha\_19;

uses crt;

var

a:array[1..100,1..100] of integer; {исходнаяматрица}

b:array[1..100,1..100] of single; {конечнаяматрица}

i:integer; {номерстроки}

j:integer; {номерстолбца}

n:integer; {количество строк}

m:integer; {количество столбцов}

max:integer;{наибольший по модулю элемент}

begin

clrscr;

writeln ('Количество строк');

readln(n);

writeln ('Количество столбцов');

readln(m);

for i:=1 to n do {ввод элементов исходной матрицы}

for j:=1 to m do

begin

write('a[',i,',',j,']=');

readln(a[i,j]);

end;

writeln('Исходная матрица:');

for i:=1 to n do {вывод элементов матрицы}

begin

for j:=1 to m do

begin

write(a[i,j]);

end;

writeln;

end;

max:=abs(a[1,1]); {нахождение наиб. элемента по модулю}

for i:=1 to n do

for j:=1 to m do

begin

if abs(a[i,j])>max then

max:=abs(a[i,j]);

end;

writeln('Наибольший по модулю элемент: ',max);

for i:=1 to n do {деление каждого эл. нового массива на max}

for j:=1 to m do

begin

b[i,j]:=a[i,j]/max;

end;

writeln('Конечная матрица:');

for i:=1 to n do {вывод элементов конечной матрицы}

begin

for j:=1 to m do

begin

write(b[i,j]);

end;

writeln;

end;

readln;

end.

**14.**Дана квадратная вещественная матрица размерности n. Является ли матрица симметричной относительно главной диагонали

program zadacha\_20;

uses crt;

var

a:array[1..50,1..50]of integer;{описываюмассивиданные}

n:integer;

i,j:integer;

begin

clrscr;

writeln('введите кол-во строк и столбцов'); {ввожу кол-во строк и столбцов}

readln(n);

for i:=1 to n do

for j:=1 to n do

begin

write('a[',i,',',j,']=');{ввожучисла}

readln(a[i,j]);

end;

begin

for i:=1 to n-1 do {транспонируюматрицу}

for j:=i+1 to n do

end;

if a[i,j]=a[j,i] then {проверяюусловие}

writeln('матрица симетричная'){вывожу ответ на экран}

else

writeln('матрица не симетричная');

readln;

end.

**15.**Дана действительная квадратная матрица порядка n. Вычислить сумму тех из m элемен-тов, расположенных по главной диагонали и выше неё, которые превосходят по величине все элементы, расположенные ниже главной диагонали. Если элементов с указанным свойством нет, то вывести соответствующее сообщение.

program zadacha\_21;

uses crt;

var

a: array[1..50,1..50] of real;

i,j: integer;

n: integer;

max: real;

s: real;

begin

clrscr;

writeln ('введите кол-во строк и столбцов');

readln (n);

for i:=1 to n do

for j:=1 to n do

begin

write('a[',i,',',j,']=');

readln(a[i,j]);

end;

s:=0;

max:=0;

for i:=1 to n do

for j:=1 to n do

begin

if i>j then

begin

if a[i,j]>max then

begin

max:=a[i,j];

end;

end;

end;

for i:=1 to n do

for j:=1 to n do

begin

if (i<=j) then

begin

if a[i,j]>max then

begin

s:=s+a[i,j];

end;

end;

end;

if s<>0 then

begin

write ('s=',s:5:2);

end

else

begin

writeln ('Элементов с указанным свойством нет');

end;

readln;

end

**16.**Имеется квадратная матрица. Повернуть её на 180 градусов

program zadacha\_22;

uses crt;

var

a,x: array [1..50,1..50] of integer;

i,j: integer;

n: integer;

begin

randomize;

clrscr;

writeln('Введите размерность массива');

readln(n);

for i:=1 to n do

for j:=1 to n do

begin

a[i,j]:=random(100);

end;

for i:=1 to n do

begin

for j:=1 to n do

write (a[i,j]:4);

writeln;

end;

writeln('Новыймассив');

for i:=1 to n do

for j:=1 to n do

begin

x[i,j]:=a[n-i+1,n-j+1];

end;

for i:=1 to n do

begin

for j:=1 to n do

write (x[i,j]:4);

writeln;

end;

readln;

end.

**Практическая часть**

**Задачи для самостоятельного решения:**

1. Дана вещественная матрица размерности n \* m. Вывести номера строк, содержащих больше положительных элементов, чем отрицательных.
2. Дана матрица размерности N на M. Найти строку, в которой максимальный элемент минимален в соответствующем столбце.
3. Сформировать матрицу по следующему правилу:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1 | 1 | 1 | 1 | 1 |
| 2 | 3 | 1 | 1 | 1 |
| 4 | 5 | 6 | 1 | 1 |
| 7 | 8 | 9 | 10 | 1 |
| 11 | 12 | 13 | 14 | 15 |

1. Найти количество элементов двумерного массива, превосходящих 10.
2. Дана вещественная матрица размерности n \* m. Поменять местами строки с максимальным и минимальным элементами.
3. Сделать циклический сдвиг столбцов таблицы на 2 позиции вперед.
4. Дана вещественная матрица размерности m \* n. Определить индексы первого нулевого элемента матрицы. Обход матрицы осуществлять по столбцам.
5. Дана вещественная квадратная матрица размерности n. Поменять местами элементы главной и побочной диагоналей матрицы по строкам;
6. Дана целочисленная квадратная матрица порядка 8. Найти наименьшее из значений элементов столбца, который обладает наибольшей суммой модулей элементов. Если таких столбцов несколько, то взять первый из них.
7. Поменять второй столбец матрицы с предпоследним.
8. Дана матрица размерности N на M. Найти максимальный элемент и строку, содержащую этот элемент, поменять с первой строкой. Полученную матрицу вывести построчно.
9. Дана действительная матрица размера n \* m. Найти среднее арифметическое элементов каждой строки, имеющей четный номер.
10. Дана действительная квадратная матрица порядка n. Выяснить, верно ли, что наибольшее из значений элементов главной диагонали больше, чем наименьшее из значений элементов побочной диагонали.
11. Дана матрица размерности N на M. В матрице найти первый по порядку столбец с максимальной суммой элементов. Вывести его номер.
12. Найти сумму элементов из области, помеченной звездочками.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| \* | \* | \* | \* | \* |
|  | \* | \* | \* | \* |
|  |  | \* | \* | \* |
|  |  |  | \* | \* |
|  |  |  |  | \* |

1. Дана вещественная квадратная матрица размерности n. Поменять местами элементы главной и побочной диагоналей матрицы по столбцам.
2. Дана матрица размерности N на M. Найти столбец, в котором максимальный элемент минимален в соответствующей строке.
3. Сформировать матрицу по следующему правилу:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 |
| 6 | 7 | 8 | 9 | 10 |
| 11 | 12 | 13 | 14 | 15 |
| 16 | 17 | 18 | 19 | 20 |
| 21 | 22 | 23 | 24 | 25 |

1. Дана вещественная матрица размерности m \* n. Расположить все элементы матрицы по убыванию. Обход матрицы осуществлять по строкам.
2. Дана матрица размерности N на M. Вычислить количество строк матрицы, элементы которых упорядочены по возрастанию.
3. Дана вещественная матрица размерности n \* m. Найти общую сумму элементов только тех столбцов, которые имеют хотя бы один нулевой элемент.
4. Дана квадратная матрица порядка N. Найти максимальный и минимальный элементы матрицы и поменять местами соответствующие им строку и столбец (строка для максимального элемента, столбец для минимального элемента).
5. Дана квадратная матрица порядка N. Найти количество четных элементов квадратной матрицы, расположенных ниже побочной диагонали.
6. Дана действительная матрица размера m\* n. Найти сумму наибольших значений элементов ее строк.
7. Дана вещественная матрица размерности m \* n. Упорядочить элементы каждой четной строки по возрастанию.
8. Дана вещественная матрица размерности m \* n. Расположить все элементы матрицы по убыванию. Обход матрицы осуществлять по строкам.
9. Дана вещественная матрица размерности m \* n. Определить индексы первого нулевого элемента матрицы. Обход матрицы осуществлять по столбцам.
10. Известно положение двух ферзей на шахматной доске. Бьют ли они друг друга?
11. Для заданной целочисленной квадратной матрицы определить, является ли сумма ее элементов, расположенных выше главной диагонали, четным числом, и вывести на печать соответствующий текст.
12. Дана матрица размера m х n. Посчитать количество элементов массива, кратных наименьшему элементу некоторой квадратной матрицы. Элементы, расположенные на побочной диагонали квадратной матрицы увеличить на наибольший элемент матрицы размера m\*n. Полученные матрицы вывести на экран.
13. Дана две матрица: квадратная и размером 2\*n х m. Третью строку квадратной матрицы поменять с последней строкой второй матрицы, при условии, если количество столбцов этих матриц совпадает. Посчитать сумму всех четных элементов, расположенных выше главной диагонали квадратной матрицы и произведение нечетных элементов, расположенных ниже побочной диагонали.
14. Дана две квадратные матрица. Найти произведение всех элементов первой матрицы, которые кратные наибольшему элементу, расположенного на главной диагонали второй матрицы. Все отрицательные элементы второй матрицы заменить их абсолютными величинами. Транспонированные матрицы вывести на экран.
15. Имеется квадратная матрица. Повернуть ее на 180 градусов.
16. Заполнить квадратную матрицу по следующей схеме:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 |
| 8 | 9 | 10 | 11 | 12 | 13 | 14 |
| 21 | 20 | 19 | 18 | 17 | 16 | 15 |
| 22 | 23 | 24 | 25 | 26 | 27 | 28 |
| 35 | 34 | 33 | 32 | 31 | 30 | 29 |
| 36 | 37 | 38 | 39 | 40 | 41 | 42 |
| 49 | 48 | 47 | 46 | 45 | 44 | 43 |

1. Дана матрица С размером 4х5. Вставить после третьего столбца массив В={b1,b2,b3,b4}, а затем после первой строки вставить массив А={а,1а2, а3, а4, а5, а6}, таким образом получить матрицу размерностью 5х6.
2. Заполнить квадратную матрицу по следующей схеме:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 0 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 1 | 1 | 1 | 0 | 0 |
| 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 1 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 0 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 |

**Контрольные вопросы:**

1. Какую задачу решает предложенный фрагмент программы? Определите значение массива *B* после выполнения следующих операторов при *N*=7 и *А*=(1, 6, 7, 3, -2, 8, 9):

for i:=1 to n do b[i]:=a[n-i+1].

1. Какую задачу решает предложенный фрагмент программы? Определите значение массива *P* после выполнения следующих операторов при *M*=3, *N*=3 и :

for i:=1 to n do

begin p[i]:=0;

for j:=1 to m do

if a[i,j]=5

then p[i]:=p[i]+1

end.

1. Какую задачу решает предложенный фрагмент программы? Определите значение массива *C* после выполнения следующих операторов при *M*=9 и *C*=(4, 5, 3, 2, 2, 3, 5, 8, 7):

k:=trunc(m/2);

for i:=1 to k do

begin q:=c[i];

c[i]:=c[m-i+1];

c[m-i+1]:=q

end.

1. Какую задачу решает предложенный фрагмент программы? Определите значение массива *A* после выполнения следующих операторов при *N*=8, *K*=3 и *А*=(3, -1, 9, -3, 4, 12, 3, 7):

n:=n-1;

for i:=k to n do a[i]:=a[i+1].

1. Какую задачу решает предложенный фрагмент программы? Определите значение массива *S* после выполнения следующих операторов при *N*=5 и :

for i:=1 to n do s[i]:=a[i,i].

1. Какую задачу решает предложенный фрагмент программы? Определите значение переменной *S* после выполнения следующих операторов при *N*=6, *T*=5 и *А*=(-4, 0, 3, 9, 5, 2):

s:=0;

k:=0;

for i:=1 to n do

if a[i]>t

then begin s:=s+a[i];

k:=k+1

end;

if k>0 then s:=s/k.

# **Практическое занятие №10 Использование стандартных функций для работы с массивами.**

# **Цель работы: научиться решать задачи содержащие функции для работы с массивами.**

# **Теоретическая часть:**

# **1.** Типизированные константы - массивы

В качестве начального значения используется список констант, отделенных друг от друга запятыми. Список заключается в круглые скобки.

Примеры:

1) ConstMas:array[1..10] ofbyte=(1,1,1,1,1,1,1,1,1,1); - заполнение массива из 10 целых чисел значением, равным единице.

1. Const massim: array[0..5] of char = ('a','b','c','d','e','f'); - заполнение

массива из 6 элементов символами - буквами латинского алфавита.

1. Const Matr: array[1..5,1..2]of byte = ((0,0),(0,0),(0,0),(0,0),(0,0)); -

обнуление матрицы из 10 целых чисел.

Замечание: количество переменных в списке констант должно строго соответствовать объявленной длине массива по каждому индексу!

### Заполнение массива данными (ввод элементов)

***Массив, описанный как типизированная константа, уже содержит данные***. Массивы, объявленные в разделе описания переменных, необходимо заполнить данными, прежде чем выполнять с ними какие-либо действия.

Значения элементов массива также можно задать следующими способами:

при вводе данных с клавиатуры:

write('Введите количество строк и столбцов');

readln(n,m);

for i:=1 to n do

for j:=1 to m do

begin

write('a[',i,',',j,']='); {Можно эту строчку в программе не использовать}

readln(a[i,j]);

end;

с помощью датчика случайных чисел:

randomize;

writeln('Введите количество элементов массива');

readln(n);

for i:=1 to n do

begin

a[i]:=random(50);

writeln('a[',i,',',j,']=',a[i,j]);

end;

присваением заданных значений (например по формуле i\*i/i+2):

writeln('Введите количество элементов массива');

readln(n);

for i:=1 to n do

begin

a[i]:=i\*i/i+2;

writeln('a[',i,',',j,']=',a[i,j]);

end;

### Выводмассива

выводвстолбец:

for i:=1 to n do

writeln(a[i,j]);

построкамистолбцам:

for i:=1 to n do

begin

for j:=1 to m do

begin

write(a[i,j]:3);

end;

readln;

end;

### 3.Обработкамассивов

Часто требуется вычислить сумму элементов массива, их среднее арифметическое значение или найти значения и номера максимального и минимального элементов, а также изменить значения элементов массива и т.д. Особенность работы с двумерными массивами заключается в том, что расширяется возможность обработки массива (появились новые элементы: строки, столбцы - являющиеся одномерными массивами). Подробно все действия можно рассмотреть в [задачах](http://pascal.yaguo.ru/product/82.html#822) разобранных в этом разделе.

Примеры решения подобных задач:

1. Дана матрица размерности m\*n. Расположить элементы последнего столбца по убыванию.

program zadacha\_3;

uses crt;

var

a:array [1..50] of integer;

b:array [1..50] of integer;

k,i,m,j,n,r,l:integer;

begin

clrscr;

write('Введите количество строк');

readln(n);

write('Введите количество столбцов');

readln(m);

for i:=1 to n do

for j:=1 to m do

begin

write('a[',i,']={b[',j,']=}');

readln(a[i]);

end;

for i:=1 to n-1 do

for k:=i+1 to n do

{for j:=1 to m do}

if a[k]>a[i] then

begin

r:=a[i];

a[i]:=a[k];

a[k]:=r;

end;

writeln('Отсортированый массив:');

for i:=1 to n do

writeln(a[i]:4);

readln;

end.

1. Дана вещественная матрица размерности n\*m. Удалить k столбец матрицы.

program zadacha\_5;

uses crt;

var

a: array [1..100,1..100] of real;

b: array [1..100,1..100] of real;

i,j: integer; {переменныесчётчики}

n,m: integer; {количество строк и столбцов в массиве}

k: integer; {№ строки которую необходимо удалить}

begin

clrscr;

write ('Введите количество строк в массиве');

readln (n);

write ('введите количество столбцов в массиве');

readln (m);

write ('Введите № строки которую надо удалить');

readln (k);

randomize; {ввод массива случайных чисел}

for i:=1 to n do

for j:=1 to m do

begin

a[i,j]:=random(100);

end;

for i:=1 to n do

for j:=1 to m do

begin

writeln ('a[',i,',',j,']=',a[i,j]);

end;

writeln ('Новыймассив');

for i:=1 to n do

for j:=1 to m do

begin

if j<>k then

{Проверка условия № столбца

неравен № столбца,

который необходимоудалить}

begin

b[i,j]:=a[i,j];

{если да, то новому массиву

присваиваем проверяемый элемент }

end;

end;

for i:=1 to n do {выводновогомассива}

for j:=1 to m do

writeln ('b[',i,',',j,']=',b[i,j]);

readln;

end.

1. Поменять второй столбец матрицы с предпоследним.

program zadacha\_2;

uses crt;

var

a: array [1..50,1..50] of integer;

b: array [1..50,1..50] of integer;

m,n,i,j: integer;

begin

clrscr;

writeln('Количество строк');

readln(n);

writeln('Количество столбцов');

readln(m);

for i:= 1 to n do

for j:= 1 to m do

begin

write ('a[',i,',',j,']=');

readln (a[i,j]);

end;

writeln('Исходнаяматрица:');

for i:=1 to n do begin

for j:=1 to m do

write (a[i,j]);

writeln;

end;

for i:=1 to n do begin

for j:=1 to m do

b[i,j]:=a[i,j];

end;

for i:=1 to n do begin

a[i,2]:=b[i,m-1];

end;

for i:=1 to n do begin

a[i,m-1]:=b[i,2];

end;

writeln('Полученнаяматрица:');

for i:=1 to n do begin

for j:=1 to m do

write (a[i,j]);

writeln;

end;

readln;

end.

