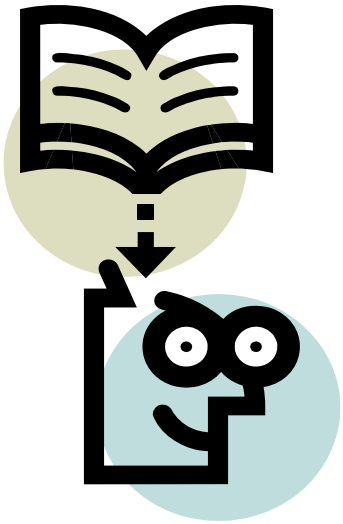
ПОСОБИЕ ДЛЯ УЧАЩИХСЯ

Основы программирования на Паскаль АВС

Составитель: Овчинников А. А.



г. Волгоград
2012 год

Рецензент:
к.т.н., ст. преподаватель кафедры «Программное обеспечение автоматизированных систем» (ПОАС) ВолгГТУ С.А. Овчинников

Аннотация

Пособие предназначено для обучения основам программирования школьников на начальном этапе. Необходимость такого пособия вызвана тем, что в базовых учебниках для средней школы раздел программирования либо вообще не представлен, либо представлен недостаточно.

В то время, как ученики нуждаются в пособии, по которому можно подготовить домашнее задание, изучить пропущенный материал или поработать дополнительно. Тем более, что пакет Паскаль АВС имеет в своей базовой комплектации «Электронный задачник», что значительно расширяет возможности пособия, так как после каждой темы даются упражнения по этому задачнику.

Надеюсь, что данное пособие может оказаться полезным и учителям информатики.

В 1970 г. профессор Никлаус Вирт из Швейцарии обосновал и разработал язык высокого уровня – Паскаль. Этот язык отличается простотой и стройностью, качествами, которые обеспечивают Паскалю популярность уже на протяжении нескольких десятилетий.

В настоящее время удобной в учебном процессе является система программирования Pascal ABC (Паскаль АВС). Система предназначена для обучения программированию на языке Паскаль и ориентирована на школьников и студентов младших курсов. Кроме того, в пакете имеется Электронный задачник.

Эта система призвана осуществить переход от простейших программ к модульному, объектно-ориентированному, событийному и компонентному программированию.

Введение. Установка Паскаль АВС

Все авторские права программного комплекса Pascal ABC 3.0 & Programming Taskbook 4.5 Mini Edition (называемого в дальнейшем системой PABC-PT ME) принадлежат только авторам: С.С.Михалковичу и М.Э.Абрамяну.

Система PABC-PT ME является бесплатной и распространяется свободно при условии, что настоящий дистрибутив не изменен. Ни одно частное лицо или организация не может брать плату за распространение системы PABC-PT ME.

Скачать программный комплекс можно в интернете совершенно бесплатно и установить на свой компьютер.
В скачанном пакете для установки запустите файл PABCInstall и в появившемся окне нажмите кнопку "Установка".

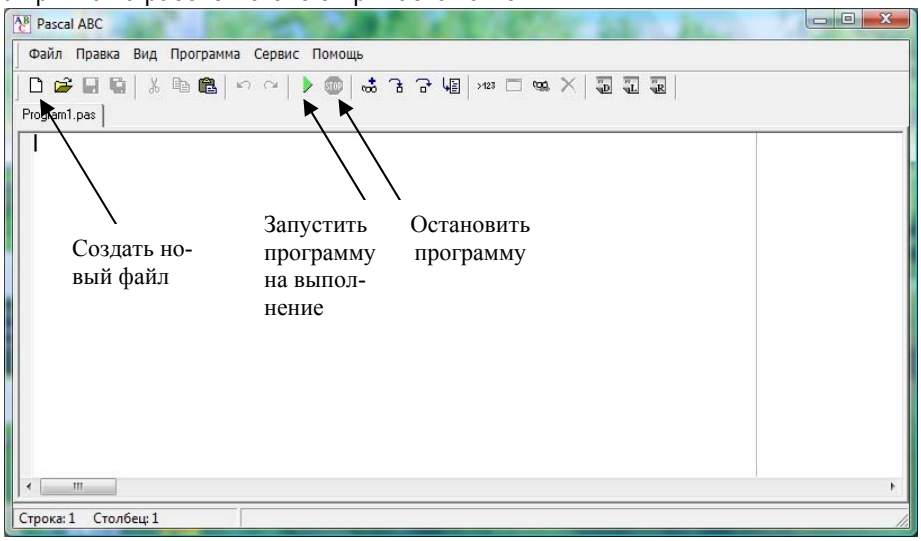
После установки автоматически запускается программа регистрации и настройки PABCSetup. В дальнейшем эту программу можно запустить повторно с помощью команды "PABC Setup - Регистрация и настройка" пункта "Pascal ABC" в группе "Программы" главного меню Windows.

ТЕМА 1. Знакомство с системой программирования
Паскаль АВС

ВНИМАНИЕ! Для каждого пользователя рекомендуется создать отдельную папку, например, KURS, а в ней папки для файлов.

Для запуска Паскаль АВС необходимо запустить ярлык Pascal ABC. На экране появится среда программирования Паскаль АВС (оболочка). Среда программирования – это пакет взаимосвязанных файлов, которые позволяют набирать, редактировать, запускать и отлаживать программы.

После запуска ярлыка на рабочем столе открывается окно:



Первая строка экрана – меню интегрированной среды, следующая строка – панель инструментов, нижняя строка экрана – строка подсказки и состояния интегрированной среды. Между ними расположено окно редактирования – рабочее поле, в котором можно открывать несколько вкладок для разных программ.

Окно редактирования предназначено для ввода и редактирования текста программы. Место ввода информации обозначено курсором. В верхней левой части окна редактирования выводится служебное имя редактируемого файла, например: *Program1.pas*

1. Найдите строку Меню (сверху) и строку-подсказку (снизу).
2. Поочередно войдите в указанные ниже разделы Меню (активизируйте Меню мышью).
3. Найдите следующие команды:

В меню *Файл*

Новый – создать новый фал

Открыть – открыть файл

Сохранить – сохранить файл
Сохранить как… – сохранить под новым именем
Выход – выйти из Паскаля

В меню *Правка*
Отменить – отменить изменение
Восстановить – вернуть изменение

В меню *Программа*
Выполнить – выполнить программу
Остановить – остановить программу.

Первые шаги

Наберем простейшую программу, соответствующую условию задачи:
Ввести в компьютер два целых числа, найти их сумму, результат вывести на экран с поясняющим текстом.
Внимание! Две косые черты (//) отделяют комментарии, их набирать не нужно.

```
program raschet;                    // название программы
uses crt;                           // подключаемые модули
var x, y, s:integer;                // объявление имен переменных и их типа
begin                               // начало исполнительной части
writeln('Введите два целых числа'); // написать на экране текст
readln(x,y);                        //  прочитать данные с клавиатуры и
                                    //  запомнить их в переменных
s:=x+y;                             //  выполнить расчет и запомнить его в
                                    //  переменной
writeln('Сумма чисел =',s);         // написать на экране текст и значение
                                    //  переменной
end.                                // конец программы
```

4. Просмотрите текст файла, обратите внимание на стуктуру программы.

Структура простейших программ выглядит следующим образом:

```
program    ...;     заголовок программы и ее имя
  var      ...;       блок объявления переменных и их типа
    begin               начало исполнительной части программы
    ...;                  предложения, обеспечивающие
    ...;                  выполнение
    ...;                  программы
    end.                конец программы (точка обязательна)
```

Программа на Паскале составляется из отдельных законченных элементов, называемых предложениями. В Паскале текст программы обычно начинается особым предложением – заголовком следующего вида:
program proba;
где *proba* – имя текущей программы.
Заголовок необязателен.
В качестве имени программы можно применять комбинацию английских букв и цифр, следует писать в одно слово и нельзя применять служебные слова языка.

Каждое предложение языка должно отделяться от следующего за ним точкой с запятой (**;**).
Исключение составляют комментарии. Они не отделяются точкой с запятой.

Обычно каждое предложение записывается с новой строки для наглядности и более легкого понимания текста. Для этих же целей используют отступы и выравнивания.

Комментарии предназначены для пояснения задачи и для временного исключения из текста программы некоторых операторов. В тексте они выделяются фигурными скобками { } или отделяются двумя косыми чертами //. Комментарии игнорируются компьютером при выполнении, однако при выводе текста программы – печатаются.

В Паскале имеется особая группа слов, таких как, например: *begin, for, end, program* и другие, за которыми закреплены специальные смысловые значения. Такие слова называются служебными (зарезервированными) и должны употребляться в строгом соответствии с заложенным в них смыслом.

Существует и другая группа имен, имеющих стандартно определенный смысл, например, *integer, writeln* и другие. Их так и называют – стандартные или предопределенные имена.

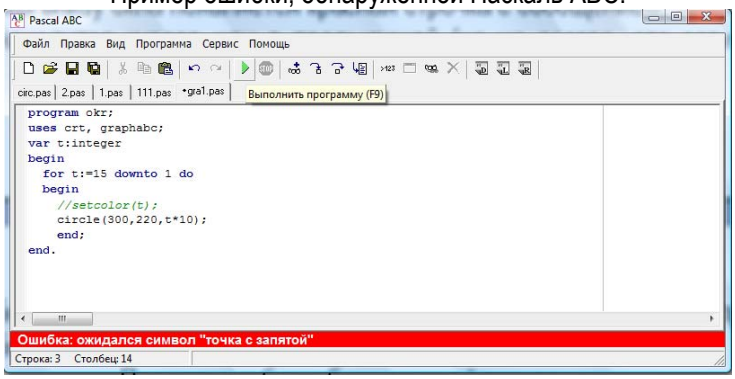
Под именем программы располагается ее *декларативная* часть, здесь компьютеру сообщается обо всех именах *констант* и *переменных*, определяемых программистом, и о той роли, которую эти имена должны исполнять в программе.

За *декларативной* частью следует *исполнительная* часть программы, обрамляемая словами-ограничителями (*логическими скобками*): *begin* и *end*. Между указанной парой слов и размещаются предложения, выполняющие в программе

те или иные действия. *Исполнительную* часть программы называют *телом программы*.

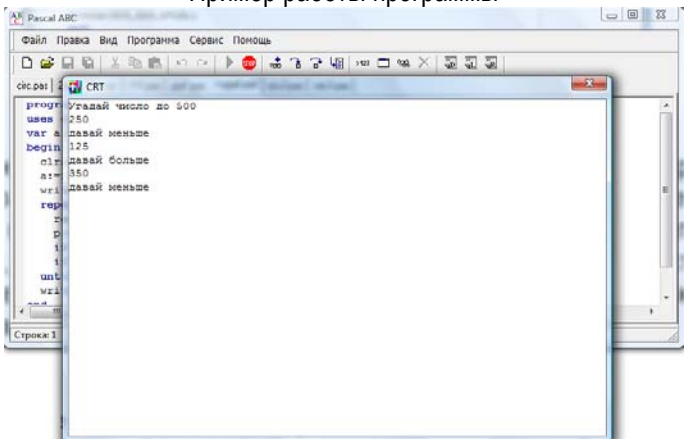
5. Запустите набранную программу на выполнение. Если после запуска программы внизу окна появляется красная строчка с сообщением (рисунок см. ниже), то в строке, где находится курсор или в предыдущей (но не всегда, это зависит от ошибки!), внимательно просмотрите всю строчку, найдите и исправьте ошибку. Если сами не справитесь, позовите учителя.

Пример ошибки, обнаруженной Паскаль АВС:



6. После исправления всех ошибок и появления в новом окне начала работы программы, введите нужные данные (если в программе подразумевается ввод нескольких переменных, то *это следует делать через Enter или пробел!*), получите результат работы и проверьте его на правильность. Так как текст программы и ее работа показываются в разных окнах (если подключен модуль *Crt*), можно сопоставить программные строки и ее выполнение.

Пример работы программы



7. Сохраните набранную программу в своей папке.
8. Разберитесь с работой программы и измените ее так, чтобы она вычисляла не сумму, а разность чисел. Проверьте правильность работы измененной программы. Сохраните программу под новым именем в своей папке.

Набор следующей программы.

9. Активизируйте пункт *Файл* и создайте новый файл (*Новый*).
10. Наберите текст программы (см. ниже). При наборе текста программы соблюдайте позиционирование (отступы) строк. Это не влияет на работу программы, но делает ее читабельной и облегчает поиск ошибок.
11. В следующей программе подсчитывается доход клиента за 1 год в зависимости от банковского процента и от величины денежного вклада.