# **Практическая часть:**

Задачи для самостоятельного решения:

1. Создайте двумерный массив целых чисел. Удалите из него строку и столбец, на пересечении которых расположен минимальный элемент.
2. Удалить столбец двумерного массива целых чисел, в котором находится максимальный элемент этого массива.
3. Дан целочисленный массив X[1..5, 1..4]. Заменить в нем все элементы, меньшие 5 числом 111.
4. Дан целочисленный массив В[1..4, 1..3]. Заменить все его элементы их квадратами.
5. Дан действительный массив А[1..5, 1..3]. Напечатать индексы его отрицательных элементов.
6. Создайте двумерный целочисленный массив А[1..10, 1..7] и найдите сумму всех его элементов, имеющих оба нечетных индекса.
7. Дан целочисленный массив А[1..5, 1..3]. Вывести на экран элементы массива, кратные 5, и их индексы.
8. Введите с клавиатуры целочисленные элементы матрицы 3\*3 и вычислите сумму элементов каждого столбца.
9. Дан целочисленный массив В[1..5, 1..5]. Напечатать его левую и правую диагонали.
10. Дан целочисленный массив В[1..5, 1..5]. Найти сумму элементов его правой диагонали и произведение элементов левой диагонали.
11. Дан целочисленный массив В[1..5, 1..5]. Найти наибольший элемент массива и сообщить его расположение относительно левой диагонали.
12. Дан целочисленный массив В[1..5, 1..5]. Найти наименьший элемент массива и сообщить его расположение относительно правой диагонали.
13. Дан целочисленный массив В[1..5, 1..5]. Вычислить сумму элементов этого массива, расположенных выше левой диагонали.
14. Дан целочисленный массив В[1..5, 1..5]. Вычислить произведение элементов этого массива, расположенных ниже левой диагонали.
15. Дан целочисленный массив В[1..5, 1..5]. Найти количество отрицательных элементов этого массива, расположенных ниже левой диагонали.
16. Дан целочисленный массив В[1..5, 1..5]. Найти количество положительных элементов этого массива, расположенных выше левой диагонали.
17. Дана таблица со сведениями о наличии сырья на каждом из 4-х участков пяти цехов. Определить номер цеха, в котором меньше всего сырья.
18. Дан массив А[1..3, 1..15]. Известно, что среди его элементов два и только два равны между собой. Напечатать их индексы.
19. Даны целые числа а1, а2, а3. Получить целочисленный массив В[1..3, 1..3], для которого B[i,j]=ai-3aj.
20. Даны действительные числа а1, а2, …, а10, b1, b2, …, b20. Получить действительный массив С[1..20, 1..10], для которого С[i,j]=aj/(1+|bi|).
21. Получить целочисленный массив А[1..10, 1..12], для которого aij=i+2j.
22. Создайте матрицу 5\*5, значение каждого элемента которой равно сумме номера строки и столбца, на пересечении которых он находится, и вычислите сумму элементов каждой строки.
23. Получить действительный массив A[1..7, 1..7], первая строка которой задается формулой a1j=2j+3, вторая строка задается формулой a2j=j+3/(2+j), а каждая следующая строка есть сумма двух предыдущих.
24. Дано натуральное число n. Выяснить, сколько положительных элементов содержит массив A[1..n, 1..n], если aij= sin(i+j/2).
25. Дан целочисленный массив А[1..4, 1..5]. Найти среднее арифметическое каждого из столбцов.

**Контрольные вопросы**:

1. Какую задачу решает предложенный фрагмент программы? Определите значение массива *C* после выполнения следующих операторов для заданных массивов *N*=3,  и :

for i:=1 to n do

for j:=1 to n do

if a[i,j]>b[i,j]

then c[i,j]:=a[i,j]-b[i,j]

else c[i,j]:=b[i,j]-a[i,j].

1. Сформулируйте задачу, решаемую в предложенном фрагменте программы, где a[i,j] – элемент массива размерности . Определите значение массива *А* при *N*=4, *M*=5, *K*=3, :

fori:=1 tondo

for j:=k to m-1 do a[i,j]:=a[i,j+1].

1. Задан одномерный массив *А*=(7, 5, 4, 6, 3, 2, 1). Какое значение будет выведено в результате выполнения программы:

var a:array[1..7] of integer;

i,j,k,c:integer;

begin write('Введите семь элементов массива ');

for k:=1 to 7 do read(a[k]);

i:=3; c:=0;

for k:=1 to 100 do

begin j:=a[i]; a[i]:=i; i:=j; c:=c+i end;

write(c)

end.

1. Какое значение будет выведено в результате выполнения программы:

var i,k,k1,c : longint;

begin k:=3; k1:=1; c:=80;

for i:=64 to 174 do

begin k1:=-k1; k:=k+k1; c:=c+k\*i end;

write(c)

end.

**Практическое занятие №11 Работа со строковыми переменными.**

**Цель работы: научиться работать со строковыми переменными.**

**Теоретическая часть**

Различают величины:

1. **Константы** - последовательность символов в апострофах.

2. **Символьные переменные** типа **CHAR** (кэр, чар) - переменные принимают значения символов: ...var a:char;...а:='м';...

3. **Строковые переменные** типа **STRING** - цепочки символов:.var а:string;

Максимальное число символов в цепочке равно 255.

1**. LENGTH(S)** -(лэнкс) - определяет и присваивает ДЛИНУ строковой величины S

( количество символов, включая пробел ). Результат – целое число.

2. **CONCAT(S1,S2)** -(конкат) - ОБЪЕДИНЕНИЕ (склеивание) символьных и строковых

величин S1 и S2. Конкатенация производится также с помощью знака **+** ,например S:='Ва'+'ся' Результат - строка ‘Вася’.

3. **POS(ST,S)** -(позишэн) - определяет и присваивает НОМЕР ПОЗИЦИИ,

с которой подстрока ST входит в строку S.

Результат - число, например pos('о','колесо') =2.

4. **COPY(S,P,D)** -(копи) - ВЫБИРАЕТ часть строковой величины S,

начиная с позиции P, длиной D символов.

Результат - выбранная строка или один символ.

ПРИМЕРЫ:

**Program Primer1; Uses crt; Program Primer2; Uses crt;**

**var b: string; var a,b,c: string;**

**y: integer; Begin**

**Begin clrscr;**

**clrscr; a:=' погода';**

**b:='информатика'; b:=' хорошая';**

**y:=length(b); c:=a+b;**

**writeln('Длина строки =',y); writeln(c)**

**End. End.**

**Program Primer3; Uses crt; Program Primer4; Uses crt;**

**var a: string; var a,b: string;**

**y: integer; Begin**

**Begin clrscr;**

**clrscr; a:='информатика';**

**a:='информатика'; b:=copy(a,3,5);**

**writeln(a); writeln(a);**

**y:=pos('р',a); writeln(b) { форма }**

**writeln('рна ',y,' месте'); End.**

**End.**

*СИМВОЛЬНЫЕ И СТРОКОВЫЕ ФУНКЦИИ (продолжение)*

**CHR(X)** -(си эйч а) - ПРИСВАИВАЕТ **СИМВОЛ** соответствующий коду X.

CHR(65) соответствует латинской заглавной букве А.

**ORD(S)** -(ода) - ПРИСВАИВАЕТ **НОМЕР** символа из упорядоченной последовательности символов (см. ASCII), например:ORD(‘B’)=66.

#### *ПРОЦЕДУРЫ НАД СТРОКОВЫМИ ВЕЛИЧИНАМИ*

1. **DELETE(S,POS,NUM)** -(дилит) - УДАЛЯЕТ из строки S участок, содержащий NUM символов с позиции POS.

2. **INSERT(OBJ,S,POS)** -(инсет) - ВСТАВИТЬ строку OBJ (переменную или константу) в строку S(переменная) с позиции POS (целое).

ПРОЦЕДУРЫ ПРЕОБРАЗОВАНИЯ ТИПОВ:

3. **STR(VALUE,ST)** -(стр) - ПРЕОБРАЗУЕТ числовое значение в СТРОКУ символов и запоминает результат в строке ST.

4. **VAL(ST,VAR,CODE)** -(вэл) - ПРЕОБРАЗУЕТ строковую величину ST в ЧИСЛОВУЮ величину (в зависимости от VAR). CODE - целая величина. Code=0 ,если нет ошибки, иначе Code равно Pos ошибочного символа, начиная с апострофа.

**Program Primer5; Uses crt; Program Primer6; Uses crt;**

**var a: string; var a: string;**

## Begin Begin

**clrscr; clrscr;**

**a:='Информатика'; a:='Инка';**

**writeln(a); writeln(a);**

**delete(a,3,7); insert('формати',a,3);**

**writeln(a); writeln(a)**

**End. End.**

**Program Primer7; Uses crt; Program Primer8; Uses crt;**

**var a,b: string; var a: string;**

**c: integer; r,x,y: integer;**

**Begin Begin**

**clrscr; clrscr;**

**c:=100; a:='100 лет';**

**a:='Ценаруб.'; val(a,x,r);**

**str(c:4,b); y:=5\*x;**

**insert(b,a,5); write('Y=',y);**

**writeln(a); readkey;**

**End. End.**

Изменить цену товара, применив: copy, delete, val, str, insert.

**Program Primer9; uses crt;**

**var a,b: string;**

**x,y,p,r: integer;**

**Begin**

**clrscr;**

**a:='Цена товара 100 руб.';** { начальная фраза }

**p:=pos('1',a);**  { позиция цифры 1 }

**b:=copy(a,p,3);** { вырезка числа 100 }

**delete(a,p,3);** { удаление числа 100 из фразы }

**val(b,x,r);** { преобразование '100' в число 100 }

**y:=2\*x;** { изменение числа 100 }

**str(y:4,b);** { преобразование 200 в константу '200' }

**insert(b,a,p);** { вставка '200' }

**writeln(a);**  { печать полученной фразы }

**readkey;**

**End.**

##### З А Г А Д К А

Задать пользователю загадку и похвалить его за правильный ответ.

**Program Primer10; uses crt;**

**var otg,name,a: string;**

**Begin**

**clrscr;**

**write('Здравствуйте. Как Вас зовут ? '); readln(name);**

**clrscr; writeln;**

**writeln(name,', отгадайтезагадку: ');**

**a:='Висит груша,нельзя скушать.';**

**repeat**

**write(a,' Отгадка: '); readln(otg); clrscr; {только READLN}**

**until (otg='лампа') or (otg='лампочка');**

**writeln;**

**writeln;**

**writeln('Молодец, ',name,' ! До свидания !');**

**readkey;**

**End.**

Получить перевертыш: ОГОРОД - ДОРОГО.

**ProgramPrimer11; usescrt;**

**var a: string;**

**i,l: integer;**

**Begin**

**clrscr;**

**write('Введитеслово :'); readln(a);**

**clrscr; { for i:=length(a) downto 1 do write(a[i]); }**

**writeln(a);**

**l:=length(a);**

**for i:=l downto 1 do writeln(copy(a,i,1));**

**readkey;**

**End.**

Очень давно слово "АБРАКАДАБРА" записывали определенным образом на ткани и это заклинание носили в виде шарфика вокруг шеи. Считалось, что заклинание защищает от малярии и других болезней. Получить эту запись в виде треугольника.

**Program Primer12; usescrt;** Программу можно написать

**var a: string;** и другими способами.

**i, k: integer;**

**Begin** Можете подумать на досуге.

**clrscr;**

**a:='a b r a k a d a b r a';**

**k:=0;**

**for i:=1 to 11 do**

**begin**

**gotoxy(30+i,9+i);**

**delete(a,22-k,2);**

**write(a);k:=k+2**

**end;**

**readkey;**

**End.**

Задача: Ввести с клавиатуры предложение.

1. Подсчитать в нем количество всех символов.

2. Подсчитать количество букв "а".

3. Заменить запятые пробелами.

4. Осуществить движение первых двух символов вниз.

**Program Simwol; Usescrt;**

**var a,** { начальное предложение }

**b,** { промежуточная переменная }

**c: string;** { конечное предложение }

**i,** { переменная цикла }

**k,**  { счетчик символов "а" }

**l: integer;** { счетчик символов }

**Begin**

**clrscr;**

**write('Введите предложение :'); readln(a);**

**l:=length(a);**

**writeln('В предложении содержится ',l,' символов');**

**k:=0; b:=''; c:='';**

**for i:=1 to l do begin b:=copy(a,i,1); if b='a' then k:=k+1 end;**

**writeln('В предложении содержится ',k,' символов "a");**

**for i:=1 to l do**

**begin**

**b:=copy(a,i,1);**

**if b<>',' then c:=c+b else c:=c+' '**

**end;**

**delay(6000);** { временная задержка вывода (пауза) }

**clrscr; write(c);**

**b:=copy(c,1,2);**

**for i:=2 to 22 do**

**begin**

**gotoxy(1,i); write(b); delay(500);** { печатьдвухсимволов }

**gotoxy(1,i); write(' ')** { стирание этих символов }

**end; readkey;**

**End.**

**Практическая часть**

*Задачи для самостоятельной работы*

1. Определить число символов в предложении: 22 сентября - день осеннего равноденствия. Вырезать слово 'день' и присвоить его переменной b. Вывести предложение и это слово на экран.
2. Ввести свою загадку и ответ. Предусмотреть различные способы ввода отгадки.

"Спрятать" в программе отгадку и вызывать ее с помощью строковых процедур и функций.

1. Ввести слово произвольной длины и напечатать его в виде треугольника:

1)Л 2) ЛИПА 3) ЛИПА 4) ЛИПКА

ЛИ ЛИП ИПА ИПК

ЛИП ЛИ ПА П

ЛИПА Л А

1. Запишите условие истинное, если:в записи фамилии «Соколенко» встречается буква «л»;
2. Запишите условие истинное, если:в записи фамилии «Соколенко» не встречается буква «л»;
3. Запишите условие истинное, если:в записи фамилии «Соколенко» встречается буква «я»;
4. Запишите условие истинное, если:в записи фамилии «Соколенко» не встречается буква «я»;
5. Запишите условие истинное, если:  в записи фамилии «Соколенко» встречается буква «н», но не встречается буква «у»;
6. Запишите условие истинное, если: из букв фамилии «Соколенко» можно образовать слово «лоск»;
7. Запишите условие истинное, если:из букв фамилии «Соколенко» можно образовать слово «локон»;
8. Запишите условие истинное, если:из букв фамилии «Соколенко» можно образовать слово «кокон»;
9. Запишите условие истинное, если: из букв фамилии «Соколенко» нельзя образовать слово «коляска»;
10. Проверьте, является ли заданный символ строчной гласной буквой русского алфавита.
11. Проверьте, является ли заданный символ согласной буквой русского алфавита (заглавной или строчной).
12. Составить программу, которая выводит на экран слово, удваивая каждую букву исходного слова.
13. Определить на какую букву заканчивается третье слово в строке.
14. Написать программу, выдающие номер позиции заданной строки, в которой впервые встречается буква "а" и сообщение "end", если в строке нет такой буквы.
15. В предложении посчитать количество букв «а» и количество букв «б».
16. Написать программу, которая подсчитывает, сколько раз среди символов заданной строки встречается буква "К".
17. Дано предложение. Вычислить процент вхождения буквы «о» в данное предложение и получить новое предложение путем увеличения данного, на полученный процент, пробелами.
18. Дана фраза определить сколько в ней предложений.
19. Написать программу, выдающую номер позиции заданной строки, в которой в пятый раз встречается буква "к" и соответствующее сообщение если эта буква встречается менее пяти раз.
20. Дано слово. Сколько букв 'а' в этом слове стоит на четных местах.
21. Дано предложение. Определить, каких букв в нем больше: «м» или «н».
22. Дано предложение напечатать все его символы, предшествующие первой встретившейся запятой.

**Контрольные вопросы:**

1. Как записываются строковые константы?
2. Как описываются строковые переменные?
3. Перечислите операции и функции, разрешенные для данных строкового типа.
4. Запишите условие истинное, если:
5. Какое значение вернет процедура val(2e5,i,k)?
6. Укажите, какие из указанных ниже операций допустимы в Паскале:

var a1,a2:string; b1,b2:string[80]; c:string[5];

* + - 1. a1=a2;
      2. b2>b1;
      3. a1<=b2;
      4. c=b1;
      5. c:=a2;
      6. a2:='b1+b2';
      7. a1:=a2;
      8. a2:=b2;

# **Практическое занятие №12. Использование стандартных функций и процедур для работы со строками**

# **Цель работы: нучиться использовать стандартные процедуры и функции дря работы со строковыми переменными**

# **Теоретическая часть**

## Стандартные функции для работы со строками

**Функции:**

функция соединения: **CONCAT(s1,s2,...,sn)** - складывает строки s1,s2,...,sn (результат не должен превышать 255)

Например: Sum:=Concat(s1,s2,s3).;

функция выделения: **COPY(S,T,K)** где S - строка, T – начало позиции, K – длина подстроки;

st:='Привет!';   s:=copy(st,4,3); {Результатом будет следующая подстрока} s='вет';   
st1:='Привет!';  s1:=copy(st,4,10); {Результатом будет следующая подстрока} s1='вет!';

функция определения длины строки: **LENGTH(S)** - результатом является число;

st:='Привет!';          k:=length(st);            {Длина результирующей строки равна} k=7;   
st1:='Всем привет!';    k1:=length(st1);   {Длина результирующей строки равна}  k1=12;

функция определения позиции: **POS(T,S),** где T - элемент, позицию которого надо определить, S - строка, в которой будет определение;

st1:='abcdef';

pos('de',st1)=4;

pos('r',st1)=0;

**Процедуры:**

процедура вырезания: **DELETE(S,T,K)**, где S - какая строка, T - с какого элемента, K - сколько элементов вырезать;

st:='abcdef';

delete(st,2,3);

{Результатом будет следующая строка} st='aef';

процедура вставки: **INSERT(T,S,K),** где T - что вставлять, S - в какую строку, K - с какой позиции;

st1:='12389';

st2:='4567';

insert(st2,st1,4);

{Результатом будет следующая строка} st1='123456789';

процедура преобразования числа в строку: **STR(T,S)**, где T - число, которое будет преобразовано, S - строка, в которую будет преобразовано число;

str(1500,st)='1500';

str(1500:6,st)=' 1500';

**VAL(S,K,T)** - преобразует строку S в числовое значение, если это возможно. Параметр T содержит ноль, если преобразование прошло успешно, и тогда в Х помещается результат преобразования, в противном случае он содержит номер позиции в строке S, где обнаружен ошибочный символ. Например:

Val('125',K,kod) - в результате выполнения этой процедуры переменная К получает целое значение, равное 125, параметр kod=0;

Val(' 1.05',M,code) - M=1.05, code=0;

Val('100, ',N,code) - это ошибочный вызов, т.к. в исходной строке на 4-й позиции располагается недопустимый для числа символ ',' и поэтому параметр code=4, а переменная N остается без изменения.