Внимание! Текст в фигурных скобках является пояснением: его не нужно набирать. Обратите внимание на значение служебных слов языка.

```
program doxod;                    {название программы}
uses crt;                         {подключаемые модули (библиотеки)}
var b,a:integer;                  {объявление переменных и их типа}
    c:real;
begin                             {начало программы}
  clrscr;                         {очистка экрана}
  writeln('Доход от вклада');     {вывод текста на экран
                                   с переводом курсора на следующую строку}
  write( 'Введите величину вклада в рублях: ' ); {вывод текста на экран без перевода
                                                  курсора на следующую строку}
  readln(b);                                     {ввод целого числа в
```

```
                                        переменную b с переходом на
                                        следующую строку}
  write('Введите величину банковского процента  ');
  readln(a);
  c:=a*b/100;                          {расчет значения переменной c}
  writeln('Ваш доход =',c,' рублей');   {вывод текста, значения переменной и текста}
end.
```

12. Запустите программу на выполнение. Введите следующие данные:
    Введите величину вклада в рублях: 1000
    Введите величину банковского процента. 10

В результате должен получиться ответ:

Ваш доход =100 рублей

13. Снова запустите программу и введите другие разумные исходные данные.

Привыкайте понимать сообщения об ошибках!

14. Вернитесь в текст, сотрите знак " **;** " в любом месте программы и запустите ее на выполнение. Проанализируйте сообщение об ошибке (красная строчка с сообщением).
15. Исправьте ошибку, затем сотрите точку после последнего *End* в программе. Эта ошибка часто встречается у начинающих. Запустите программу и посмотрите, как реагирует Паскаль на подобную ошибку.
16. Сотрите любую букву, например, в слове *writeln*. Посмотрите, как реагирует Паскаль на подобную ошибку.
17. Сотрите в блоке *var* объявленную переменную и посмотрите, как отреагирует компьютер на запуск программы с такой ошибкой. Запоминайте сообщения компьютера.
18. Исправьте ваши ошибки и сохраните программу.
19. Напишем программу, соответствующую следующей задаче:

Запросить имя пользователя и его возраст. Определить год рождения (текущий год запросить с клавиатуры), вывести его на экран и попрощаться по имени.

```
program vozrast;
uses crt;
var voz, gr, tg:integer;      {для возраста, года рождения, текущего года: целые числа}
    im:string;                {для имени: буквы, слова}
begin
  clrscr;
  write( 'Как тебя зовут? ' );
  readln(im);
  write('Сколько тебе лет?  ');
  readln(voz);
  write('Какой сейчас год?  ');
  readln(tg);
  gr:=tg-voz;
  writeln('Ты родился в  ',gr,' году');
  writeln('До свидания,  ',im);
end.
```

Запомните:

тип integer для хранения целых чисел
тип real для хранения любых чисел, в том числе дробных
тип string для хранения слов и букв
clrscr очистить экран
write написать на экране и оставить курсор в текущей строке
writeln написать на экране и перейти на следующую строку
readln прочитать данные с клавиатуры и поместить их в переменную

Упражнения

Выполните задания с *Begin1°* по *Begin8°*, с *Begin31°* по *Begin34°* по электронному задачнику (*стр.11*). Открыть его можно в режиме *Помощь – Электронный задачник РТ*. Задачник откроется отдельным файлом в формате *pdf*.

ТЕМА 2. Типы переменных. Простые числовые операции и функции

Операторы ввода-вывода данных.

В Паскале используется несколько типов представления числовых значений, на начальном этапе будут рассмотрены лишь некоторые из них:

integer – целые числа в интервале от -2147483648 до 2147483647

real – вещественные (реальные) – целые и дробные положительные и отрицательные числа

Описания *констант* в декларативной части производится перед переменными, и предусматривают определенную форму записи чисел (дополнительно тип константы не оговаривается): если константа записана с точкой, тип константы считается real. При записи значения константы используется знак равенства.

Пример описания констант:

```
const
  c1=3.14159265;          // c1 имеет тип real
  c2=2;                   // c2 имеет тип integer
```

Переменная – это вид объектов в программе, предназначенный для хранения информации во время выполнения программы. По правилам Паскаля каждая переменная должна быть объявлена, т.е. описана в декларативной части программы.

Переменная не имеет какого-либо конкретного значения до тех пор, пока компьютеру не будет дано точное предписание, поместить что-либо определенное в соответствующую ячейку памяти.

На Паскале такого рода предписание обычно выражается предложением назначения, имеющим вид:

имя_переменной:=выражение_или_значение

например: a:=25; или rt:=a+b;

Выполнение такого предложения начинается вычислением выражения, стоящего справа от символа назначения (:=). Полученное значение потом помещается в переменную (присваивается ей), указанную слева от =:

Описание переменных следует за описанием констант. В описании переменных после двоеточия указывается тип переменной:

```
var a,d,c : integer;
    b,f : real;
```

В Паскале возможны следующие действия (группы операций записаны в порядке приоритета):

1) Операция возведения в степень (в стандартном Паскале отсутствует) – функция i:=power(x,y), где i - результат, x - основание, y - степень;
2) умножение (*), деление (/), деление целочисленное (div), получение остатка от целочисленного деления (mod);
3) сложение (+), вычитание (-).

В пределах одной группы приоритета порядок выполнения операций, если нет скобок, определяется последовательностью записи.

Если хотя бы одна из переменных, используемых в операциях умножения, сложения, вычитания относится к типу real, результат операции будет типа real.

Результат операции деления всегда типа real. Результат операций div и mod - integer.

Пример записи математической операции:

$$y=(a:b+c^2-d)*c$$

в программе на Паскале это будет выглядеть так:

*y:=(a/b+sqr(c)-d)*c;*

Примечание. Квадрат числа *sqr(c)* можно (и проще!) записать как *c*c*

Обратите внимание на использование знака операции присваивания :=

При проведении математических операций следует учитывать, что вещественные числа (real) представляются с некоторым приближением.

Некоторые стандартные математические функции

При использовании стандартных функций необходимо контролировать тип аргумента (возможные типы указаны в скобках).

a:=random(x) – случайное число (если аргумент не указан, то результат real – число в интервале от 0 до 1, если x целое число от 0 до 65535, то результат случайное целое число в интервале от 0 до x-1;
a:=abs(x) – модуль (абсолютная величина) x (real или integer);
a:=int(x) – целая часть числа x (число real, результат integer), округление не проводится, дробная часть отсекается;
a:=frac(x) – дробная часть числа x (число и результат real),
a:=round(x) – целое число, полученное в результате округления числа x по правилам математики
a:=sqr(x) – квадрат числа x (real, integer);
a:=sqrt(x) – квадратный корень из числа x (real, x>0);
a:=sin(x) – синус x (x задается в радианах, real);
a:=cos(x) – косинус x (x задается в радианах, real);
a:=arctan(x) – арктангенс x (real);
a:=power(x,y) – степень числа
pi – число пи.

Наберите следующую программу и разберитесь в ее работе:

Вычислить сумму цифр трехзначного числа *a*, введенного с клавиатуры. В программе каждая цифра определяется как количество сотен, десятков и единиц с помощью арифметических операций.

```
program summa;
uses crt;
  var  a, s, d, e, o : integer;
begin
 writeln('Сумма цифр трехзначного числа');
 write('Введите целое трехзначное число  ');
 readln(a);
 clrscr;
                        {первый способ}
 s:= trunc(a/100);                    {количество сотен}
 d:= trunc((a-s*100)/10);             {количество десятков}
 e:=a-s*100-d*10;                     {количество единиц}
 writeln('Сумма цифр трехзначного числа=', s+d+e);

                        {второй способ}
 s:=a div 100;                        {количество сотен}
 o:=a mod 100;
 d:=o div 10;                         {количество десятков}
 e:=a-s*100-d*10;                     {количество единиц}
 writeln('Сумма цифр трехзначного числа=', s+d+e);
end.
```

Упражнения.

Выполните задания с *Begin9°* по *Begin30°* по электронному задачнику (*стр.11*). Открыть его можно в режиме *Помощь – Электронный задачник РТ*.

Задачи для самостоятельного решения

1. Написать программу вычисления расстояния между двумя точками с координатами x1, y1, x2, y2. Использовать теорему Пифагора. Результат вывести на экран с поясняющим текстом.
2. Составить программу, чтобы компьютер запросил имя пользователя и его год рождения, затем подсчитал количество лет, дней и минут, прожитых, примерно, этим человеком. Результаты вывести на экран.
3. Составить программу, чтобы компьютер по закону Ома для участка цепи, запрашивая с клавиатуры значения напряжения на концах участка и его сопротивления, определял и выводил на экран значение силы тока (I=U/R).
4. Составить программу для того, чтобы компьютер, используя генератор случайных чисел, записал на экране случайное число, значение которого лежит в пределах от 0 до 1.
5. Составить программу для того, чтобы компьютер, используя генератор случайных чисел, записал на экране случайное число, значение которого лежит в пределах от 5 до 6.
6. Составить программу для того, чтобы компьютер, используя генератор случайных чисел, записал на экране случайное число, значение которого лежит в пределах от 5 до 10.
7. Составить программу для того, чтобы компьютер, используя генератор случайных чисел, записал на экране случайное число, значение которого лежит в пределах от 5 до 6 и имеет 2 знака в дробной части.

ТЕМА 3. Условный оператор (ветвление)

Условный оператор If в зависимости от значения *некоторого условия* выполняет либо оператор, стоящий после *Then* (условие выполняется), либо оператор, стоящий после *Else* (условие не выполняется).

Структура условного оператора:

If условие *Then* группа операторов [*Else* группа операторов];

Перед *Else* знак " **;** " никогда не ставится. В квадратных скобках указана необязательная часть.

В качестве условия может быть использовано любое выражение логического типа.

Условия можно объединять с помощью *Or – Или* и с помощью *And – И*. Если условий несколько, то каждое из них необходимо заключить в скобки:

```
If  (a>2) and (b<2) Then
```

Наберите и отладьте программу treug1

Программа должна запросить три стороны треугольника и по ним определить тип треугольника. Определение типа производится сравнением сторон. После отладки проверьте ее при работе со сторонами

а) 2, 2, 2; ответ должен быть "равносторонний"
б) 4, 4, 8; ответ должен быть "равнобедренный"
в) 4, 3, 5 ответ должен быть "разносторонний".

```
program treug1;
uses crt;
var a,b,c:integer;
    d:string;
begin
 clrscr;
 write('Введи три стороны треугольника ');
 readln (a,b,c);
 if (a=b) or (b=c) or (a=c) then  d:='треугольник равнобедренный';
 if (a=b) and (b=c) then d:='треугольник равносторонний';
 if (a<>c) and (a<>b) and (b<>c) then  d:='треугольник разносторонний';
 writeln (d);
end.
```

Счетчики

Для подсчета любых данных, ситуаций, событий, удобно использовать *счетчики*.
Счетчиком можно назвать расчетную строку, в которой слева и справа используется одинаковая переменная.
Например, k:=k+… или m:=m*… или d:=d-… и т.д., где вместо многоточия (…) записывается число или переменная.
Строку типа k:=k+1 можно назвать *счетчиком количества*. Работает она так: *число, лежащее в ячейке k увеличить на 1 и снова записать в ту же ячейку.*
Вот фрагмент программы, показывающий работу такого счетчика:

```
readln(a,b,c);
if a>0 then p:=p+1 else o:=o+1;
if b>0 then p:=p+1 else o:=o+1;
if c>0 then p:=p+1 else o:=o+1;
writeln('положительных чисел - ',p, ' отрицательных чисел или 0 - ',o );
```

так же часто используется счетчик суммы: *s:=s+a*
Применение счетчиков такого типа будет рассмотрено позднее, в работе циклов.

Работа с символьными переменными

В Паскаль АВС используется несколько типов символьных переменных, в этой теме будут рассмотрены два: *char* и *string*:
char: значения переменной – отдельный символ (один);
string: цепочка символов (несколько букв, символов, слов).
Например: var a:char; b:string;
Символьные переменные можно объединять (складывать):

```
f:='ab';
d:='cd';
s:=f+d;  или  s:='ab'+'cd';
```

результатом операции будет значение переменной s, равное 'abcd'.

В работе с символьными переменными могут использоваться операции отношения: =, <>, >, <, >=, <=, в которых проводится посимвольное сравнение кодов (номеров) символов. Если коды первых символов равны, то сравниваются коды следующих символов.

1. Наберите и отладьте программу imena (определение полного имени по короткому). После отладки проверьте ее при работе с именами *Саша, Коля и Витя*.

```
program imena;
uses crt;
var ik, ip : string;
begin
 clrscr;
 write('Введите имя ');
 readln (ik);
 ip := 'Я такого не знаю';
```

```
 if ik = 'Саша' then ip := 'Александр';
 if ik = 'Коля' then ip := 'Николай';
 if ik = 'Петя' then ip := 'Петр';
 writeln (ip);
end.
```

2. Написать программу нахождения максимальной из двух величин a и b, запрошенных с клавиатуры. Используется дополнительная переменная m, которой присваивается значение большего из чисел a и b.

```
program r1;
uses crt;
  var a, b, m:real;
begin
 writeln('Максимум двух чисел');
 write('Введи первое число  ');
 readln(a);
 write('Введи второе число  ');
 readln(b);
 if a>=b then m:=a else m:= b;
 clrscr;
 writeln('a =',a,' b =',b);
 writeln('max =',m);
end.
```

Задачи для самостоятельного решения

1. Составить программу, чтобы компьютер запросил имя пользователя и его год рождения, затем подсчитал возраст человека, в зависимости от возраста разработайте вариант диалога с пользователем (еще не учишься, учишься в таком-то классе (использовать формулу!), уже не учишься).
2. Написать программу вычисления y в зависимости от значения *x*

   y = 1/x при x < 0 y = 2*x² при других x
3. Написать программу вычисления y в зависимости от значения *x*

   y = 1/ x² при x > 0 y = x/6 при других x
4. Запросить с клавиатуры координаты точки (X, Y) и горизонтального отрезка прямой (Xn, Xk, Yn) и определить, лежит точка на отрезке прямой или нет. Сообщение об этом вывести на экран.