**Символьный тип(char)**

K типу char относится любой символ заключенный в апострофы. Для представления апострофа как символьную переменную, надо заключить его в апостроф.

Каждый символ имеет свой код и номер. Порядковые номера цифр 0,1..9 упорядочены по возрастанию. Порядковые номера букв также упорядочены по возрастанию, но не обязательно следуют друг за другом.

К символьным данным применимы знаки сравнения: > , < , >=, <=, <> .

Например: 'A' < 'W'

**Функции**, которые применимы к символьным переменным:

ORD(X) - определяет порядковый номер символа Х.

Пример: ord('a')=97.

CHR(X) - определяет символ по номеру.

Пример: chr(97)='a'.

PRED(X) - выдает символ, стоящий перед символом Х

Пример: pred('B')='A'

SUCC(X) - выдает символ, следующий после символа Х

Пример: succ('A')='B'.

UPCASE(ch) - преобразует строчную букву в прописную. Обрабатывает только буквы латинского алфавита.

**Примеры задач**

1. Посчитать количество слов в предложении.

program zadacha\_2;

uses crt;

var

st:string;

k,i,n:integer;

begin

clrscr;

writeln ('Введите предложение');

readln(st);

n:=length(st);

for i:=1 to n do

begin

if (st[i]=' ') or (st[i]=', ') or (st[i]='. ') then

k:=k+1;

end;

writeln ('Количество слов в строке',k);

readln;

end.

1. Даны два слова. Из первого слова удалить все буквы, которые встречаются во втором слове.

program zadacha\_3;

uses crt;

var

i,j,l,m:integer;

sl,sl1:string;

begin

clrscr;

writeln('введитеслово');

readln(sl);

writeln('введите другое слово ');

readln(sl1);

l:=length(sl);

m:=length(sl1);

for i:=1 to l do

for j:=1 to m do

if sl[i]=sl1[j] then

begin

delete(sl,i,1);

end;

writeln(sl);

readln;

end.

1. Определить наименьшую длину слова в заданной строке.

program zadacha\_4;

uses crt;

var

a: array [1..50] of integer;

s:string;

min,dl,i,l,k:integer;

begin

clrscr;

writeln('введите предложение');

readln(s);

l:=length(s); {строковаяфункция}

min:=l; {минимальному слову присваиваем длину}

dl:=0; {длинатекущегослова}

for i:=1 to l do {цикл}

begin

if s[i]<>' ' then

dl:=dl+1;

else

begin

a[k]:=dl;

dl:=0;

k:=k+1;

end;

begin

if (min>dl) and (dl>0) then

begin

min:=dl;

end;

end;

end;

writeln( 'длина наименьшего слова: ', min);

readln;

end.

1. Написать программу, определяющую, сколько слов заданной строки оканчивается на последнюю букву первого слова этой строки.

uses crt;

var

st,w:string [30];

i,n,k,t:integer;

begin

clrscr;

writeln('Введите предложение');

readln(st);

begin

n:=length(st);

end;

begin

k:=pos(' ',st);

w:=st[k-1];

end;

for i:=k+1 to n do

if (st[i]=' ') and (st[i-1]=w) then

begin

t:=t+1;

end;

if st[n]=w then

begin

t:=t+1;

end;

writeln(t);

readln;

end.

1. Из предложения удалить все символы, стоящие на местах, кратных трем.

program zadacha\_7;

uses crt;

var

sl:string[255];

k,l,i,m:integer;

begin

clrscr;

write(' Введите предложение ');

readln(sl);

l:=length(sl);

k:=1;

for i:=1 to l do

if sl[i]=' ' then

k:=k+1;

write(' Введите № слова' ,k);

readln(k);

for i:=1 to l do

begin

if i mod 3=1 then

delete(sl,k+i,1);

end;

writeln(' sl=',sl,' ',m);

readln

end.

1. Посчитать, сколько слов в предложении оканчиваются буквой "я".

program zadacha\_9;

uses crt;

var

st: string[80]; {строка-предложение}

k: integer; {кол-во слов, оканчивающихся на «я»}

l: integer; {длина строки}

i: integer; {переменная - счетчик}

begin

clrscr;

write ('введите предложение ');

Readln (st);

l:=length (st);

for i:=1 to l do

begin

if (copy(st,i,2)='я.') or (copy(st,i,2)='я, ') or (copy(st,i,2)='я ') then

begin

k:=k+1;

end;

end;

writeln('символов в строке = ',l,' ');

writeln('строка - ',st,' ');

writeln('слов, которые оканчиваются на "я" = ',k,' ');

readln;

end.

1. Написать прогрму которая из даннго слова получает новое путем удалениея "\*".

program zadacha\_10;

uses crt;

var

sl:string[255];

n,i:integer;

begin

clrscr;

write(' Введите строку ');

readln(sl);

n:=length(sl);

for i:=1 to n do

if sl[i]='\*' then

begin

delete(sl,i,1);

end;

for i:=1 to 3 do

begin

write(sl);

readln;

end

end.

1. Опредилить на каких местах стоит буква "р" и "п".

program zadacha\_11;

uses crt;

var

sl:string[255];

n,i:integer;

begin

clrscr;

write(' Введите строку ');

readln(sl);

n:=length(sl);

for i:=1 to n do

begin

if sl[i]='п' then

writeln(' Место на каком стоит буква "п" ',i);

if sl[i]='р' then

writeln(' Место на каком стоит буква "р" ',i);

end;

readln;

end.

1. Посчитать количество слогов слова.

program zadacha\_13;

uses crt;

var

sl,sl1:string[255];

k,i,n,j:integer;

begin

clrscr;

write(' Введите предложение ');

readln(sl);

k:=0;

n:=length(sl);

sl1:='ауыеоэюияё';

for i:=1 to n do

for j:=1 to 10 do

begin

if (sl[i]=sl1[j]) then

k:=k+1;

end;

writeln(' Слогов ', k);

readln;

end.

1. В строке все пробелы, стоящие на четных местах, заменить на двойные пробелы.

program zadacha\_14;

uses crt;

var

sl,h:string[100];

l,i,j,k:integer;

begin

clrscr;

writeln (‘Введитестроку');

readln (sl);

l:=length(sl);

j:=1;

for i:=1 to l do

if (i mod 2 <> 0) or (sl[i]<>' ') { проверяет условие если I не стоит на

then четных местах и не равно пробелу }

begin

h[j]:=sl[i]; {то новой строке присваиваем старую}

j:=j+1;

end

else

begin

h[j]:=sl[i]; {иначе новая строка равна старой плюс два пробела}

j:=j+2;

h[j+1]:=sl[i];

end;

for i:=1 to j-1 do

begin

write (h[i]);

end;

readln;

end.

1. Из строки символов удалить группу символов «абвг».

program zadacha\_15;

uses crt;

Var

st,st1:string [100];

t,i,r:integer;

begin

clrscr;

writeln('Введитестроку');

readln(st);

t:=length(st);

r:=0;

for i:=1 to t do

if copy(st,i,4)<>'абвг' then

begin

r:=r+1;

st1[r]:=st[i];

end

else

i:=i+3;

for i:=1 to r do

write(st1[i]);

readln;

end.

1. Дано слово из 12 букв. Получить новое слово, разместив первую треть слова на месте третей, вторую – на месте первой, третью – на месте второй.

program zadacha\_16;

uses crt;

var

s: string[12];

s1,s2,s3: string[4];

l,i:integer;

begin

clrscr;

writeln ('Введите слово из 12 букв.');

readln (s);

l:=length(s);

i:=1;

s1:=copy(s,i,4);

s2:=copy(s,i+4,4);

s3:=copy(s,i+8,4);

s:=s2+s3+s1;

writeln ('Новоеслово');

write (s);

readln;

end.

end.

1. Выяснить сколько раз встречается каждая буква алфавита в строке.

program zadacha\_18;

uses crt;

var

st: string [100];

procedure alphavit (st: string);

var

s: string;

k, i,l,j,n: integer;

begin

s: ='абвгдеёжзиклмнопрстуфхцчшщьыъэюя';

l: =length(s);

n: =length (st);

for i: =1 to l do

begin

k: =0;

for j: =1 to n do

begin

if s[i] = st[j] then

begin

k:=k+1;

end;

end;

if k>0 then

begin

writeln('Буква ',s[i],' встречается в предложении ',k,'раз');

readln;

end;

end;

end;

begin

clrscr;

writeln ('Введитепредложение');

readln (st);

alphavit (st);

readln;

end.

1. Из слова «спаниель» путем вырезок и склеек его букв получить слово «апельсин».

program zadacha\_20;

uses crt;

const

s='спаниель';

var

s1: string[8];

s1,s2,s3: string[4];

l,i:integer;

begin

clrscr;

l:=length(s);

i:=1;

s1:=s[3]+s[2]+copy(s,6,3)+s[1+s[5]+s[4];

writeln ('Из слова спаниель получено слово - ',s1);

readln;

end.

1. Дано предложение. Верно ли ,что число слов в предложение больше трех?

program zadacha\_22;

uses crt;

var

st: string;

i,k,l: integer;

begin

clrscr;

writeln ('Введите предложение ');

readln (st);

l:=length(st);

for i:=1 to l do

begin

if (st[i]=' ') or (st[i]=',') or (st[i]='.') then

begin

k:=k+1;

end;

end;

if k>3 then

begin

writeln ('В введённом предложении больше трёх слов');

end

else

begin

writeln ('В введённом предложении меньше трёх слов');

end;

readln;

end.

1. Дан текст. Верно ли, что в нем есть, пять идущих подряд одинаковых чисел.

program zadacha\_23;

uses crt;

var

st:string; {текст}

i:integer; {номер буквы слова}

k:integer; {количество букв в тексте}

c:integer; {количество в тексте пяти идущих подряд одинаковых чисел}

begin

clrscr;

writeln('Введитестроку');

readln(st);

k:=length(st); {определяемдлину}

c:=0;

for i:=1 to k do

if (st[i]=st[i+1]) and (st[i]=st[i+2]) and (st[i]=st[i+3]) and

(st[i]=st[i+4])then

{определяем есть ли пять идущих подряд одинаковых символов}

begin

c:=c+1; {если есть то число увеличиваем на 1}

end;

if c>=1 then

begin

writeln('Верно');

end

else

writeln('Нет');

readln;

end.

1. Дан текст. Определить количество букв "и" в первом предложение.

program zadacha\_24;

uses crt;

var

st: string;

i,k,l: integer;

begin

clrscr;

writeln ('Введитетекст ');

readln (st);

l:=length (st);

i:=1;

while (st[i] <> '.') and (i < l) do

begin

if (st[i]='и') or (st[i]='И') then

begin

k:=k+1;

end;

i:=i+1;

end;

writeln ('Количество букв и в первом предложение ',k);

readln;

end.

1. Дано предложение. Определить есть ли в нем сочетание "чу" или "щу".В случае положительного ответа найти также порядковый номер первой буквы первого из них.

program zadacha\_25;

uses crt;

var

st:string; {описываюданные}

j,n,i,l:integer;

begin

clrscr;

writeln('введитестроку');

readln(st);

l:=length(st); {определяюдлинустроки}

for i:=1 to l do

begin

if((st[i]='щ')and(st[i+1]='у'))or((st[i]='ч')and(st[i+1]='у'))then

begin

writeln('в предложении есть сочетание "щу" или "чу"');

readln;

writeln(i); {определяю номер 1 буквы одного из этих сочетаний}

readln;

end;

end;

end.

1. Дано предложение напечатать все его символы, расположенные между двумя первыми встретившимися запятыми.

program zadacha\_26;

uses crt;

var

st:string [100];

procedure asd(st1:string);

var

sl,sl1:string [100]; {предложение}

i,n:integer; {переменные счетчика}

begin

n:=length(st1); {вычисление текущей длины строки}

for i:=1 to n do

If st1 [i]=',' then

sl:=copy(st1,i+1,n); {вырез из строки подстроку}

for i:=1 to n do

If st1 [i]=',' then

delete (st1,i,n); {удаление}

sl1:=st1+sl;

writeln (sl1);

readln;

end;

begin

clrscr;

writeln('введите предложение');

readln (st);

asd(st);

end.

1. Дано слово, состоящее из четного кол-ва букв. Поменять местами первую букву со второй, третью с четвертой и т.д.

program zadacha\_27;

uses crt;

var

s:string[16]; {слово}

c:char; {переменная для замены букв}

l:integer; {длина слова}

i:integer; {переменная-счетчик}

begin

clrscr;

writeln('введите слово, состоящее из четного кол-ва букв');

readln(s);

l:=length(s); {находимдлинустроки}

i:=i+1;

for i:=1 to l do

begin

if l mod 2=0 then {если в слове четное кол-во букв, то}

begin

c:=s[i]; {присваиваем переменной первую, третью и т.д.букву}

s[i]:=s[i+1]; {первой, третьей и т.д. букве присваиваем следующ.}

s[i+1]:=c; {след. присв. первую, третью и т.д.

i:=i+1; {увеличиваем счетчик

end;

end;

writeln('s=',s); {выводим новое слово}

readln;

if l mod 2>0 then {если в слове нечетное кол-во букв, то выводим}

begin

writeln('в слове нечетное кол-во букв');

readln;

end;

end.

# **Практическая часть**

*Задачи для самостоятельного решения:*

1. Дано предложение, в котором есть несколько букв «е». Найти порядковый номер первой и последней из них.
2. Дано предложение. Определить есть ли в нем буква «а». Есть таких букв несколько, то определить порядковый номер первой их них.
3. Дана последовательность символов, в начале которой имеется некоторое количество одинаковых символов. Определить это количество.
4. Даны два слова. Определить, сколько начальных букв первого слова совпадают с начальными буквами второго слова.
5. Да последовательность слов. Проверить правильность написания сочетания «ча» и «ща». Исправить ошибки.
6. Дано предложение, в котором имеются буквы «с» и «т». Определить, какая из них встречается позже (при просмотре слова слева направо). Если таких букв несколько, то должны учитываться последние из них.
7. Отредактировать заданное предложение, удаляя из него все слова с нечетными номерами и переворачивая слова с четными номерами.
8. Дана строка, в которой имеется текст в скобках. Написать программу удаления части текста, заключенного в скобки, вместе со скобками. Оставшуюся часть строки отсортировать в алфавитном порядке.
9. Написать программу, которая определяет количество слов заданной строки, длина каждого из которых равна длине первого слова этой строки. Строку отсортировать в алфавитном порядке.
10. Написать программу нахождения самого короткого слова заданной строки.
11. Написать программу подсчета в заданной строке количества букв латинского алфавита.
12. Написать программу подсчета количества букв «д» в последнем слове заданной строки.
13. Дано предложение. Переписать предложение, переставляя слова в нем в обратном порядке.
14. Написать программу нахождения в заданной строке первого слова, которое имеет заданную длину.
15. Проверить, являются ли заданные слова в предложении палиндромами (перевёртышами).
16. Написать программу получения из заданной строки новой строки путем удаления всех символов, не являющихся буквами латинского алфавита
17. Написать программу, которая выводит сообщение «ДЛИНА», если все слова в тексте имеют одинаковую длину.
18. Написать программу, определяющую, сколько слов в заданной строке оканчивается буквой «я», если известно, что слова обязательно стоит либо пробел, либо точка.
19. Написать программу определения слов заданной строки, которые оканчиваются на «ова».
20. Написать программу подсчета количества слов, имеющих один слог
21. Составьте программу, вычеркивающую. Каждую третью букву из заданного слова.
22. Из слова «вертикаль» путем вырезок и склеек его букв получить слова «тир» и «ветка».
23. Дана квадратная матрица символьного типа. Транспонировать ее.
24. Дано слово. Получить и вывести на экран буквосочетание, состоящее из третьего и последнего символов.
25. Из слова «программа» путем вырезок и склеек его букв получить слова «ром» и «рампа».

*Задачи для дополнительного решения*

1. Дано предложение. Все пробелы в нем заменить на символ «\_».
2. Дана квадратная матрица символьного типа. Транспонировать ее. Вывести на экран элементы, расположенные выше главной диагонали.
3. Дано слово. Получить и вывести на экран буквосочетание, состоящее из второго, третьего и четвертого символов.
4. Из слова «трос» путем вырезок и склеек его букв получить слова «сорт», «рост» и «торс»
5. Дано слово, состоящее из четного числа букв. Получить и вывести на экран его первую половину.
6. Дано предложение. Сколько раз и на каких позициях встречается буквосочетание «нн».
7. Из слова «клоун» путем вырезок и склеек его букв получить слова «уклон» и «кулон».
8. Дана квадратная матрица символьного типа. Транспонировать ее. Вывести на экран элементы, расположенные на побочной диагонали.
9. Дано слово. Получить и вывести на экран его часть, образованную идущими подряд буквами, начиная с m – ой n - ой.
10. Из слова «клоун» путем вырезок и склеек его букв получить слова «колун» и «кулон».
11. Из слова 'понедельник', используя функции строковых данных, получите слово 'плодик'.
12. Дано слово. Перенести три его первые буквы в конец.
13. Дано предложение. Заменить в нем все вхождения буквосочетания «ах» на «ух».
14. Дано слово. Поменять местами его половины.
15. Определить длину последнего слова в строке.
16. Дано предложение. Добавить к нему в конце столько звездочек, сколько букв в этом слове.
17. Из слова «апельсин» путем вырезок и склеек его букв получить слово «спаниель».
18. Дано предложение. Заменить в нем все вхождения буквосочетания «да» на «не».
19. Дана строка, содержащая текст на русском языке. Выяснить, входит ли данное слово в указанный текст, и если да, то сколько раз.
20. В предложении некоторые слова записаны подряд несколько раз. Отредактировать этот текст, удаляя из него подряд идущие одинаковые слова.
21. Преобразуйте прописные буквы русского алфавита в заглавные.
22. Проверьте, является ли заданный символ цифрой.
23. Дан текст. Вывести на экран слова, состоящие из трех букв.
24. Дана строка, содержащая произвольный текст. Выяснить чего в нем больше: русских букв или цифр.
25. Дана строка, содержащая текст на русском языке и некоторая буква. Найти слово, содержащее наибольшее количество указанных букв.