   *Подсказка***.** Если координата Y точки не равна координате Y прямой, то НЕ лежит, если координата X точки не находится в пределах между Xn начала и Xk конца прямой, то точка НЕ лежит на прямой.
5. Написать программу вычисления подоходного налога по формулам:

   а) при сумме менее 2500 рублей налог не взимается,

   б) от 2500 до 10000 руб, берется 13% от суммы,

   в) при сумме более 10000 руб берется 1300 рублей плюс 15% от суммы превышающей 10000 рублей.

   Указание: программу проверить при суммах 1000, 8000 и 12000 руб. Ответы должны быть соответственно 0, 1040 и 1600 руб.
6. Запросить с клавиатуры 3 стороны треугольника и по ним определить, является ли он прямоугольным, сообщение вывести на экран.

   *Подсказка***.** Для каждой стороны применить теорему Пифагора и проверить, выполняется ли она. Если выполняется, то треугольник является прямоугольным. Переменные должны быть целыми числовыми. Функция квадрата - sqr(x), корня квадратного - sqrt(x). Программу проверить при сторонах 3, 4, 5 - прямоугольный, а 4, 5, 6 - не прямоугольный.
7. Запросить радиус круга R и сторона квадрата A. Определить, поместится ли круг в квадрате. Круг поместится в квадрате, если диаметр круга меньше или равен стороне квадрата.
8. Запросить радиус круга R и сторона квадрата A. Определить, поместится ли квадрат в круге. Квадрат поместится в круге, если диагональ квадрата меньше или равна диаметру окружности.
9. Написать программу для определения подходящего возраста для вступления в брак, используя следующее соображение: возраст девушки равен половине возраста мужчины плюс 7, возраст мужчины определяется соответственно как удвоенный возраст девушки минус 14. Данные для проверки работы программы задать самостоятельно.
10. Написать программу, контролирующую знание закона Ома. Обучаемый вводит формулу закона Ома в символьную переменную, которая далее сравнивается с правильным ответом, хранящимся в другой символьной переменной.
11. Написать программу вычисления значения функции y

    y = x², если -2 <= x <= 2, y = 4 в остальных случаях.
12. Задать с помощью условного оператора следующие действия:

а) меньшее из двух значений переменных вещественного типа x и y заменить 0, а в случае их равенства заменить нулями оба;

б) наибольшее их трех различных значений переменных x, y, z уменьшить на 0.3

13. Найти сопротивление цепи из двух соединенных проводников. Сопротивления проводников и тип соединения запрашивать с клавиатуры. При последовательном соединении проводников R = R1 + R2, при параллелельном соединении проводников R = R1 * R2/(R1 + R2)

Упражнения

Выполните задания с *If1* по *If15* по электронному задачнику (*стр.20*). Открыть его можно в режиме *Помощь – Электронный задачник PT*.

ТЕМА 4. Операторы цикла

Циклы применяются для повторения какой-либо последовательности операторов несколько раз. В Паскале существуют три вида циклов:

- Цикл For (со счетчиком);
- Цикл While (с предусловием);
- Цикл Repeat (с постусловием).

Оператор цикла For

Цикл for позволяет выполнить серию действий заданное число раз.

Общая форма цикла for такова:

```
for i:=start  to  limit do
    тело цикла
или
for i:=start  downto limit  do
     тело цикла
```

где start и limit – переменные программы

В данной конструкции «i» играет роль управляющей переменной цикла или счетчика и должна быть только целого типа. Слово start – обозначает здесь начальное значение переменной a, limit – ее конечное значение. Тело цикла должно состоять из одного оператора, но можно поместить в тело цикла несколько операторов, для этого их нужно взять в логические скобки *begin – end*.

Цикл for выполняется следующим образом. Сначала производится инициализация (присваивание начального значения) *i* – его начальным значением становится *start*.

Если i меньше или равно *limit* , тело цикла выполняется (при значении *i* равном *start*). Всякий раз, когда тело цикла завершается, значение *i* автоматически увеличивается на 1, и тело цикла выполняется вновь, но уже с новым (следующим по порядку) значением *i*.

Циклические повторения тела будут продолжаться до тех пор, пока не будет превзойдено конечное значение. Когда это случится, цикл завершится, и будет выполняться строка программы, непосредственно следующая за конструкцией For.

Если в теле цикла находится несколько операторов, обрамленные парой слов-ограничителей begin и end, то общая форма цикла выглядит следующим образом:

```
for i := start to limit do
begin
  предложение 1;
  предложение 2;
  .............
  предложение n
end;
```

В этой структуре при каждом входе в тело цикла будут выполняться предложение 1, предложение 2, ..., предложение n.

В варианте цикла For со словом *downto* (вместо *to*) переменная пробегает последовательность значений от начального к конечному в обратном порядке, уменьшая каждый раз на 1. Внешне это выглядит так:

```
for i:= limit downto start do
  тело цикла
```

1. Наберите следующую программу. Компьютер выведет на экран в столбик 15 случайных чисел от 8 до 39.

```
program sly;
uses crt;
var c, s : integer;
begin
 writeln('15 случайных чисел от 8 до 40');
  for c:=1 to 15 do
 begin
```

```
   s:= random(32) + 8;  {Генерируется случайное число в диапазоне от 0 до 31, затем
                         к нему прибавляется 8, получаем случайное число от 8 до 39}
   writeln (s);
  end;
end.
```

Самостоятельно измените программу так, чтобы на экране были напечатаны в строчку 8 случайных дробных чисел от 5 до 25.