# **Контрольные вопросы:**

1. Укажите, какие из указанных ниже выражений допустимы в Паскале:

var a1,a2:string; b1,b2:string[80]; c:string[5];

* + - 1. copy(a1,5,3)+length(a2);.
      2. length(b2)+pos('aa',b1);
      3. pos(c,a1)+length(copy(c,6,2));
      4. b1+copy(b2,65,20)+copy(pos('a',c),1,7));
      5. pos(copy(a1,4,6),delete(a2,2,1))+length(b2);
      6. insert(copy(c,2,3),a2+b2,3);
      7. delete(copy(a2,5,6),3,4));
      8. insert(copy(a1,5,5),a2,3);

1. Установите значения следующих выражений:
   * + 1. concat('кро',a,b) при a='код', b='ил';
       2. copy(s,2,3) при s='крокодил';
       3. copy(s,4,2)+s[2]+s[7]+s[6]+s[5]+s[2] при s='крокодил';
       4. pos('око',s) при s='крокодил';
       5. pos('я','крокодил');
       6. length('зеленый'+' '+s) при s='крокодил';
       7. insert('чо',w,5) при w='моряк';
       8. insert('Н',d,3) при d='око';
       9. delete(w,5,2) при w='морячок';
       10. str(123,s);
       11. pos('а','аяоёуюыиэе')>0;
       12. copy(s,pos(' ',s)+1,length(s)-pos(' ',s)) при s='ВасяКук – информатик';
       13. delete(s,length(s),1) при s='123456789';
       14. copy(s,i,j-i+1) при s='123456789', i=3, j=7;
       15. insert(b1,b2,length(b1)) при b1='1', b2='22';

3. Установите, какую задачу решает приведенный ниже фрагмент:

1) s:='';

for i:=1 to length(t) do

if pos(copy(t,i,1),'aeiouy')>0

then s:=s+copy(t,i,1);

2) s:='';

for i:=length(t) downto 1 do

s:=copy(t,i,1)+s.

# **Практическое занятие №13 Работа с данными типа множество.**

**Цель работы: научиться решать задачи неупорядоченной совокупности неповторяющихся элементов одного типа, имеющих общее имя.**

**Теоретическая часть**

## Представление множества в языке TurboPascal

***Множество***- неупорядоченная совокупность неповторяющихся элементов одного типа, имеющих общее имя.

Количество элементов, входящих в множество, может меняться в пределах от 0 до 255. Множество, не содержащее элементов, называется *пустым*. Множество имеет имя. Тип элементов, входящих в множество, называется базовым. В качестве базового типа можно использовать любой порядковый тип, кроме Word, Integer, Longint.

Множества должны быть объявлены либо в разделе Var, либо в разделах Type и Var, одновременно:

**Var Имя множества:Setof базовый тип;**

**или**

**Type Имя типа=Setof базовый тип;**

**Var Имя множества:Имя типа;**

Например:

Type

TM=Set of 1..100;

TS=Set of 'a'..'z';

VarMch:TM; {Множество целых чисел от 1 до 100}

MSym:TS; {Множество строчных латинских букв}

M: Setof 1..10; {Множество целых чисел от 1 до 10}

Значения переменных множества задаются в разделе операторов с помощью ***конструктора***множества, который представляет собой список элементов базового типа, заключенный в квадратные скобки.

Например:

Var M1,M2,M3:set of 1..99;

Begin . . .

M1:=[]; { Множество пустое}

M2:=[1,3,5,7,9]; { Множество нечетных чисел в первом десятке}

M3:=[2,4,6,8]; { Множество четных чисел в первом десятке}

. . .

End.

В качестве элементов в изображении множеств допускается использовать константы и выражения, тип которых совместим с базовым типом.

Типизированная константа - множество задается в виде правильного конструктора множества, например:

**Пример 1.**

Type

Type\_month=(Jn,Feb,Mar,Apr,May,Jun,Jul,Aug,Sep,Oct,Nov,Dec);

TDays=Set of 1..31;

Tmonth=Set of 1..12;

Tsym=Set of 'A'..'Z';

Tmno=Set of Type\_month;

Const

SymMno:Tsym=['A','E','I','O','U']; {подмножество гласных букв}

DaysMno:TDays=[1,8,15,22,29]; {подмножество выходных дней месяца}

Spring\_Mes:Tmonth=[3,4,5]; {подмножество весенних месяцев года}

Spring\_Month:Tmno=[Mar,Apr,May]; {то же, что и предыдущее}

**Пример 2.**  
    Type   
        digitChar = Set Of ‘0’ .. ‘9’;   
        digit = Set Of 0 .. 9;   
    Var   
        s1, s2, s3 : digitChar;   
        s4, s5, s6 : digit;   
        . . . . . .   
        s1 := [‘1’, ‘2’, ‘3’];   
        s2 := [‘3’, ‘2’, ‘1’];   
        s3 := [‘2’, ‘3’];   
        s4 := [0. .3, 6];   
        s5 := [4, 5];   
        s6 := [3. .9];. . . . . .

## Операции над множествами

Над множествами определены следующие операции:

|  |  |  |  |
| --- | --- | --- | --- |
| Операция | Смысл | Пример\* | Результат |
| **\*** | **пересечение множеств** | **s1\*s3** | **[‘2’, ‘3’]** |
| **+** | **объединение множеств** | **s4+s5** | **[0. .6]** |
| **-** | **разность множеств** | **s2-s3** | **[‘1’]** |
| **=** | **True, если множества**  **эквивалентны** | **s1=s2**  **s1=s3** | **True**  **False** |
| **<>** | **True, если множества неэквивалентны** | **s1<>s2**  **s1<>s3** | **False**  **True** |
| **<=** | **True, если первое множество содержится во втором** | **s5<=s6** | **True** |
| **>=** | **True, если второе множество содержится в первом** | **s2>=s3** | **True** |
| **In** | **проверка принадлежности значения выражения, стоящего слева множеству, стоящему справа** | **7-5 In s4** | **True** |

\* - примеры приведены согласно описаниям и конструкторам множеств в примере 2.

Дополнительно к этим операциям можно использовать две процедуры:

**Include**- включает новый элемент во множество: Include(M,elem);

где М - множество элементов некоторого базового типа, а elem - элемент того же типа, который необходимо включить в множество М.

**Exclude -** исключает элемент из множества: Exclude(M,elem).

В отличие от операций "+" и "-", реализующих аналогичные действия над двумя множествами, эти процедуры оптимизированы для работыс одиночными элементами множества и поэтому отличаются высокой скоростью выполнения.

Основным достоинством использования множеств является экономия памяти: внутренне устройство множества таково, что каждому его элементу ставится в соответствие один двоичный разряд (один бит). Если элемент включен в множество, то соотвествующий разряд имеет значение 1, в противном случае - 0. Минимальной единицей памяти является 1 байт, содержащий 8 бит, поэтому для хранения множества мощностью 256 элементов выделяется память 32 смежных байта.

Рассмотрим работу с множествами на следующем примере.

Из множества целых чисел от 1 до 20 выделить:

1. множество чисел, делящихся на 2 и 3 одновременно;
2. множество чисел, делящихся на 2 или на 3.

Первая задача соответствует нахождению пересечения множеств чисел, одно из которых содержит числа, делящиеся на 2, а другое на 3. Вторая - объединению этих двух множеств.

Обозначим множество чисел, делящихся на 2 через М2; множество чисел, делящихся на 3 через М3; множество чисел, делящихся на 2 и 3 через М2and3; множество чисел, делящихся на 2 или 3 через М2or3.

**Пример 3.**

Type TM=Set of 1..20; {Описание типа множества целых чисел от 1 до 20}

Var M2,M3,M2and3,M2or3:TM; {Описание множеств}

k:1..20; {Описание переменной}

Begin

M2:=[]; M3:=[]; {Пустые множества}

for k:=1 to 20 do

begin

if k mod 2 = 0 then Include(M2,k); {Включениеэлементаделящегосяна 2 в

множество М2}

if k mod 3 = 0 then Include(M3,k); { Включение элемента делящегося на 3 в

множество М3}

end;

M2and3:=M2\*M3; {Пересечение двух множеств}

M2or3:=M2+M3; {Объединение двух множеств}

write(' На 2 и 3 делятся числа: ');

for k:=1 to 20 do { Цикл для опеределения элементов в множестве}

if k in M2and3 then write(k:3); { вывод элементов делящихся на 6}

writeln; {Переход на новую строку экрана}

write(' На 2 или 3 делятся числа: ');

for k:=1 to 20 do

if k in M2or3 then write(k:3); readln; {Остановка для просмотра}

End.

**Практическая часть**

***Задачи для самостоятельной работы***

1. Дан текст из строчных латинских букв, за которым следует точка. Напечатать первые вхождения букв в текст, сохраняя их взаимный исходный   порядок.

2. Дан текст из строчных латинских букв, за которым следует точка. Напечатать все буквы, входящие в текст не менее двух раз.

3. Дан текст из строчных латинских букв, за которым следует точка. Напечатать все буквы, входящие в текст по одному разу.

4. Дан текст, за которым следует точка. В алфавитном порядке напечатать все строчные русские гласные буквы (а, е, и, о, у, ы, э, ю, я), входящие в этот текст.

5. Дана непустая последовательность слов из строчных русских букв; между соседними словами - запятая, за последним словом - точка. Напечатать в алфавитном порядке все гласные буквы, которые входят в каждое слово.

6. Дана непустая последовательность слов из строчных русских букв; между соседними словами - запятая, за последним словом - точка. Напечатать в алфавитном порядке все согласные буквы, которые не входят ни в одно слово.

7. Дана непустая последовательность слов из строчных русских букв; между соседними словами - запятая, за последним словом - точка. Напечатать в алфавитном порядке все звонкие согласные буквы, которые входят хотя бы в одно слово.

8. Дана непустая последовательность слов из строчных русских букв; между соседними словами - запятая, за последним словом - точка. Напечатать в алфавитном порядке все глухие согласные буквы, которые не входят хотя бы в одно слово.

9. Дана непустая последовательность слов из строчных русских букв; между соседними словами - запятая, за последним словом - точка. Напечатать в алфавитном порядке все согласные буквы, которые входят только в одно слово.

10. Дана непустая последовательность слов из строчных русских букв; между соседними словами - запятая, за последним словом - точка. Напечатать в алфавитном порядке все глухие согласные буквы, которые не входят только в одно слово.

11. Дана непустая последовательность слов из строчных русских букв; между соседними словами - запятая, за последним словом - точка. Напечатать в алфавитном порядке все звонкие согласные буквы, которые входят более чем в одно слово.

12. Дана непустая последовательность слов из строчных русских букв; между соседними словами - запятая, за последним словом - точка. Напечатать в алфавитном порядке все гласные буквы, которые не входят более чем в одно слово.

13. Дана непустая последовательность слов из строчных русских букв; между соседними словами - запятая, за последним словом - точка. Напечатать в алфавитном порядке все звонкие согласные буквы, которые входят в каждое нечетное    слово и не входят ни в одно четное слово.

14. Дана непустая последовательность слов из строчных русских букв; между соседними словами - запятая, за последним словом - точка. Напечатать в алфавитном порядке все глухие согласные буквы, которые входят в каждое нечетное слово и не входят хотя бы в одно четное слово. (гласные буквы - а, е, и, о, у, ы, э, ю, я; звонкие согласные буквы - б, в, г, д, ж, з, л, м, н, р; глухие согласные буквы - к, п, с, т, ф, х, ц, ч, ш, щ.)

**Дополнительно:**

6. Дан текст, являющийся русскими словами. Выполнить разделение его на части для переноса.

Далее правила переноса русских слов.   
a) Две идущие подряд гласные можно разделить, если первой из них предшествует согласная, а за второй идет хотя бы одна буква (буква й при этом рассматривается вместе с предшествующей гласной как единое целое).   
b) Две идущие подряд согласные можно разделить, если первой из них предшествует гласная, а в той части слова, которая идет за второй согласной, имеется хотя бы одна гласная (буквы ъ, ь вместе с предшествующей согласной рассматриваются как единое целое).   
c) Если не удается применить пункты a, b, то следует попытаться разбить слова так, чтобы первая часть содержала более чем одну букву и оканчивалась на гласную, а вторая содержала хотя бы одну гласную.

Вероятность правильного разбиения увеличивается, если предварительно воспользоваться хотя бы неполным списком приставок, содержащих гласные, и попытаться прежде всего выделить из слова такую приставку.

**Контрольные вопросы.**

* + - 1. Что такое множество? Каким требованиям должны удовлетворять все элементы множества? Преимущества использования типа множество?
      2. Какое множество называется пустым? Как оно обозначается?
      3. Как задается описание множественного типа?
      4. Какие операции допустимы над множествами? Каков тип результатов выражений с применением операций над множествами?
      5. Какие множества считаются равными, неравными? Имеет ли значение для сравниваемых множеств порядок следования элементов?
      6. Для чего применяютя операции «больше или равно», «меньше или равно»? В чем их отличие?
      7. Для чего применяется операция in? Особенности ее применения.
      8. Что называетсся объединением множеств?
      9. Что называется пересечением множеств?
      10. Что называется разностью множеств?
      11. Какие из следующих конструкций являются множествами (в смысле языка Паскаль), а какие нет и почему?   
          а) [9,6,3,0]; б) [2...3,5,7];   
          в) [1...15,4...18]; г) [ '\*', '\*'];   
          д) [0...0]; е) [true...false];   
          ж) [sgrt(9)]; з) [ '=', '>='] .

# **Практическое занятие №14 – 15 Организация процедур. Использование процедур.**

# **Цель работы: нуачиться решать задачи с использованием процедур.**

# **Теоретическая чась**

# *П Р О Ц Е Д У Р Ы*

Программу, состоящую только из операторов, называют МОНОЛИТНОЙ.

Программу, содержащую процедуры и функции называют **МОДУЛЬНОЙ**.

**ПРОЦЕДУРА** - часть программы, имеющая имя и предназначенная для решения определенной задачи. Каждая процедура определяется только однажды, но может использоваться многократно. Процедуры определяются в разделе описаний вслед за разделом переменных.

**PROCEDURE Name;** prоcedure - служебноеслово;

**BEGIN** Name - идентификатор отражающий смысл данной задачи;

**ИНСТРУКЦИЯ** Описание процедуры само по себе никакого действия не

**END;** вызывает. Чтобы исполнить процедуру, необходимо в нужном

месте программы поместить обращение к ней, указав Name; .

Например: Напечатать сорок любых символов.

В программе используется функция **WhereX** – она определяет позицию курсора в строке (принимает значения от 1 до 80, изменяется автоматически при выводе информации на экран). С помощью этой функции можно проверить, где находится курсор. Аналогичная функция **WhereY**– определяет строку, в которой в данный момент находится курсор (принимает значения от 1 до 25). Условие (wherex<40) означает, что при выводе на экран еще не достигнута 40 позиция в текущей строке.

**Program Stroka; Uses crt;** { Stroka - имяпрограммы }

**procedure Str\_Simw;** { Str\_Simw - имяпроцедуры }

**begin**

**while wherex<40 do write('$');** { - телопроцедуры }

**writeln**

**end;**

**Begin**

**clrscr;**

**Str\_Simw; Str\_Simw;** { - вызовпроцедуры (двараза)}

**Readkey;**

**End.**

Переменные, описанные VAR после слова PROGRAM , называют **ГЛОБАЛЬНЫМИ.**

Глобальные переменные передаются в любую часть программы, в том числе и в процедуры.

**Program Stroka; Uses crt;**

**var i: integer;** { i - глобальнаяпеременная }

**procedure St\_Sim;**

**begin**

**for i:=1 to 40 do write('$');**

**writeln;**

**end;**

**Begin**

**clrscr;**

**St\_Sim; St\_Sim;**

**End.**

Переменные , описанные в процедуре, называют **ЛОКАЛЬНЫМИ** ,так как они используются только в данной процедуре. Доступа к ним из программы нет.

**ProgramStroka; Usescrt;**

**procedure St\_Sim;**

**var i: integer; {i - локальнаяпеременная}**

**begin**

**for i:=1 to 40 do write('-');**

**end;**

**Begin**

**clrscr;**

**St\_Sim;**

**Readkey;**

**End.**

Нежелательно, чтобы глобальные и локальные переменные совпадали.

Часто процедура содержит список параметров.

**procedure имя (СПИСОК ПАРАМЕТРОВ);**

...

Список параметров - перечень имен для обозначения исходных данных и

результатов работы процедуры с указанием их типов. Параметры, перечисляемые в списке, называют **ФОРМАЛЬНЫМИ**.

При обращении к процедуре, имеющей список параметров, в скобках указываются конкретные значения (выражения) и имена, подставляемые на место формальных параметров процедуры при ее выполнении. При вызове процедуры формальные параметры, указанные в ее заголовке, заменяются значениями в порядке их следования. Количество и тип фактических и формальных параметров должно совпадать.

Вывести на экран строку содержащую задаваемое с клавиатуры число символов.

**ProgramStroka;Usescrt;**

**varm: integer;** {m – глобальная переменная}

**procedure Str\_Sim( n: integer; c: char);** { n, c - параметры-значения }

**var i:integer;** { i - локальная переменная }

**begin**

**for i:=1 to n do write(c);** { конструкцияпечатисимволов }

**writeln;**

**end;**

**Begin**

**clrscr;**

**Str\_Sim(64,'\*');** { вызов процедуры с указанием числа звездочек }

**writeln; writeln;**

**write('Сколько символов напечатать? :'); readln(m);**

**Str\_Sim(m,'=');** { вызов процедуры с указанием числа знаков '='}

**End.**

Параметры-значения нельзя использовать для передачи результатов из процедуры в основную программу ! Если возникла такая необходимость, то применяют формальные параметры-переменные ,перед которыми в списке формальных параметров стоит **var.**

**Program Pr1; Uses crt; Program Pr2; Uses crt;**

**var x: integer; var x: integer;**

**procedure Izm(y: integer); procedure Izm(var y: integer);**

**begin begin**

**y:=1 y:=1**

**end; end;**

**Begin**Y не передает ре- **Begin**  Все изменения формальных

**x:=0;** зультат из проце-  **x:=0;** параметров относятся и

**Izm(x);** дуры в основную  **Izm(x);** к фактическим.

write(x) программу. **write(x)**

**End. End.**

Результат: x=0. Результат: x=1.

# *Примеры решения задач:*

1. Даны фамилия, имя, отчество. Вывести на экран инициалы.

program zadacha\_1;

uses crt;

var

s: string;

procedure fio(s1: string);

var

i: integer;

l: integer;

st: string;

begin

l:=length(s1);

i:=1;

st:=s1[1]+'. ';

for i:=1 to l do

begin

if s1[i]=' ' then

begin

st:=st+s1[i+1]+'. ';

end;

end;

writeln ('Новаястрока');

writeln (st);

readln;

end;

begin

clrscr;

writeln ('Введитефио');

readln (s);

fio (s);

end.

1. Выяснить сколько раз встречается каждая буква алфавита в строке.

program zadacha\_2;

uses crt;

var

st: string [100];

procedure alphavit (st: string);

var

s: string;

k, i,l,j,n: integer;

begin

s: ='абвгдеёжзиклмнопрстуфхцчшщьыъэюя';

l: =length(s);

n: =length (st);

for i: =1 to l do

begin

k: =0;

for j: =1 to n do

begin

if s[i] = st[j] then

begin

k:=k+1;

end;

end;

if k>0 then

begin

writeln('Буква ',s[i],' встречается в предложении ',k,'раз');

readln;

end;

end;

end;

begin

clrscr;

writeln ('Введите предложение');

readln (st);

alphavit (st);

readln;

end.

1. Процедура для решения квадратного уравнения.

program zadacha\_5;

uses crt;

var

a,b,c: integer;

procedure kvur (x,y,z:integer);

var

d:integer;

x1,x2:real;

begin

d:=sqr(y)-4\*x\*z;

if d > 0 then

begin

x1:=(-y+sqrt(d))/2\*x;

x2:=(-y-sqrt(d))/2\*x;

writeln ('x1=',x1:5:2,' x2=',x2:5:2);

end;

if d=0 then

begin

x1:=-y/2\*x;

x2:=x1;

writeln ('x1=',x1:5:2,' x2=',x2:5:2);

end;

if d < 0 then

begin

writeln ('Корнейнет');

end;

readln;

end;

begin

clrscr;

writeln ('Введите кооффициенты квадратного уравнения');

readln (a,b,c);

kvur(a,b,c);

end.

1. Составить процедуру для нахождения суммы элементов некоторого столбца двумерного массива.

program zadacha\_6;

uses crt;

var

m,n: integer;

procedure sum (l,k:integer);

var

a: array [1..30,1..30] of integer;

s,i,j: integer;

begin

for i:=1 to l do

for j:=1 to k do

begin

write('a(',i,',',j,')=');

readln (a[i,j]);

end;

s:=0;

for i:=1 to l do

begin

s:=s+a[i,2];

end;

write('суммаэлементоввторогостолбцаравна',s);

readln;

end;

begin

clrscr;

writeln ('Введите количество строк ');

readln (m);

writeln ('Введите количество столбцов ');

readln (n);

sum(m,n);

end.

1. Составить программу для проверки, является ли счастливым троллейбусный биле, т.е. сумма первых трех цифр билета равна сумме трех последних цифр билета.

program zadacha\_7;

uses crt;

var

x: longint;

procedure stast (y: longint);

var

a,b,c,d,e,f,s1,s2:integer;

begin

y:=abs(y);

a:=y div 100000;

b:=y div 10000 mod 10;

c:=y div 1000 mod 10;

d:=y div 100 mod 10;

e:=y div 10 mod 10;

f:=y mod 10;

s1:=a+b+c;

s2:=d+e+f;

if s1=s2 then

writeln('Билет счастливый')

else

writeln('Билет не счастливый');

readln;

end;

begin

clrscr;

writeln('Введите шестизначный номер троллейбусного билета ');

readln(x);

stast(x);

end.

1. Дан одномерный массив A[N]. Найти max (a2, a4,….,a2k) + min(a1,a3,…,a2k+1).

program zadacha\_8;

UsesCRT

Var

y:integer;

procedure sum(n:integer);

var

a:array[1..30] of integer;

i,min,max:integer;

s:integer;

begin

for i:=1 to n do

begin

write('a[',i,']=');

readln(a[i]);

end;

max:=a[2];

min:=a[1];

for i:=1 to n do

begin

if (a[i] mod 2=0) and (a[i]>max) then

begin

max:=a[i];

end;

if (i mod 2>0) and (a[i]< min) then

begin

min:=a[i];

end;

end;

s:=min+max;

writeln('s=',s);

readln;

end;

begin

clrscr;

writeln('введитеколичествоэлементов');

readln(y);

sum(y);

end.

1. Описать процедуру, которая находит все натуральные числа, не превосходящие заданного числа a, которые делятся на каждую из своих чисел

program zadacha\_9;

uses crt;

var

m:integer;

procedure nat (var a: integer);

label l,q,w;

var

i,x,n,b:integer;

begin

for i:=1 to a do

begin

q: b:=i;

if i>a then

begin

goto w;

end

else

n:=i;

l: x:=n mod 10;

if x=0 then

begin

i:=i+1; goto q;

end

else

if b mod x=0 then

begin

if n=x then

begin

writeln (b);

readln;

i:=i+1; goto q;

end

else

n:=n div 10; goto l;

end

else

n:=n+1;

end;

readln;

w: end;

begin

clrscr;

writeln ('m=');

readln (m);

nat(m);

end.

1. Дана последовательность чисел 1, 2, 3…n (n не больше 100). Среди них найти автоморфное число (число и последние цифры его квадрата равны). Например: 5 и 25, 25 и 625.

program zadacha\_10;

uses crt;

var

procedure schaste (n:integer); {начало подпрограммы}

var

i:integer; {переменная - счетчик, число}

q:integer; {переменная1}

w:integer; {переменная2}

begin

for i:=1 to n do

begin

if (i>0) and (i<10) then

begin

q:=Sqr(i); {переменной1 присваиваем квадрат числа}

w:=q mod 10; {переменной2 присваиваем остаток от

деления квадрата числа на 10}

if i=w then {если число равно остатку от деления, то…}

begin

writeln('число ',i,' автоморфное '); {…вывод автоморфного числа}

end;

end;

if (i>10) and (i<100) then {если число находится в промежутке от 10 до 100}

begin

q:=i\*i;

w:=q mod 100;

if i=w then

begin

writeln('число ', i ,' автоморфное ');

end;

end;

end;

readln;

end;

begin {начало основной программы}

clrscr;

writeln('введите натуральное число ,',n,',

(до которого нужно проверять) не больше 100');

readln(b); {обращение к подпрограмме}

schaste(b);

end.

1. Написать программу нахождения большего и меньшего из трех чисел.

program zadacha\_11;

program a1;

uses crt;

var

r:integer; {1-ечисло}

t:integer; {2-ечисло}

y:integer; {3-ечисло}

procedure god(r,t,y:integer); {процедура для нахождения макс. и мин. среди}

var

max:integer; {максимальное число}

min:integer; {минимальноечисло}

begin

if (r>=t) and (r>=y) then

max:=r;

if (t>=r) and (t>=y) then

max:=t;

if (y>=r) and (y>=t) then

max:=y;

if (r<=t) and (r<=y) then

min:=r;

if (t<=r) and (t<=y) then

min:=t;

if (y<=r) and (y<=t) then

min:=y;

writeln('Максимальное: ',max);

writeln('Минимальное: ',min);

readln;

end;

begin

clrscr;

write('Введитечисла ');

readln(r,t,y);

god(r,t,y); {вызываниепроцедуры}

end.

# **Практическая часть**

*Задачи для самостоятельного решения:*

1.Найти сумму чисел ряда 1,2,3,...,60 с использованием процедуры нахождения суммы и процедуры подчеркивания результата. В обеих процедурах использовать локальные переменные. Количество символов в строке задать глобальной переменной.

2.Найти сумму чисел ряда 1,2,3,4,... от числа n до числа m. Суммирование оформить процедурой с параметрами. Значения n и m программа должна запрашивать.

1. Напишите программу вычисления расстояния между двумя точкамис заданными координатами Х1, Y1,Х2,Y2.
2. Найти произведение элементов квадратной матрицы, расположенных ниже побочной диагонали.
3. Найти периметр четырехугольника по координатам его вершин. Использовать функцию нахождения длины отрезка по его координатам начала и конца.
4. Разработать процедуры для подсчета количества отрицательных элементов одномерного массива вещественных чисел.
5. Разработать процедуры для подсчета суммы квадратов положительных элементов одномерного массива вещественных чисел.
6. Разработать процедуры для подсчета суммы квадратов отрицательных элементов одномерного массива щественных чисел.
7. Посчитать сумму квадратов диагональных элементов двумерного массива вещественных чисел.
8. Написать программу нахождения суммы большего и меньшего из трех чисел.
9. Дано четырехзначное число, в котором все цифры разные. Проверить, равна ли разность чисел, состоящая из первых двух цифр и последних двух цифр сумме всех четырех цифр этого числа.
10. Дан массив. Найти сумму элементов массива, кратных трем.
11. Найти максимальный элемент некоторой строки двумерного массива.
12. Составить подпрограмму проверки трехзначного числа, которое при делении на 47 даёт в остатке 43.
13. Написать процедуру, которая вычисляет объем и площадь поверхности параллелепипеда.

**Контрольные вопросы:**

1. Что такое процедура?
2. Чем отличаются процедуры без параметров от процедур с параметрами?
3. Что такое глобальные и локальные данные? Чем они отличаются?
4. Что такое параметр? Какие бывают параметры?
5. Как устанавливается связь между аргументами и параметрами?
6. Могут ли быть процедуры вложенными? Приведите примеры.
7. Как передать массив в процедуру с помощью метода аргумент-параметр?
8. Объясните разницу в передаче массива в качестве параметра:

const nn=100;

type mas=array[1..nn]of real;

var a:mas;

procedure p1(const b:mas);

begin {телопроцедуры} end;

procedure p2(b:mas);

begin {телопроцедуры} end;

procedure p3(var b:mas);

begin {телопроцедуры} end;

procedure p4(b:array of real);

begin {телопроцедуры} end;

begin {вводмассиваа}

p1(a); {1}

p2(a); {2}

p3(a); {3}

p4(a); {4}

end.

Укажите, в какой точке - первой, второй или третьей - массив может изменить своё значение.

1. Установите, что будет напечатано в результате работы первого и второго варианта программы. Сравнитерезультаты, сделайтевыводы:

program FirstVariant; program SecondVariant;

var x:real; var x:real;

procedure writeX; procedure writeX;

var x:real;

begin write(x) end; begin write(x) end;

begin x:=pi; begin x:=pi;

writeX; writeX;

end. end.

1. Установите, вычисляют ли приведенные ниже процедуры одно и то же значение? Объясните своё решение. Посмотрите, как изменяются описания переменных *p* и *q*.

1) varn,m:integer; {исходные целые числа}

c:integer; {число сочетаний из n по m}

fn,fm,fnm:integer; {переменные для хранения n!, m! и (n-m)!}

p:integer; {хранит факториал}

q:integer; {исходное число для вычисления факториала}

procedure fact1;

var i:integer;

begin p:=1; for i:=1 to q do p:=p\*i end;

begin write('Введите n и m ');

readln(n,m);

q:=n; fact1; fn:=p;

q:=m; fact1; fm:=q;

q:=n-m; fact1; fnm:=p;

write('число сочетаний = ',fndiv (fm\*fnm))

end.

2) varn,m:integer; {исходные целые числа}

c:integer; {число сочетаний из n по m}

fn,fm,fnm:integer; {переменные для хранения n!, m! и (n-m)!}

procedure fact2(q:integer;var p:integer);

var i:integer;

begin p:=1; for i:=1 to q do p:=p\*i end;

begin write('Введите n и m ');

readln(n,m);

fact2(n,fn);

fact2(m,fm);

fact2(n-m,fnm);

write('числосочетаний = ',fn div (fm\*fnm))

end.

1. Определите результат выполнения программы:

procedure XX(n : word);

begin write(n:4);

if n<150 then XX(2\*n);

write(n:4)

end;

begin XX(1);

writeln;

write('Нажмите<Enter>');

readln;

end.

# **Практическое занятие №16 - 17 Организация функций. Использование функций.**

**Цель работы: научиться решать задачи с использование функций**

**Теоретическая часть**

**ФУНКЦИЯ** - это подпрограмма, результат выполнения которой есть единственное скалярное значение, присваиваемое имени этой функции. Функция - частный случай процедур.

ОТЛИЧИЯ от процедур:

1. В описании указывается тип результата функции.

2. Результат выполнения функции - ОДНО значение.

3. Имени функции обязательно присваивается результат.

4. Результат передается в программу как значение имени этой функции.

5. Функция записывается в выражениях, а не отдельной командой как процедура.

Описание функции аналогично описанию процедуры:

**FUNCTION имя ( список формальных параметров ): тип;**

**[var локальные переменные: тип;]**

**BEGIN**

**блок функции**

**END;**

Допускается описание функции без параметров:

FUNCTION имя : тип;

...

Найти сумму чисел ряда 1,2,3,... ,используя функцию.

**Program Summa; usescrt;**

**var n:integer;** { n -глобальная переменная }

**functionSum(m: integer): integer;** { m -параметр-значение функции }

**var i,s: integer;** { i,s - локальныепеременные }

**begin**

**s:=0;**

**for i:=1 to m do s:=s+i;**

**Sum:=s** {- присваивание имени функции значения }

**end;**

**Begin**

**clrscr;**

**write('До какого числа суммировать? :'); readln(n);**

**write('Сумма чисел ряда S=',Sum(n))** { -вызов функции Sum с парам.}

**readkey;**

**End.**

Составить программу вычисления степени числа X с целым показателем n, используя функцию без параметров.

**Program Step; uses crt;**

**var n:integer;**

**x,y:real;**

**function St:real;**

**var i: integer;**

**begin**

**y:=1;**

**for i:=1 to n do y:=y\*x;**

**St:=y**

**end;**

**Begin**

**clrscr;**

**write('Введите число :'); readln(x);**

**write('Введите степень :'); readln(n);**

**if n<0 then begin n:=-n; x:=1/x end;**

**write('Число ',x,' в степени ',n,' равно ',St)**

**readkey;**

**End.**

*Пример решения:*

1.Определить является ли число простым.

program zadacha\_16;

Uses crt;

var

b,k:integer;

procedure prost(var a,k:integer);

var

i:integer;

begin

for i:=1 to a do

begin

if a mod i=0 then

begin

k:=k+1;

end;

end;

writeln('Количество делителей числа ',b,' равно ',k);

ifk=2 then

begin

writeln('число ',b, ' является простым');

readln;

end

else

begin

writeln('число ',b, ' не является простым');

readln;

end;

end;

begin

clrscr;

write('Введите число - ');

read(b);

prost(b,k);

readln;

end.

2.Составть функции для нахождения длины окружности и площади круга.

program zadacha\_3;

uses crt;

var

r: real;

dl: real;

pl: real;

funchion dlina (l: real): real;

var

c: real;

begin

c:=2\*Pi\*l;

end;

funchion plo (l: real): real;

var

s: real;

begin

s:=Pi\*l\*l;

end;

begin

clrsscr;

writeln ('Введитерадиус');

readln (r);

dl:=dlina(r);

pl:=plo(r);

writeln ('Длина окружности равна ',dl);

writeln ('Площадь круга равна ',pl);

readln;

end.

3.Описать функцию, которая в массиве A[n] находит второе по величине число (т.е. меньше максимального, но больше всех других элементов)

program zadacha\_4;

uses crt;

var

m,x:integer;

function mas (n:integer):integer;

var

a: array [1..30] of integer;

i,j,k: integer;

begin

randomize;

for i:=1 to n do

begin

a[i]:=random(100);

end;

for i:=1 to n do

begin

writeln ('a[',i,']=',a[i]);

read;

end;

for i:=1 to n do

for j:=1 to n-1 do

begin

if a[j] < a[j+1] then

begin

k:=a[j];

a[j]:=a[j+1];

a[j+1]:=k;

end;

end;

writeln ('второе по величине число=',a[2]);

readln;

end;

begin

clrscr;

write ('введите количество элементов в массиве');

readln (x);

m:=mas (x);

end.

4.Рассчитать значение х, определив и использовав необходимую функцию:  15

programzadacha\_15;

usescrt;

var

n,n2,n1: integer; {числа}

x: real; {значение}

function x1(y1,y2: integer):real;

begin

x1:=(y1+sin(y1))/(sin(y2)+y2); {функция}

end;

begin

clrscr;

n:=1;

n1:=7;

n2:=3;

x:=x1(n,(n+3))+x1(n1,(n1-2))+x1(n2,(n2-1));

writeln('');

writeln ('x=(1+sin4)/(sin1+4)+(7+sin5)/(sin5+7)+(3+sin2)/(sin3+2) ');

writeln(x:2:5);

readln;

end.