2. Напечатать таблицу стоимости порций сыра стоимостью 280 рублей от 100 г до 1 кг с шагом 100 г.

```
program sir;
uses crt;
var s,k:real;
   c:integer;
begin
  clrscr;
  writeln('таблица стоимости порций сыра');
  for c:=1 to 10 do
  begin
    k:=280*c/10;
    writeln (c*100,' г стоит ',k,' рублей');
  end;
end.
```

Задачи для самостоятельного решения

1. Вычислить сумму первых 10 натуральных чисел. Использовать счетчик типа S=S+A, т.е. счетчик суммы. Использовать управляющую переменную цикла.
2. Напечатать таблицу соответствия между весом в фунтах и весом в кг для значений от 1 до 10 фунтов с шагом 1 фунт. 1 фунт=400 г.
3. Напечатать таблицу перевода расстояний в дюймах в сантиметры (1 дюйм=2.54 см) для значений от 1 до 10 дюймов с шагом 1.
4. Напечатать таблицу перевода температуры по Фаренгейту в градусы по Цельсию от 15 до 30. Перевод осуществляется по формуле F=1.8*C+32.
5. Запросить с клавиатуры в цикле 5 любых целых чисел, найти их сумму и среднее арифметическое, результаты вывести на экран.
6. Напечатать все нечетные натуральные числа от 1 до 50 в столбик. Использовать управляющую переменную цикла.
7. Напечатать все четные, натуральные числа в диапазоне, заданном пользователем с клавиатуры в строку. Использовать управляющую переменную цикла.
8. Составить таблицу умножения для числа 12 в виде:
   12 * 2 = 24
   12 * 2 = 24
   12 * 3 = 36 и т.д. до
   12 * 10 = 120
9. Вычислить сумму квадратов первых 7 натуральных чисел.
10. Вычислить все числа Фибоначчи от 3-го до N-го. Числа Фибоначчи образуют последовательность, у которой каждый очередной член равен сумме двух предыдущих:
    0 → 1 → 1 → 2 → 3 → 5 → 8 → 13 → 21 → 34 ...
    2+3=5 13+21=34

Упражнения

Выполните задания с *For1* по *For25* по электронному задачнику (*стр.25*). Открыть его можно в режиме *Помощь – Электронный задачник РТ*.

ТЕМА 5. Графика в Паскаль АВС

Основные графические процедуры

По умолчанию размеры графического экрана 640 на 400 точек.

Все графические объекты имеют определенный цвет. Каждому цвету соответствует название:

clBlack – черный	clRed – красный
clPurple – фиолетовый	clNavy – темно-синий
clWhite – белый	clGreen – зеленый
clMaroon – темно-красный	clBrown – коричневый

clBlue – синий
clSkyBlue – голубой
clYellow – желтый
clCream – кремовый
clAqua – бирюзовый
clOlive – оливковый
clFuchsia – сиреневый
clTeal – сине-зеленый
clGray – темно-серый
clLime – ярко-зеленый
clMoneyGreen – цвет зеленых денег
clLtGray – светло-серый
clDkGray – темно-серый
clMedGray – серый
clSilver – серебряный

Также можно задать цвет номером, например random(16777215) – случайный цвет из всей палитры цветов Паскаля

Ниже указаны некоторые процедуры модуля GraphABC, применяющиеся для построения примитивов.

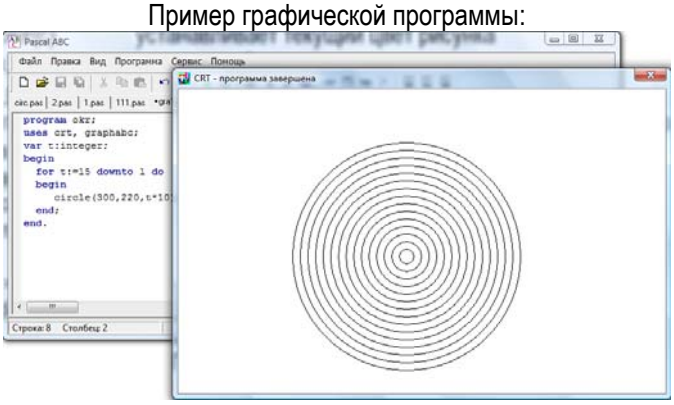
строка	*результат*
setpixel(x,y,c)	построить точку (x,y) цветом c
lineto(x,y)	рисует отрезок от текущего положения пера до точки (x,y)
line(x1,y1,x2,y2)	соединить две точки отрезком
rectangle(x1,y1,x2,y2)	построить прямоугольник с заданными концами диагонали и сторонами, параллельными осям координат
circle(x,y,r)	построить окружность с центром (x,y) и радиусом R
arc(x,y,a,b,r)	построить дугу окружности: a,b – начальный и конечный углы в градусах
ellipse(x1,y1,x2,y2)	нарисовать эллипс, заданный описанным прямоугольником с вершинами (x1,y1) и (x2,y2)
setpencolor(c)	устанавливает текущий цвет рисунка
floodfill(x,y,c)	заливает область цветом c, начиная с точки (x,y)

Пример программы

Начертить на экране 15 окружностей уменьшающегося радиуса, чтобы радиус был в 10 раз больше номера окружности.

```
program okr;
uses crt, graphabc;
var t:integer;
begin
  for t:=15 downto 1 do
    circle(300,220,t*10);
end.
```

Пример графической программы:



Простейшие задачи на графику:

1. В центре окна (640x400) нарисовать окружность радиусом 40 и закрасить ее красным цветом.
2. В центре окна (640x400) нарисовать прямоугольник 80x50 и закрасить его зеленым цветом.
3. По углам окна (640x400) нарисовать по одной окружности радиусом 15 и закрасить их разными цветами.
4. В центре окна (640x400) нарисовать квадрат 80x80, в его центре нарисовать окружность радиусом 40 и закрасить их разными цветами.
5. Нарисовать несложный домик и около него простую фигурку человечка.
6. Нарисовать в центре экрана простой самолетик на взлетной полосе и вдоль нее несколько деревьев.

Дополнительные задания: использование графики в цикле

1. Расставить на экране 100 точек со случайными координатами X и Y ("звездное небо"). Координаты выбирать с помощью random.
2. Нарисовать в центре экрана Олимпийские кольца.
3. Расставить на экране 30 окружностей со случайными координатами X и Y, случайного радиуса и цвета ("мыльные пузыри").
4. Через точку в центре экрана провести 20 отрезков, вторая координата которых выбирается случайно ("разбитое стекло").
5. Написать графическую иллюстрацию к задаче: запросить с клавиатуры координаты точки (X, Y) и горизонтального отрезка прямой (Xn, Xk, Yn) и определить, лежит ли точка на прямой. Сообщение вывести на экран.

ТЕМА 6. Виды циклов (продолжение).
Построение изображений в графическом режиме (продолжение)

Если число повторений заранее известно, то подходящей конструкций является оператор For. В противном случае следует использовать операторы While или Repeat.

Для управления циклом можно использовать Break и Continue. *Break* преждевременно завершает оператор цикла, *Continue* продолжает цикл со следующей итерации (следующего прохода) этого оператора, не завершив текущую.
Их использование считается нежелательным и должно применяться лишь в особых случаях.

Оператор цикла с постусловием (repeat)

Общая конструкция цикла:

```
repeat
  тело цикла
until  условие
```

Цикл *repeat – until* работает следующим образом. Сначала выполняется тело цикла. По достижению пункта *until* проверяется условие. Если оно не выполняется, тело цикла выполняется снова, завершаясь новой проверкой условия. Если же условие выполнено, то тело цикла больше не повторяется, цикл завершается, а программа переходит к выполнению предложения, следующего за *until*. В этом цикле не используются ограничители *begin-end*.

Условие может быть как числовым (например, *until a=25*), тогда счетчик (a:=a+1 или любой другой шаг) должен быть реализован внутри цикла, так и нечисловым *(until keypressed*).

Таким образом, после *until* ставится условие *выхода* из цикла. Последовательность операторов выполнится, по крайней мере, один раз, поскольку проверка условия производится после каждого выполнения последовательности операторов.

Пример использования оператора цикла с постусловием:

```
program sumnum;
var num, sum : integer;
begin
 num:=0;
 sum:=0;
 repeat
   num :=num+1;
   sum :=sum+num;
   writeln (num,' ',sum);
 until num=5;
 writeln (num,' ', sum)
end.
```

Оператор цикла с предусловием (while)

Общая форма цикла while:

```
while  условие do
begin
  тело цикла
end;
```

где условие есть некоторое выражение, результатом которого могут быть значения "истина" или "ложь".
В теле цикла может использоваться один оператор или несколько, но в последнем случае их нужно объединить логическими скобками begin – end.

Цикл *while* работает следующим образом. Сначала проверяется условие. Если оно истинно, то тело цикла выполняется, затем условие проверяется снова, и процесс повторяется. Тело цикла выполняется каждый раз, когда проверка условия дает "истину". Если условие ложно, то цикл завершается, входа в тело цикла не происходит, и следующим выполняется предложение, стоящее непосредственно после цикла.

Таким образом, после *while* ставится условие *работы* цикла. Проверка условия производится до выполнения цикла.

Пример оператора цикла с предусловием:

```
k := 20;
while k > 0.1 do k := k / 2;
```

Основным отличием цикла с предусловием от цикла с постусловием является то, что тело цикла с послеусловием в любом случае выполняется хотя бы один раз. Поэтому не всегда эти циклы однозначно преобразуются друг в друга.

Наберите и проанализируйте программу "Угадай число". Здесь компьютер загадывает случайное число от 0 до 500, а пользователь должен угадать его. Компьютер при этом должен выдавать подсказки типа ДАВАЙ БОЛЬШЕ.

```
program ug_sl;
uses crt;
var a,b,p:integer;
begin
 clrscr;
 a:=random(500)+1;
 writeln ('Угадай целое число до 500');
 repeat
  readln(b);
  p:=p+1;
  if a>b then writeln ('давай больше');
  if a<b then writeln ('давай меньше');
 until a=b;
 writeln ('угадал за ',p,' попыток!');
end.
```

Задачи для самостоятельного решения

1. Самостоятельно измените программу так, чтобы компьютер запрашивал желаемое количество попыток, и при превышении этого числа сообщал загаданное число и заканчивал программу.
2. В этой программе реализована игровая ситуация для проверки вашей реакции.

```
program press;
uses crt;
var q,w:integer;
begin
 writeln('После надписи "Жми" быстрей нажми любую клавишу');
 delay (2000);                          {задержка времени}
 writeln('Жми!!!');
 q:=0;
 while not keypressed do
  q:=q+1;                               {Если не нажата любая клавиша, то наращивать q}
 writeln('Пока вы соображали, прошло ',q,' циклов');
end.
```

Измените программу так, чтобы использовать цикл repeat.

3. Начертить в центре экрана квадрат со стороной 50.
4. Начертить на экране 5 окружностей уменьшающегося радиуса, начиная с 100 точек с шагом 5, со смещением центра по горизонтали вправо на 20 точек.
5. Начертить 100 закрашенных эллипсов со случайными координатами, размером и цветом.
6. Шарик, двигающийся из левого верхнего угла экрана в правый верхний.
7. Шарик, двигающийся из левого верхнего угла экрана в правый нижний.
8. Шарик, двигающийся из левого верхнего угла экрана по периметру в ту же точку.
9. Шарик, падающий вниз из середины экрана.
10. Написать программу движения по экрану окружности радиусом 10 по диагонали из левого верхнего в правый нижний угол экрана.
11. Написать программу движения по экрану окружности радиусом 10 по диагонали из правого верхнего в левый нижний угол экрана.
12. Начертить прямоугольник с координатами углов x1=130, y1=80, x2=220, y2=200 и в нем расставить 100 случайных точек ("звезды в окне").
13. Написать программу движения по экрану прямоугольника со сторонами 15 и 10 точек по вертикали сверху вниз.
14. Написать программу движения по периметру экрана окружности радиусом 10.
15. Написать программу движения по экрану окружности радиусом 10 по вертикали в центре экрана снизу вверх.
16. Написать программу движения по экрану окружности радиусом 10 по горизонтали центре экрана справа налево.

17. Нарисовать на экране ряд окружностей уменьшающегося радиуса от 100 до 10, по горизонтали справа налево "воронка". По оси X использовать весь экран.
18. Написать программу одновременного (по очереди) движения по вертикали на экране 5 точек с любыми координатами ("капли дождя").
19. В центре экрана из отдельных элементов (линия, эллипс, окружность, дуга, прямоугольник) в графике нарисуйте свои инициалы.
20. Начертить в центре экрана 5 квадратов со стороной, увеличивающейся от 50 с шагом 20.
21. Написать программу одновременного (по очереди) движения по экрану трех окружностей радиусом 5 по горизонтали x1=100, x2=130, x3=150 со случайным приращением координаты по оси X для каждой окружности ("бег наперегонки").