1. Рассчитать значение х, определив и использовав необходимую функцию:  14

program zadacha\_14;

uses crt;

var

f,l,t:integer;

x:real;

function k(n,m: integer):real;

begin

k:=(n+sin(m))/(sin(n)+m);

end;

begin

clrscr;

f:=2;

l:=1;

t:=4;

x:=k(f,(f+1))+k(l,(l+4))+k(t,(t+3));

writeln('Значениефункции');

writeln('((2+sin(3))/(sin(2)+3)+(1+sin(5)/sin(1)+5)+(sin(7)+4)/sin(3)+7)');

writeln('x=',x:1:1);

readln;

end.

1. В последовательности чисел определить четные. Использовать функцию, определяющую четное число или нет.

program zadacha\_13;

uses crt;

var

a: array [1..100] of integer;

n,i,z: integer;

function chot(m: integer):integer;

var

x: integer;

begin

if m mod 2=0 then

begin

writeln ('Число ',m,' чётное');

end;

end;

begin

clrscr;

write ('введите количество чисел в последовательности');

readln (n);

for i:=1 to n do

begin

write ('a[',i,']=');

readln (a[i]);

end;

for i:=1 to n do

begin

z:=chot(a[i]);

end;

readln;

end.

**Практическая часть**

*Задачи для самостоятельного решения*

1. Описать процедуру Minmax(A,B), записывающую в переменную A минимальное из значений A и B, а в переменную B — максимальное из этих значений (A и B — вещественные параметры, являющиеся одновременно входными и выходными). Используя четыре вызова этой процедуры, найти минимальное и максимальное из ·исел A, B, C, D.
2. Описать функцию Fact(N) целого типа, вычисляющую значение факториала N! = 1·2·...·N (N > 0 — параметр целого типа). С помощью этой фунЄции вычислить факториалы 10 данных чисел.
3. Описать функцию FactR(N) вещественного типа, позволяющую вычислять приближенное значение факториала N! = 1·2·...·N для целых N (> 0). С помощью этой функции вычислить факториалы пяти данных чисел.
4. Описать функцию Fact2(N) целого типа, вычисляющую значение "двойного факториала": N!! = 1·3·5·...·N, если N — нечетное, N!! = 2·4·6·...·N, если N — четное (N > 0 — параметр целого типа). С помощью этой функции вычислить двойные факториалы десяти данных чисел.
5. Описать процедуру SumDigit(N,S), находящую сумму цифр S целого числа N (N — входной, S — выходной параметр). Используя эту процедуру, найти суммы цифр пяти данных чисел.
6. Описать функцию Otr(Ax,Ay,Bx,By) вещественного типа, находящую длину отрезка AB на плоскости по координатам его концов: |AB| = sqrt((Ax-Bx)2 + (Ay-By)2) (Ax, Ay, Bx, By — вещественные параметры). С помощью этой функции найти длины отрезков AB, AC, AD, если даны координаты точек A, B, C, D.
7. Используя функцию Otr из задания 20, описать функцию Perim(Ax,Ay,Bx,By,Cx,Cy) вещественного типа, находящую периметр треугольника ABC по координатам его вершин (Ax, Ay, Bx, By, Cx, Cy — вещественные параметры). С помощью этой функции найти периметры треугольников ABC, ABD, ACD, если даны координаты точек A, B, C, D.
8. Используя функции Otr и Perim из заданий 20 и 21, описать функцию Area(Ax,Ay,Bx,By,Cx,Cy) вещественного типа, находящую площадь треугольника ABC по формуле Герона:
9. SABC = sqrt(p·(p–|AB|)·(p–|AC|)·(p–|BC|)), где p — полупериметр. С помощью этой функции найти площади треугольников ABC, ABD, ACD, если даны координаты точек A, B, C, D.
10. Используя функции Otr и Area из заданий 20 и 22, описать процедуру Dist(Px,Py,Ax,Ay,Bx,By,D), находящую расстояние D от точки P до прямой AB по формуле D = 2SPAB / |AB|, где SPAB — площадь треугольника PAB. С помощью этой процедуры найти расстояния от точки P до прямых AB, AC, BC, если даны координаты точек P, A, B, C.
11. Используя процедуру Dist из задания 23, описать процедуру Heights(Ax,Ay,Bx,By,Cx,Cy,hA,hB,hC), находящую высоты hA, hB, hC треугольника ABC, проведенные соответственно из вершин A, B, C. С помощью этой процедуры найти высоты треугольников ABC, ABD, ACD, если даны координаты точек A, B, C, D.
12. Описать функцию Min2(A,B) вещественного типа, находящую минимальное из двух вещественных чисел A и B. С помощью этой функции найти минимальные из пар чисел A и B, A и C, A и D, если даны числа A, B, C, D.

**Контрольные вопросы:**

1. Что такое описание процедуры или функции и чем они отличается от оператора процедуры или указателя функции?  Могут ли быть процедуры и функции вложенными? Приведите примеры
2. Напишите функцию без параметров и функцию с параметрами, которая находит номер максимального элемента последовательности, заданной формулой общего члена: *ai=F(i,x)*. Вычисление очередного элемента последовательности оформите в виде функции.
3. Найдите все ошибки в следующем фрагменте программы:

type M = set of char;

function f(a,b : M; x:char):M;

begin if a\*b<0

then a:=[x]

else if a<b

then a:=b+x

else if ord(x) in a-b

then a:=a-[x..'<='];

f:=a+b

end.

1. Укажите правильные описания функций для вычисления факториала:

1) function f(n:integer):integer;

begin f:=n\*f(f(n-1) end;

2) function f(n:integer):integer;

begin if n=0 then f:=1 elsee f:=f(n+1)/(n+1) end;

3) function f(var n:integer):integer;

begin if n=0 then f:=1 else f:=f(n-1)\*n end;

4) function f(n:integer):integer;

begin if n=0 then f:=1 else f:=n\*(n-1)\*f(n-2) end;

5) function f(n:integer):integer;

begin if n<=1 then f:=1 else f:=n\*(n-1)\*f(n-2) end;

6) function f(n:integer):integer;

begin if n=0 then f:=1 else f:=n\*f(n-1) end.

1. Рассмотрите следующее описание функции:

function f(x,y:real):real;

begin if x>=y

then f:=(x+y)/2

else f:=f(f(x+2,y-1),f(x+1,y-2))

end.

Определите *f*(1,10). Каким более простым способом можно представить и вычислить *f*(a,b)?

1. Что будет напечатано после выполнения программы:

var a:integer;

function f(var x:integer):integer;

begin f:=x; x:=0 end;

begin a:=1;

writeln(f(a)-f(a)) end.

7. Напишите процедуру без параметров, процедуру с параметрами и функцию, которые находят номер максимального элемента последовательности, заданной формулой общего члена: *ai=F(i,x)*. Вычисление очередного элемента последовательности оформите в виде функции.

**СООБЩЕНИЯ И КОДЫ ОШИБОК**

Среда Турбо Паскаля дает исчерпывающую информацию о характере и месте обнаруженной компилятором ошибки. При обнаружении ошибки среда автоматчики загружает в окно редактора исходный файл и помещает курсор около того места, где в исходном тексте обнаружена ошибка. При этом в верхней строке редактора появляется диагностическое сообщение. После нажатия на любую клавишу (кроме F1) верхняя строка восстанавливает свой первоначальный вид, и среда переходит к режиму редактирования. Если после появления сообщения об ошибке нажать F1, на экране появится окно справочной службы с легальной информацией об ошибке и рекомендациями по ее устранению. Некоторые ошибки в исходном тексте обнаруживаются не сразу, а в ходе продолжающегося контекстного разбора. Например, несоответствие типов в операторе присваивания не может быть обнаружено до тех пор, пока не будет вычислено целиком выражение в правой части этого оператора. В таких случаях ищите ошибку слева от курсора или в предыдущей строке текста.

Ниже приводятся сообщения об ошибках в том виде, в каком они появляются в верхней строке редактора, а также перевод сообщений справочной службы.

**1. Out of memory (Выход за границы памяти).** Компилятору не хватает памяти. Имеется ряд возможных решений этой проблемы:

* если в опции COMPILE / DESTINATION установлено значение MEMORY, замените эту опцию на DISK;
* если в опции OPTIONS / LINKER / LINK BUFFER установлено значение MEMORY, замените эту опцию на DISK;
* если Вы используете резидентные обслуживающие программы, такие как SIDEKICK, NORTON, удалите их из памяти;
* если Вы используете интегрированную среду TURBO.EXE, то попробуйте воспользо-ваться компилятором ТРС.ЕХЕ - он занимает меньше памяти.

Если ни одна из рекомендаций не помогает, то, возможно, Ваша программа просто слишком велика, чтобы компилировать ее в таком объеме памяти. В этом случае Вы должны разбить ее на два или более модулей.

**2. Identifier expected (He указанидентификатор).** В том месте должен находиться идентификатор. Возможно, Вы пытаетесь использовать в качестве идентификатора зарезервированное слово.

**3. Unknown identifier (Неизвестный идентификатор).** Этот идентификатор не был описан.

**4. Duplicate identifier (Двойной идентификатор).** Попытка дважды описать один и тот же идентификатор.

**5. Syntax error (Синтаксическая ошибка).** В исходном тексте найден недопустимый символ. Возможно, Вы забили заключить в апострофы строковую константу.

**6. Error in real constant (Ошибка в вещественной константе).** Синтаксис констант вещественного типа

**7. Error in integer constant (Ошибка в целой константе.).** Учтите, что после целых действительных чисел, превышающих диапазон представления целых чисел (-214748З648…+2147483647), должны ставиться точка и ноль, например, 12345678912.0.

**8. String constant exceeds line (Строковая константа превышает допустимые размеры).** Вероятно, Вы забыли поставить апостроф в конце строковой константы.

**9. Too many nested files (Слишком много вложенных файлов).** Компилятор допускает не более пяти уровней вложенности включаемых файлов (т.е. в исходном файле есть ссылка на включаемый файл, в котором есть ссылка на еще одни включаемый файл и т.д.).

**10. Unexpected end of file (He найденконецфайла).**Вы могли получить это сообщение об ошибке по одной из следующих причин:

* Ваш исходный файл закончился перед последним END основного раздела операторов;
* вероятно, в Вашей программе неодинаковое количество операторов BEGIN и END;
* включаемый файл заканчивается в середине раздела операторов;
* каждый раздел операторов должен целиком помещаться в одном файле;
* Вы не закончили комментарий.

**11. Line too long (Слишком длинная строка).**Максимальная длина строки, обрабатываемой компилятором, равна 126 символам (обратите внимание: редактор среды может обрабатывать строки практически неограниченной длины).

**12. Type identifier expected (Здесь нужен идентификатор типа).** Не указан тип идентификатора.

**13. Too many open files (Слишком много открытых файлов) .** Появление такой ошибки означает, что конфигурационный файл CONFIG.SYS операционной системы не включает параметр FILES=xx или этот параметр указывает слишком мало файлов. Увеличьте число файлов до нужного значения, например, до 80.

**14. Invalid file name (Неверное имя файла).** Имя файла неверно или указан несуществующий путь.

**15. File not found (Файлненайден).**Файл не был найден в просмотренных каталогах.

**16. Disk full (Диск заполнен).** Удалите некоторые файлы или воспользуйтесь новым диском.

**17. Invalid compiler directive (Неправильная директива компилятора).** Неверная буква в директиве компилятора, один из параметров директивы компилятора неверный, или Вы пользуетесь глобальной директивой компилятора, когда компиляция тела программы уже началась.

**18. Too many files (Слишком много файлов).** В компиляции программы или программного модуля участвуют слишком много файлов. Попытайтесь не использовать так много файлов, например, объединяя включаемые файлы или делая короче имена файлов.

**19. Undefined type in pointer definition (Неопределенный тут в объявлении указателя).** Попытка объявить типизированный указатель, связанный с ранее не объявленным типом данных.

**20. Variable identifier expected (Отсутствует идентификатор переменной).** На этом месте должен быть идентификатор переменной.

**21. Error in type (Ошибка в объявлении типа).** Объявление типа не может начинаться с этого символа.

**22. Structure too large (Слишком большая структура).** Максимально допустимый размер любого структурного типа составляет 65520 байт.

**23. Set base type of range (Базовый тип множества нарушает границы).** Базовый тип множества должен представлять собой тип-диапазон с границами в пределах от 0 до 255 или перечисляемый тип с не более чем 256 значениями.

**24. File components may not be files (Компонентамифайланемогутбытьфайлы).** Конструкции типа file of file (файл файлов) или file of object (файл объектов) не допускаются. Нельм объявлять любые структурные типы, которые используют в качестве компонентов объекты или файлы.

**25. Invalid string length (Неверная длина строки).** Объявляемая длина строки должна находиться в диапазоне от 1 до 255.)

**26. Type mismatch (Несоответствие типов).** Это сообщение может быть вызвано следующими причинами:

* несовместимые типы переменной и выражения в операторе присваивания;
* несовместимые типы фактического и формального параметров в обращении к процедуре или функции;
* тип выражения не совместим с типом индекса при индексировании массива;
* несовместимые типы операндов в выражении.

**27. Invalid subrange base type (Неправильный базовый тип для типа-диапазона).** Допустимыми базовыми типами являются все порядковые типы.

**28. Lower bound greater than upper bound (Нижняя граница больше верхней).** Описание типа-диапазона содержит неправильные границы.

**29. Ordinal type expected (Нужен порядковый тип).** Вещественные, сороковые, структурные, процедурные типы и указатели в данном месте программы не допускаются.

**30. Integer constant expected (Нужнацелаяконстанта).**

**31. Constant expected (Нужнаконстанта).**

**32. Integer or real constant expected (Нужнацелаяиливещественнаяконстанта).**

**33. Type identifier expected (Нуженидентификатортипа)**

**34. Invalid function result type (Неправильный тип результата функции).** Правильными типами результата функции являются все простые типы, строковые типы и указатели.

**35. Label Identifier expected (Нуженидентификаторметки).**Meтка не обозначена с помощью идентификатора, как это требуется из контекста программы.

**36. BEGIN expected (Нужен BEGIN).**

**37. END expected (Нужен END).**

**38. Integer expression expected (Нужновыражениетипа INTEGER).**

**39. Ordinal expression expected (Нужновыражениеперечисляемоготипа).**

**40. Boolean expression expected (Нужновыражениетипа BOOLEAN).**

**41. Operand types do not match operator (Типы операндов не соответствуют операции).** Данная операция не может быть применена к указанным операндам, например, ' А' div ' 2 ' .

**42. Error in expression (Ошибкаввыражении).**Данный символ не может участвовать в выражении указанным образом. Возможно, Вы забыли указать операцию между двумя операндами.

**43. Illegal assignment (Неверное присваивание).** Файлам и нетипизированным переменным нельзя присваивать значения. Идентификатору функции можно присвоить значение только внутри раздела операторов данной функции

**44. Field identifier expected (Нуженидентификаторполя).** Попытка использовать запись целиком в том месте, где требуется ссылка на какое-либо поле записи.

**45. Object file too large (Объектный файл слишком большой).** Турбо Паскаль не может компоновать файлы .OBJ больше 64 Кбайт.

**46. Undefined external (Неопределенная внешняя процедура).** Внешняя процедура или функция не имеет соответствующего определения PUBLIC в объектном файле. Убедитесь, что Вы указали все объектные файлы в директивах {$L <имя .OBJ файла>} и проверьте написание идентификаторов процедуры или функции в файле. ASM.

**47. Invalid object file record (Неправильная запись объектного файла).** Файл OBJ содержит неверную объектную запись. Убедитесь, что данный файл является действительно файлом .OBJ.

**48. Code segment too large (Сегмент кода слишком большой).** Максимальный размер кода программы или программного модуля равняется 65520 байтам. Разбейте Вашу программу или модуль на два или более модулей.

**49. Data segment too large (Сегмент данных слишком велик).** Максимальный размер сегмента данных программы равен 65520 байтам, включая данные, используемые программными модулями. Если Вам нужно большее количество глобальных данных, опишите большие структуры с помощью указателей и выделяйте для них память динамически с помощью процедуры NEW.

**50. DO expected (Нуженоператор DO).**

**51. Invalid PUBLIC definition (Неверное PUBLIC-определение).** Возможные причины сообщении:

* данный идентификатор получил тип PUBLIC с помощью соответствующей директивы языка ассемблера, но не cooтветствует описанию EXTERNAL в программе или программном модуле Паскаля;
* две или более директивы PUBLIC языка ассемблера определяют один и тот же идентификатор;
* файлы.OBJ определяют символы PUBLIC, не находящиеся в сегменте CODE.

**52. Invalid EXTRN definition (Неправильное EXTRN-определение).** Возможные причины сообщения:

* программа на ассемблере ссылается с помощью директивы EXTRN на идентификатор, который не описан в программе на Паскале и не был описан в интерфейсных секциях используемых программных модулей;
* ассемблерная программа ссылается на идентификатор, обозначающий абсолютную переменную (т.е. определенную словом ABSOLUTE);
* ассемблерная программа ссылается на идентификатор процедуры или функции типа INLINE.

**53. Too many EXTRN definition (Слишкоммного EXTRN-определений).** Турбо Паскаль не может обрабатывать файлы. OBJ при более чем 256 определениях EXTRN.

**54. OF expected (Требуется OF).**

**55. INTERFACE expected (Требуется интерфейсная секция).**

**56. Invalid relocatable reference (Неправильная перемещаемая ссылка).** Возможные причины сообщения:

* файл .OBJ содержит данные и перемещаемые ссылки в сегментах, отличных от CODE. Например, Вы пытаетесь описать инициализированные переменные в сегменте DATA;
* файл .OBJ содержит ссылки с размерами в байтах на перемещаемые символы. Такая ошибка происходит в случае, если Вы используете, операторы HIGH и LOW с перемещаемыми символами или если Вы ссылаетесь в директивах DB на перемещаемые символы;
* операнд ссылаемся на перемещаемый символ, который не был определен в сегменте CODE или а сегменте DAТА;
* операнд ссылается на процедуру EXTRN или функцию EXTRN со сдвигом, например, CALL SortProc+8,

**57. THEN expected (Требуется THEN).**

**58. TO or DOWNTO expected (ТребуетсяТОили DOWNTO).**

**59. Undefined forward (Неопределенное опережающее описание).** Возможные причины сообщения:

* была описана процедура иди функция в интерфейсной секции программного модули, но ее определение отсутствует в секции реализации;
* процедуры или функции были описаны с помощью опережающего описания, но их определение не кайлено.

**60. Too many procedures (Слишкоммногопроцедур).** Турбо Паскаль допускает не более 512 процедур или функций в одном модуле. Если Вы компилируете программу, то поместите некоторые процедуры или функции в модули. Если Вы компилируете модуль, то разбейте его на два или несколько модулей.

**61. Invalid typecast (Неверное преобразование типа).** Возможные причины сообщения:

* Вы пытаетесь разместить в памяти, занимаемой некоторой переменной, значение выражения другого типа в случае, когда размер размещаемого значения не равен размеру переменной;
* Вы пытаетесь осуществить преобразование типа выражения, когда разрешается только ссылка на переменную, процедуру или функцию.

**62. Division by zero (Деление на ноль).** Предшествующая операция пытается выполнить деление на ноль.

**63. Invalid file type (Неверный файловый тип).** Данный файловый тип не обслуживается процедурой обработки файлов. Например, процедура READLN используется для типизированного файла или процедура SEEK - для текстового файла.

**64. Cannot Read or Write variables of this type (Нет возможности считать или записать переменные данного типа).** Нарушены следующие ограничения:

* процедуры READ и KEADLN могут считывать переменные символьного, целого, действительного и строкового типов;
* процедуры WRITE и WRITELN могут выводить переменные символьного, целого, действительного, булевского и строкового типов.

**65. Pointer variable expected (Нужно использовать переменную-указатель).** Предыдущая переменная должна быть указателем.

**66. String variable expected (Нужна строковая переменная).** Предшествующая переменная должна иметь строковый тип.

**67. String expression expected (Нужно выражение строкового типе).** Предшествующее выражение должно иметь строковый тип.

**68. Circular unit reference (Перекрестная ссылка модулей).** Два модуля ссылаются друг на друга:

Unit A; Unit В;

Uses В; Uses A;

**69. Unit name mismatch (Несоответствие имен программных модулей).** Имя программного модуля, найденное в файле .TPU, не соответствует имени, указанному в предложении USES.

**70. Unit version mismatch (несоответствие версий модулей).** Один или несколько программных модулей, используемых данной программой, были изменены после их компиляции. Воспользуйтесь опцией COMPILE/MAKE или COMPILE/BUILD в интегрированной среде или опциями /М или /В в компиляторе ТРС, что позволит автоматически скомпилировать программные модули, нуждающиеся в перекомпиляции.

**71. Duplicate unit name (Повторное имя программного модуля).** Вы уже указали этот программный модуль в операторе USES.

**72. Unit file format error (Ошибкаформатафайламодуля).**Файл .TPU не соответствует стандарту Турбо Паскаля.

**73. IMPLEMENTATION expected (Отсутствует исполняемая часть модули).**

**74. Constant and case types do not match (Типы констант и тип выражения CASE не соответствуют друг другу).** Тип константы оператора CASE не совместим с выражением в операторе варианта.

**75. Record variable expected (Нужна переменная типа запись).** Предшествующая переменная должна иметь тип запись.

**76. Constant out of range (Константа нарушает границы).** Возможные причины сообщения:

* Вы пытаетесь указать индекс массива, выходящий за его границы;
* Вы пытаетесь присвоить переменной значение, выходящее за границы, допустимые для переменной;
* Вы пытаетесь передать в качестве фактического параметра процедуре или функции константу, выходящую за границы, допустимые для типа соответствующего формального параметра.

**77. File variable expected (Нужна файловая переменная).** Предшествующая переменная должна иметь файловый тип.

**78. Pointer expression expected (Нужно выражение типа указатель).** Предшествующее выражение должно иметь тип указателя.

**79. Integer or real expression expected (Нужно выражение вещественного или целого типа).** Предшествующее выражение должно иметь тип REAL или INTEGER.

**80. Label not within current block (Метка не находится внутри текущего блока).** Оператор GOTO не может ссылаться на истку, находящуюся вне текущего блока.

**81. Label already defined (Метка уже определена).** Данная метка уже помечает оператор.

**82. Undefined label in processing statement part (Неопределенная метка в предшествующем разделе операторов).** Данная метка была описана, и на нее осуществлялась ссылка в предшествующем разделе опера она не указана в тексте программы.

**83. Invalid @ argument (Неправильный аргумент операции @).** Правильными аргументами являются идентификаторы переменных, процедур и функций.

**84. Unit expected (Нужно кодовое слово UNIT).**

**85. “;” expected (Нужно указать «;»).**

**86. “:” expected (Нужно указать”:”).**

**87. “,” expected (Нужно указать «,»).**

**88. “(” expected (Нужно указать “(”).**

**89. «)» expected (Нужно указать «)»).**

**90. «=» expected (Нужно указать “=”).**

**91. “:=” expected (Нужно указать":=”).**

**92. “[” or “(.” expected (Нужно указать “[” или “(.”).**

**93. “]” or “.)” expected (Нужно указать"]” или «.)»).**

**94. “.” expected (Нужно указать «.»).**

**95. “..” expected (Нужно указать «..»).**

**96. Too many variables (Слишком много переменных).** Нарушены следующие ограничения:

* общин размер глобальных переменных, описанных в программе или программной модуле, не может превышать 64 Кбайт;
* размер локальных переменных, описанных в процедуре или функции, не может превышать 64 Кбайт.

**97. Invalid FOR control variable (Неправильный параметр цикла оператора FOR).** Параметр цикла оператора FOR должен быть переменной порядкового типа, определенной в разделе описаний текущей подпрограммы.

**98. Integer variable expected (Нужна переменная целого типа).** Предшествующая переменная должна иметь целый тип.

**99. File and procedure types are not allowed here (Здесь не могут использоваться файлы или процедурные типы).** Типизированная константа не может иметь файловый или процедурный тип.

**100. String length mismatch (Несоответствие длины строки).** Длина строковой константы не соответствует количеству элементов символьного массива.

**101. Invalid ordering of fields (Неверныйпорядокполей).**Поля в константе типа запись должны записываться в порядке их описания.

**102. String constant expected (Нужнаконстантастроковоготипа).**

**103. Integer or real variable expected (Нужнапеременнаятипа INTEGER или REAL).**Предшествующая переменная должна иметь целый или вещественный тип.

**104. Ordinal variable expected (Нужна переменная порядкового типа).** Предшествующая переменная должна иметь порядковый тип.

**105. INLINE error (Ошибка в операторе INLINE).** Оператор «<» не допускается в сочетании с перемещаемыми ссылками на переменные. Такие ссылки всегда имеют размер в слово.

**106. Character expression expected (Предшествующее выражение должна иметь символьный тип).**

**107. Too many relocation items (Слишком много перемещаемых элементов).** Размер таблицы перемещения файла .ЕХЕ превышает 64 Кбайта, что является верхним пределом в Турбо Паскале. Если Вы обнаружили эту ошибку, то это значит, что программа просто слишком велика для обработки компоновщиком Турбо Паскаля. Возможно также, что она слишком велика для выполнения в MS-DOS. В таком случае нужно выделить в программе основной раздел, который выполнял бы обращение к двум или более вспомогательным разделам с помощью процедуры ЕХЕС из модуля DOS.

**108. Overflow in arithmetic operator (Переполнение при выполнении арифметического оператора).** Результат предыдущей арифметической операции не лежит в диапазоне -2146483648… +2147483647. Исправьте операцию или используйте вещественные типы вместо целочисленных.

**109. No enclosing FOR, WHILE or REPEAT statement (Нетоператоров, заканчивающихоператоры FOR, WHILE или REPEAT).**Процедуры BREAK и CONTINUE не могут вызываться вне тела оператора цикла.

**110. Debug information table overflow (Переполнение информационной таблицы отладки).** Возможно, программа содержит более 65536 имен или 65536 строк. Отключите генерацию таблиц отладки директивой компилятора {SD-} или исправьте один или более модулей.

**111.** Ошибка с этим кодом не описана в версии 7.0 системы Турбо Паскаль.

**112. CASE constant out of range (Константа CASE нарушает допустимые границы).** Целочисленные константы оператора CASE должны находиться в диапазоне от -32768 до 32767.

**113. Error in statement (Ошибкавоператоре).** Данный символ не может быть первым символом в операторе.

**114. Cannot call an interrupt procedure (Невозможно вызвать процедуру прерывания).** Вы не можете непосредственно вызвать процедуру обработки прерывания.

**115.** Ошибка с этим кодом не описана в версии 7.0 системы Турбо Паскаль.

**116. Must be in 8087 mode to compile this (Длякомпиляциинеобходимрежим 8087).**Данная программа может быть скомпилирована только в режиме {SN+}. В состоянии {SN-} операции с типами SINGLE, DOUBLE EXTENDED и COMP не допускаются.

**117. Target address not found (Указанный адрес не найден).** Команда COMPILE / FIND ERROR в среде Турбо Паскаля (или поиск с помощью опции /F в командной строке компилятора ТРС.ЕХЕ) не обнаружила оператор, соответствующий заданному адресу.

**118. Include files are not allowed here (Здесьнедопускаютсявключаемыефайлы).**Раздел операторов должен целиком размешаться в одном файле.

**119. No inherited methods are accessible here (В этом месте программы нет унаследованных методов).** Вы используете зарезервированное слово INHERITED вне метода объекта или в методе, который не унаследован от родительского объекта.

**120.** Ошибка с этим кодом не описана в версии 7.0 системы Турбо Паскаль.

**121. Invalid qualifier (Неверный квалификатор).** Возможные причины сообщения:

* Вы пытаетесь индексировать переменную, которая не является массивом;
* Вы пытаетесь укачать поля в переменной, которая не является записью;
* Вы используете в качестве указателя переменную, которая не является указателем.

**122. Invalid variable reference (Недействительная ссылка на переменную).** Предыдущая конструкция удовлетворяет синтаксису ссылки на переменную, но она не указывает адрес памяти. Возможно, Вы вызываете функцию-указатель, но забываете сослаться на результат с помощью знака.

**123. Too many symbols (Слишком много символов).** Программа или программный модуль содержат более 64 Кбайт символов. Если Вы компилируете программу с директивой {SD+}, то попробуйте отключить эту директиву или разбейте программу на несколько модулей.

**124. Statement part too large (Слишком большой раздел операторов).** Турбо Паскаль ограничивает размер раздела операторов, но величины примерно 24 Кбайта. Если Вы обнаружили эту ошибку, поместите части раздела операторов в одну или несколько процедур и вообще сделайте Вашу программу более структурированной.

**125.** Ошибка не этим колом не описана в версии 7.0 системы Турбо Паскаль.

**126. Piles must be war parameters (Файлы должны передаваться как параметры-переменные).** Вы пытаетесь передать процедуре или функции параметр-значение файлового типа. Параметры файлового типа должны быть параметрами-переменными.

**127. Too many conditional symbols (Слишком много условных символов) .** Недостаточно памяти для определения условных символов (слов, управляющих командами условной компиляции). Попытайтесь удалить некоторые символы или уменьшить их длину.

**128. Misplacedconditionaldirective (Пропущена условная директива).** Компилятор обнаружил директиву {$ELSE} или {$ENDIF} без соответствующих директив {$IFDEF},. {$1FNDEF) или {$IFOPT}.

**129. ENDIF directive missing (Пропущена директива EHDIF).** Исходный файл закончился внутри конструкции условной компиляции. В исходном файле должно быть равное количество директив {$IFxxx} и {$ENDIF}.

**130. Error in initial conditional defines (Ошибкавусловныхопределениях).**Исходные условные символы, указанные в опции OPTIONS/COMPILER / CONDITIONAL DEFINES являются недействительными. Турбо Паскаль требует нуля или более идентификаторов, разделенных пробелами, запятыми или точками с запятой.

**131. Header does not match previous definition (Заголовок не соответствует предыдущему определению).** Возможные причины сообщения:

* заголовок процедуры или функции, указанный в интерфейсной секции, не соответствует заголовку в исполняемой части.
* заголовок процедуры или функции, указанный с помощью опережающего описания FORWARD, не соответствует заголовку найденной далее одноименной процедуры или функции.

**132. Critical disk error (Критическая ошибка диска).** Но время компиляции произошла критическая ошибка диска (например, дисковод находится в состоянии “не готов”).

**133. Cannot evaluate this expression (Нельзявычислитьданноевыражение).**В выражении-константе или в отладочном выражении Вы пытаетесь использовать неподдерживаемые средства, например, я описании константы пытаетесь использовать функцию SIN или вызвать в отладочном выражении определенную пользователем функцию.

**134. Expression incorrectly terminated (Некорректное завершение выражения).** Контекстуально в данном месте программы должен быть конец выражения или оператора.

**135. Invalid format specifier (Неверный спецификатор формата) .** Используется неверный спецификатор формата или числовой аргумент спецификатора формата выходит за допустимые границы.

**136. Invalid indirect reference (Недопустимая косвенная ссылка).** Оператор пытается осуществить недопустимую косвенную ссылку. Например, Вы используете абсолютную переменную, базовая переменная которой в текущем модуле неизвестна, или используете программу типа INLINE, в которой делается ссылка на переменную, неопределенную в текущем модуле.

**137. Structured variable are not allowed here (Здесь нельзя использовать переменную структурного типа).** Делается попытка выполнить нал переменной структурного типа неподдерживаемую операцию. Например. Вы пытаетесь перемножить две записи.

**138. Cannot evaluate without System unit (Нельзя вычислить выражение без модуля SYSTEM).** Чтобы отладчик смог вычислить выражение, в файле TURBO.TPL должен содержаться модуль SYSTEM.

**139. Cannоt access this symbol (Нет доступа к данному символу).** Как только Вы скомпилируете программу, все множество ее символов становится доступным. Однако к отдельным символам (например, к переменным) нельзя получить доступ, пока Вы не запустите программу.

**140. Invalid floating-point operation (Недопустимая операция с плавающей запятой).** При выполнении операции с плавающей запятой произошло переполнение или деление на ноль.

**141. Cannot compile overlay to memory (Нельзя выполнить компиляцию оверлейных модулей в память).** Программа, использующая оверлейные модули, должна компилироваться на диск.

**142. Procedure or function variable expected (Должна использоваться переменная процедурного типа).** В этом контексте оператор получения адреса @ может использоваться только с переменной процедурного типа.

**143. Invalid procedure or function reference (Недопустимаяссылканапроцедуруилифункцию).**Возможные причины сообщения:

* Вы пытаетесь вызвать процедуру в выражении;
* процедура или функция, использующаяся в качестве параметра вызова другой процедуры или функции, должна компилироваться в состоянии {$F+} и не может описываться с помощью зарезервированных слов INLINE или INTERRUPT.

**144. Cannot overlay this unit (Этот модуль не может использоваться в качестве оверлейного).** Попытка использовать в качестве оверлейного модуля, который не был скомпилирован с директивой {$O+}.

**145. Too many nested scopes (Слишком много вложений).** В программе не может быть больше 512 вложений с не более чем 128 вложениями в каждом модуле. Вложениями считаются:

* каждый модуль в предложении USES;
* каждая вложенная запись в типе RECORD;
* каждый вложенный оператор WITH.

**146. File access denied (Отказано в доступе к файлу).** Возможные причины:

* Вы пытаетесь использовать файл с атрибутом "только для чтения” в качестве выводного файла;
* Вы используете имя каталога вместо имени выводного файла.

**147. Object type expected (Здесьдолженбытьтип OBJECT).**Этот идентификатор должен принадлежать к типу OBJECT.

**148. Local object types are not allowed (Нельзяобъявлятьлокальныеобъекты).**Нельзя объявить объект в процедуре (функции).

**149. VIRTUALexpected (Пропущено слово VIRTUAL).**

**150. Method identifier expected (Пропущен идентификатор инкапсулированного правила).**

**151. Virtual constructor are riot allowed (Конструктор не может быть виртуальным).**

**153. Destructor identifier expected (Пропущен идентификатор деструктора).**

**154. Fail only allowed within constructor (Обращение к стандартной процедуре FAIL может содержаться только в конструкторе).**

**155. Invalid combination of opcode and operands (Недопустимая комбинация хода команды и операндов).** Код ассемблерной команды не может иметь такие операнды. Причины ошибки:

* указано слишком много или слишком мало операндов для данной команды, например, INC АХ,ВХ или MOV АХ;
* количество операндов правильное, но их типы или порядок следования не соответствуют данной команде, например, DEC 1, MOV AX, CL или MОV 1,АХ.

**156. Memory reference expected (Отсутствуетадрес).**Операнд ассемблерной команды не содержит адрес. Вероятно, Вы забыли выделить квадратными скобками индексный регистр.

**157. Cannot add or subtract relocatable symbols (Нельзя складывать или вычитать перемещаемые символы).** В ассемблерных выражениях обоими операндами в операциях сложения или вычитания могут быть только ссылки на константу. Идентификаторы переменных, процедур, функций и меток являются переметаемыми символами и не могут использоваться в качестве двух операндов одновременно в таких операциях. Если VAR -идентификатор переменной, a CONST - константы, то предложения MOV АХ, CONST+CONST и MOVE AХ, VAR+CONST будут правильными, в то время как выражение MOV AX, VAR+VAR Недопустимо.

**158. Invalid register combination (Недопустимая комбинация регистров) .** Допустимыми комбинациями индексных регистров являются [ВХ] , [ВР] , [SI], [DI] , [BX+SI], [BX+DI], [BP+SI] и [BP+DI] . Другие комбинации (например, [AХ] , [ВР+BХ] , [SI + DX]) недопустимы. Заметим, что локальные переменные размещаются на стеке и доступ к ним организуется через регистр BP. Ассемблер автоматически добавляет [ВР] в ссылках на такие переменные, поэтому в операндах типа LOCAL [BX], где LOCAL - локальная переменная, образуется недопустимая ссылка LOCAL[BP+BX] .

**159. 286/287 instructions are not enabled (Недоступен набор команд микропроцессоров 286/287).** Используйте директиву {SG+}, но учтите, что такую программу можно исполнять только на ПК, оснащенных микропроцессором Intel 80286/80287 или более поздним.

**160. Invalid symbol reference (Недопустимая ссылка на символ).** К указанному символу нельзя обратиться в ассемблерной программе. Это может быть следствием таких причин:

* Вы пытаетесь использовать ссылки на стандартные процедуры (функции) или специальные массивы MEM, MEMW, MEML, PORT я PORTW в ассемблерных операндах;
* Вы обращаетесь к строковой, вещественной константе или константе типа SET в операнде ассемблерной команды;
* в ассемблерном операнде Вы обращаетесь к процедуре или функции, написанной я режиме INLINE;
* Вы пытаетесь получить с помощью операции @Result доступ к результату, возвращаемому функцией;
* Вы пытаетесь использовать короткую форму команды JMP для передачи управления в непомеченный оператор программы.

**161. Code generation error (Ошибка генерации кода).**

**162. ASM expected (Отсутствует зарезервированное слово ASM).**

**ОШИБКИ, ВОЗНИКАЮЩИЕ**

**ВО ВРЕМЯ ВЫПОЛНЕНИЯ ПРОГРАММ**

Некоторые ошибки, обнаруженные но время выполнения программы, приводят к появлению на экране сообщения вида **Runtime error nnn at хххх: уууу (Ошибка периода исполнения nnn по адресу хххх: уууу)**, где nnn - номер ошибки; хххх:уууу - адрес (сегмент и смещение). После этого сообщения программа завершает свою работу.

Ошибки периода исполнения делятся на четыре категории: ошибки, обнаруживаемые ДОС (коды ошибок 1 до 99), ошибки ввода-ввывода (с 100 по 149), критические ошибки (с 150 по 199) и фатальные ошибки (коды ошибок с 200 до 255).

**ОШИБКИ, ОБНАРУЖИВАЕМЫЕ ДОС**

**1. Invalid function number (Неверный номер функции).** Вы пытаетесь вызвать несуществующую функцию ДОС.

**2. File not found (Fie найденфайл).** Ошибка генерируется процедурами RESET, APPEND, RENAME или ERASE в случае, если имя, присвоенное файловой переменной, указывает несуществующий файл.

**3. Path not found (Путь не найден).** Ошибка генерируется процедурами:

* RESET, REWRITE, APPEND или ERASE в случае, если имя, присвоенное файловой переменной, является недействительным иди указывает на несуществующий подкаталог;
* CHDIR, MKDIR или RMDIR в случае, если путь является недействительным или указывает несуществующий подкаталог.

**4. Too many open files (Слишком много открытых файлов).** Ошибка генерируется процедурами RESET, REWRITE или APPEND в случае, если программа имеет слишком много открытых файлов, ДОС не позволяет использовать более 15 открытых файлов для каждого процесса. Если ошибка возникла при наличии менее 15 открытых файлов, то она может указывать на то, что файл CONFIG.SYS не содержит параметра FILES=xxx или что этот параметр задает слишком мала файлов. Увеличьте параметр FILES=xxx до какого-либо подходящего значения, например, до 80.

**5. File access defined (Отказано в доступе к файлу).** Данная ошибка генерируется процедурами:

* RESET или APPEND в случае, когда имя. присвоенное файловой переменной, указывает каталог или файл, доступный только для чтения, в то время как параметр F1LEMODE файловой переменной содержит указание на запись данных;
* REWRITE и случае, когда каталог заполнен, или если имя, присвоенное файловой переменной, задает каталог или существующий файл, доступный только для чтения;
* RENAME в случае, если имя, присвоенное файловой переменной, указывает каталог или если новое имя указывает существующий файл;
* ERASE в случае, если имя, присвоенное файловой переменной, указывает каталог или файл, доступный только для чтения;
* MKDIR в случае, если файл с тем же именем уже существует в порождающем каталоге, если нет места в порождающем каталоге или если путь к каталогу содержит имя логического устройства;
* RMDIR в случае, если каталог не является пустым, если путь не определяет каталог или если пул залает корневой каталог;
* READ или BLOCKREAD в случае типизированного или нетипизированного файла, если файл не открыт для чтения;
* WRITE или BLOCKWRITE для типизированного или нетипизированного файла в случае, если этот файл не открыт для записи.

**6. Invalid filе handle (Недопустимый файловый канал).** Данная ошибка генерируется в случае, когда системному вызову ДОС передается недопустимый файловый канал. Эта ошибка не должна возникать в правильно работающей программе. Ее появление является свидетельством того, что файловая переменная каким-либо образом испорчена.

**12. Invalid file access code (Недействительный код доступа к файлам).** Ошибка генерируется процедурами RESET или APPEND, если значение параметра FILEMОDE файловой переменной не является допустимым.

**15. Invalid drive number (Недопустимый номер дисковода).** Ошибка генерируется процедурой CETDIR в случае, если номер дисковода не является допустимый.

**16. Cannot remove current directory (Нельзяудалитьтекущийкаталог).**Ошибка генерируется процедурой RMDIR в случае, если путь указывает текущий каталог.

**17. Cannot rename across drives (Нельзя при переименовании указывать разные дисководы).** Генерируется процедурой RENAME в случае, если оба файла не находятся на одном и том же диске.

**ОШИБКИ ВВОДА-ВЫВОДА**

Если один из операторов компилировался с директивой {$I+}, то ошибка ввода-вывода приводит к прекращению выполнения программы. В состоянии {$I-} программа продолжает выполняться, а ошибка возвращается функцией IORESULT.

**100. Disk read error (Ошибка чтения с виска).** Генерируется процедурой READ в типизированном файле в случае, если Вы пытаетесь осуществить считывание из полностью прочитанного файла.

**101. Disk write error (Ошибка записи на диск).** Ошибка генерируется процедурами CLOSE, WRITE, WRITELN, FLUSH в случае, если диск заполнен.

**102. File not assigned (Файлунеприсвоеноимя).**Ошибка генерируется процедурами RESET, REWRITE, APPEND, RENAME и ERASE в случае, если файловой переменной не было присвоено имя файла с помощью обращения к процедуре ASSIGN.

**103. File not open (Файлнеоткрыт).**Ошибкагенерируетсяпроцедурами CLOSE, READ, WHITE, SEEK, EOF, FILEPOS, F1LESIZE, FLUSH, BLOCKHEAD, BLOCKWR1TE вслучае, еслифайлнеоткрыт.

**104. File not open for input (Файлнеоткрытдляввода).** Ошибка генерируется процедурами READ, READLN, EOF, EOLN, SEEKEOF или SEEKEOLN в текстовом файле в случае, если файл не открыт для ввода.

**105. File not open for output (Файл не открыт для вывода).** Ошибка генерируется процедурами WRITE или WR1TELN в текстовом файле в случае, если файл не открыт для вывода.

**106. Invalid numeric format (Неверный числовой формат).** Генерируется процедурами READ или READLN в случае, если числовое значение, считанное из текстового файла, не соответствует правильному числовому формату.

**КРИТИЧЕСКИЕ ОШИБКИ**

**150. Disk is write protected (Диск защищен от записи).**

**151. Unknown unit (Неизвестный модуль).**

**152. Drive not ready (Дисковод находится в состоянии “не готов”).**

**153. Unknown command (Неопознанная команда).**

**154. CRC error in data (Ошибка в исходных данных).**

**155. Bad drive requiest structure length (При обращении к диску указана неверная длина структуры).**

**156. Disk seek error (Ошибка при операции установки головок на диске).**

**157. Unknown media type (Неизвестный тип носителя).**

**158. Sector not found (Сектор не найден).**

**159. Printer out of paper (Кончилась бумага на принтере).**

**160. Device write fault (Ошибка при записи на устройство).**

**161. Device read fault (Ошибка при чтении с устройства).**

**162. Hardware failure (Сбой аппаратуры).**

**ФАТАЛЬНЫЕ ОШИБКИ**

Эти ошибки всегда приводят к немедленной остановке программы.

**200. Division by zero (Деление на ноль).**

**201. Range check error (Ошибка при проверке границ).** Ошибка генерируется операторами, скомпилированными в состоянии {$R+}, при возникновении одной из следующих ситуаций:

* индексное выражение массива находилось вне допустимого диапазона;
* была осуществлена попытка присвоить переменной значение, находящееся вне диапазона переменной;
* была осуществлена попытка передать значение, находящееся вне допустимого диапазона, а качестве параметра процедуре или функции.

**202. Stack overflow error (Переполнение стека)** Эта ошибка генерируется на входе в процедуру или функцию, скомпилированную в режиме {$S+}, случае, если нет достаточной области для размещения локальных переменных подпрограммы. Увеличьте размер стека, используя директиву компилятора {$М}

**203. Heap overflow error (Переполнение кучи).** Эта ошибка генерируется процедурами NEW или GETMEM в случае, если в куче нет свободной памяти требуемого размера.

**204. Invalid pointer operation (Недействительная операция с указателем).** Эта ошибка генерируется процедурами DISPOSE или FREEMEM в случае, когда указатель имеет значение NIL или указывает на адрес, лежащий за пределами динамически распределяемой области памяти.

**205. Floating point overflow (Переполнение при операции с плавающей запятой).** В результате выполнения операции с плавающей запятой получено слишком большое вещественное число.

**206. Floating point underflow (Исчезновение порядка при операции с плавающей запятой).** Эта ошибка генерируется только в том случае, если используется сопроцессор 8087/80287/80387 с управляющим словом, которое демаскирует ошибку исчезновения порядка. По умолчанию исчезновение порядка приводит к возвращению результата, равного нулю.

**207. Invalid floating point operation (Недопустимая операция с плавающей запятой).** Возможные причины сообщения:

* аргумент функций TRUNC или ROUND не может быть преобразован в целое число, находящееся внутри диапазона типа LONGINT (от -2147483648 до -12147483647);
* отрицательный аргумент функции SQRT (извлечение квадратного корня);
* аргумент функции LN (логарифм) равен нулю или имеет отрицательное значение;
* произошло переполнение стека сопроцессора.

**208. Overlay manager not installed (He установлена подсистема управления оверлеем).** Ваша программа вызывает оверлейную процедуру или функцию, а подсистема управления оверлеем не инициирована. Вероятнее всего, в программе отсутствует обращение к процедуре OVRINIT или обращение к этой процедуре завершилось с ошибкой. Нужно иметь в виду, что если в каком-либо из оверлейных модулей содержится раздел инициации, то в программе необходимо создать дополнительный или использовать имеющийся неоверлейный модуль, вызывающий процедуру OVRINIT в своем разделе инициализации, и указать этот модуль в предложении USES перед любым из оверлейных модулей.

**209. Overlay file read error (Ошибка чтения оверлейного файла).** Ошибка чтения произошла, когда подсистема управления оверлеем пыталась считать оверлейный модуль из оверлейного файла.

**210. Object not initialized (He инициированобъект).**Вы обращаетесь к виртуальному правилу применительно к неинициированному объекту (до вызова конструктора).

**211. Call to abstract method (Вызовабстрактногоправила).** Эта ошибка генерируется правилом ABSTRACT модуля OBJECT при работе в среде Turbo Vision в случае обращения к абстрактному правилу, т.е. к виртуальному правилу, которое разработано специально для его замены в объектах-потомках.

**212. Stream registration errata (Ошибка в регистрируемом потоке).** Эта ошибка оперируется правилом REGISTERTYPE модуля OBJECT при работе в среде Turbo Vision в случае возникновения одной из следующих ошибок:

* запись регистрируемого потока не содержит сегмента данных;
* поле OBJTYPE записи регистрируемого потока нулевое;
* указанный гип уже был зарегистрирован;
* существует другой тип с таким же полем OBJTYPE.

**213. Collection index out of range (Набираемый индекс выходит из границ диапазона).** Индекс, передаваемый объекту TCOLLECTION при работе в среде Turbo Vision, выходит за границы диапазона.

**214. Collection overflow error (Переполнение коллекции).** Эта ошибка генерируется объектом TCOLLECTION при работе в среде Turbo Vision в случае, если делается попытка добавить элемент к коллекции, которую нельзя расширять.

**Литература**

***Основная***

1. Бобровский, С., Delphi 5 [Текст]: учебный курс / С. Бобровский. - "[Питер](http://www.colibri.ru/pubinfo.asp?cod=1436)", 2009. – с. 640.
2. Глинский, Я.Н., Turbo Pascal 7.0 и Delphi [Текст] / Я.Н. Глинский и др. - "ДиаСофт", 2010. – 208 с.
3. Голицына, О.Л., Основы алгоритмизации и программирования [Текст]: Учебное пособие / О.Л. Голицына, И.И. Попов – М.: Форум: Инфра-М, 2010.
4. Гофман, В.Э. Delphi 6. В подлиннике [Текст] / В.Э. Гофман, и др. – "[BHV-Санкт-Петербург](http://www.colibri.ru/pubinfo.asp?cod=12)", 2007. – 1152 с.
5. Грызлов, В.И., Турбо Паскаль 7.0. [Текст] / В.И. Грызлов, Т.П. Грызлова – М.: ДМК, 2008.
6. Климова, Л.М. Pascal 7.0. Практическое программирование. [Текст]: Решение типовых задач. 3-е изд., доп. / Л.М. Климова – "[КУДИЦ-Образ](http://www.colibri.ru/pubinfo.asp?cod=931)", 2008. – 528 с.
7. Культин, Н. Turbo Pascal в задачах и примерах [Текст] / Н. Культин. – "[BHV-Санкт-Петербург](http://www.colibri.ru/pubinfo.asp?cod=12)", 2010. – 256 с.
8. Меняев, М.Ф.Информатика и основы программирования[Текст]/ М. Меняев – « Омега-Л», 2007
9. Немнюгин, С.Н. Pascal [Текст]: учебник / С.Н. Немнюгин – "[Питер](http://www.colibri.ru/pubinfo.asp?cod=1436)", 2001. – 496 с.
10. Фаронов, В.В. Turbo Pascal 7.0 [Текст]: практика программирования. 7-е издание / В.В. Фаронов – "[Нолидж](http://www.colibri.ru/pubinfo.asp?cod=1324)", 2010
11. Фаронов, В.В. Delphi 6. [Текст]: учебный курс / В.В. Фаронов - "[Нолидж](http://www.colibri.ru/pubinfo.asp?cod=1324)", 2010. – 608 с.
12. Фаронов, В.В. Программирование баз данных в Delphi 7. [Текст]: учебный курс / В.В. Фаронов –«Питер», 2006- 728с.
13. Фаронов, В.В. Искусство создания компонентов Delphi [Текст]: учебный курс / В.В. Фаронов –«Питер», 2005.

***Дополнительная***

1. Альфред, В. А, Моника С. Лам, Рави Сети, Джеффри Д. Ульман Компиляторы. Принципы, технологии и инструментарий [Текст]: учебный курс /В.А. Альфред, Моника С. Лам, Рави Сети, Джеффри Д. Ульман- «Вильямс», 2008 – 786 с.
2. Зеленяк, О.П. Решение задач по планиметрии. Технология алгоритмического подхода на основе задач-теорем. Моделирование в среде Turbo Pascal [Текст]:\ О.П. Зеленяк – « ДиаСофтЮП, ДМК пресс», 2008 – 346 с.
3. Тейксейра, С., Delphi 5. [Текст]: руководство разработчика. Т. 1. Основные методы и технологии программирования. / С. Тейксейра, К. Пачеко – «Вильямс», 2000. – 832 с.
4. Тейксейра, С., Delphi 5. [Текст]: руководство разработчика Т. 2. Разработка компонентов и программирование баз данных + (CD) / С. Тейксейра, К. Пачеко – "Вильямс", 2000. – 992 с.
5. Фленов, М Библия Delphi, 2-ое издание[Текст]: руководство разработчика\М. Фленов-« БХВ-Петербург», 2008- 658 с.

***Интернет-источники:***

1. [Электронный ресурс] http://www.codenet.ru
2. [Электронный ресурс ]<http://www.chemisk.narod.ru/html/algoritm01.html>
3. Хант, Э Томас, Д Программист-прагматик. Путь от подмастерья к мастеру [Текст]: руководство разработчика \ Э. Хант, Д. Томас « Лори, Питер Пресс», 2007
4. Хомоненко , А.Самоучитель Delphi .NET[Текст]: руководство разработчика./А. Хоменко- «БХВ-Петербург», 2006-604 с.

Учебное издание

Татьяна Николаевна Коренькова

Елена Николаевна Запасник

ОСНОВЫ АЛГОРИТМИЗАЦИИ И ПРОГРАММИРОВАНИЯ

Учебное пособие для студентов специальности

09.02.04 – Информационные системы (по отраслям)

*Технический редактор:* Иванова Н.И.

*Компьютерная верстка:* Коренькова Т.Н.

Подписано к печати \_\_\_\_\_\_\_\_\_\_\_\_\_\_Бумага для множительной техники

Формат \_\_\_\_\_\_\_\_Усл.печ.лист.\_\_ \_\_\_\_\_\_\_Тираж\_\_\_\_\_\_\_\_экз. Заказ\_\_\_\_\_\_\_\_\_\_

Отпечатано с авторского оригинала в отделе оперативной печати

Старооскольского технологического института.

Старый Оскол, микрорайон Макаренко, 40.