ТЕМА 7. Обработка строк

Для работы со строками существуют следующие стандартные процедуры и функции:

d:=copy(a,x,y) функция, возвращает копию из y (integer) знаков части строки a (string), начиная с позиции x (integer). Формат: d:=copy(a,x,y). Результат записывается в переменную d, например:

```
a:='бегемот';
d:=copy(a,5,3);
writeln(d);
```

результат - мот

delete(a,x,y) процедура, удаляет y (integer) знаков части строки a (string), начиная с позиции x (integer). Формат: delete(a,x,y). Результат записывается в переменную a, например:

```
a:='антрекот';
delete(a,1,5);
writeln(a);
```

результат - кот

insert(a,b,x) процедура, вставляет строку a (string) в строку b (string), в позицию x (integer). Формат: insert(a,b,x). Результат записывается в переменную b, например:

```
b:='коледж';
a:='л'
insert(a,b,3);
writeln(b);
```

результат - колледж

x:=pos(a,b) функция, возвращает номер позиции с которой располагается подстрока a (string), в строке b (string). Формат: x:=pos(a,b). Результат записывается в переменную x, например:

```
b:='колледж';
a:='л'
x:=pos(a,b);
writeln(b);
```

результат – 3

a:=IntToStr(x) функция, преобразует число x (integer) в строку a (string). Например:

```
x:=2000;
a:=IntToStr(x);
writeln(a);
```

результат - '2000'

a:=FloatToStr(x) функция, преобразует число x (real) в строку a (string).

x:=StrToInt(a) функция, преобразует строку a (string) в число x (integer). Результат записывается в переменную x, например:

```
a:='2000';
x:=StrToInt(a);
writeln(x);
```

результат – 2000

x:=StrToFloat(a) функция, преобразует строку a (string) в число x (real) .

x:=length(a) функция, возвращает длину строки a (string). Например:

```
a:='школа';
x:=length(a);
writeln(x);
```

результат 5.

x:=ord(a) функция, возвращает цифровой код символа a (char). Например:

```
a:='Й';
x:=ord(a);
```

```
        writeln(x);
```
результат 137

a:=chr(x) функция, возвращает символ a (char) по его цифровому коду. Например:

```
        x:=137;
        a:=chr(x);
        writeln(a);
```
результат - Й.

Эта программа позволяет получить цифровой код нажатой клавиши с помощью функции ord.

```
uses crt;
var k:integer;
   c:char;
begin
 writeln('Вводи с клавиатуры символы: ');
 writeln('Выход из программы клавишей ESC');
 repeat
   c:=readkey;
   k:=ord(c);
   writeln(c,' - ',k);
 until k=27;                    {код клавиши ESC}
end.
```

Задачи для самостоятельного решения

1. Слова и фразы, которые можно читать справа налево и слева направо. Такие слова или фразы называются перевертыши или палиндромы. Например, ШАЛАШ, РОТОР.

   Данная программа проверяет, является ли слово палиндромом. При вводе выражения пробелы не вводить.

```
program palindrom;
uses crt;
var     q:integer;
   a,b,c:string;
begin
 clrscr;
 writeln('Введите слово - палиндром ');
 readln(a);
 for q:=1 to length(a) div 2 do      {проверка до середины слова}
 begin
   b:=copy(a,q,1);                  {очередная буква слева}
   c:=copy(a,length(a)+1-q,1);      {соответствующая буква справа}
   if b<>c then
   begin
     write('Это не палиндром !!!');
     exit;
   end;
 end;
 write('Правильно.');
end.
```

Переделайте программу так, чтобы в ней не использовался exit.

2. Если выражение состоит из нескольких слов, то нужно, чтобы компьютер анализировал наличие пробелов. Это реализовано в следующей программе:

```
program palindrom2;
uses crt;
var q:integer;
   a:string;
begin
 clrscr;
 write('Введите фразу ');
 readln(a);
 {==========Алгоритм удаления пробелов==========}
 for q:=1 to length(a) do
 if copy(a,q,1)=' ' then delete(a,q,1);
```

```
 {=========================================}
 for q:=1 to length(a) div 2 do
   begin
     if copy(a,q,1)<>copy(a,length(a)+1-q,1) then
     begin
       write('Это не палиндром, вот пример палиндрома: "аргентина манит негра"');
       exit;
     end;
   end;
 write('Правильно.');
end.
```

Переделайте программу так, чтобы в ней не использовался exit.

3. Из числовых переменных, равных 4, 7, 9 с помощью строковых функций получить число 794 и вывести его на экран.
4. Запросить с клавиатуры слово и число до 10000, определить количество в них знаков, напечатать результат на экране.
5. Запросить с клавиатуры три числа, сложить их, затем определить количество знаков в сумме, напечатать результат на экране.
6. Составить программу, чтобы компьютер сгенерировал 10 случайных целых трехзначных чисел. Длину определять с помощью строковых функций.
7. Из слова "ВЕЛИКОЛЕПНЫЙ" с помощью строковых функций получить слова: ВЕЛИК, КОЛ, ЛИК, ЛЕВ, ВЕК и вывести их на экран.
8. Из своих фамилии и имени с помощью строковых функций получить 10 слов и вывести их на экран.
9. Составить программу, чтобы компьютер сгенерировал случайное целое трехзначное число, с помощью строковых функций определил, сколько в нем единиц, десятков и сотен, вывел эту информацию на экран.
10. Составить программу, чтобы компьютер сгенерировал три случайных целых однозначных чисел и из них сформировал трехзначное число, все числа вывел на экран.

Упражнения.

Выполните задания с *String1* по *String10* по электронному задачнику (*стр.72*). Открыть его можно в режиме *Помощь – Электронный задачник PT*.

Недостающие для решения строковые процедуры и функции найдите через встроенную справку.

ТЕМА 8. Массивы

Массивом называется способ организации однородной информации, при котором облегчается доступ к любому элементу информации по его индексу (номеру) и достаточно легко производится обработка информации.

Объявление массивов производится в блоке описания следующим образом:

x:array[1..100] of integer; – массив на 100 элементов для целых чисел с индексами от 1 до 100.

a,b:array[10..29] of string; – два массива по 20 элементов для символьных переменных с индексами от 10 до 29.

Двумерный массив объявляется так:

y:array[1..10,1..5] of real; – массив размером 10 на 5 для вещественных чисел с индексами от 1 до 10 и от 1 до 5.

Конкретный элемент массива обозначается с помощью ссылки на переменную (имя) массива, за которой в квадратных скобках указывается индекс данного элемента, например: *a[7]*

Строковая переменная типа string также может рассматриваться как массив переменных типа char, например: *b:string[20]* Это дает доступ к каждому символу в строковой переменной, если значение символа имеет тип *Char*.

Конкретный символ в строковой переменной обозначается с помощью ссылки на строковую переменную, за которой указывается индекс, определяющий позицию символа в строке, например: *b[15].*

В программах можно использовать присваивание элементам массива:

```
a[5] := 8; числовому,  b[3] := 'Виктор'  строковому
```

и выборочную печать элементов:

```
write (a[5]);
writeln (b[3]);
```

С массивами удобно работать с помощью циклов:

- заполнение

```
for t := 1 to 20 do
  begin
  write('Введите ',t,'-й элемент');
  readln(a[t]);
  end;
```

- вывод на печать

```
            for t := 1 to 20 do
              writeln(t,'-й элемент равен',a[t]);
```

- выбор по условию

```
        for t := 1 to 20 do
        if a[t]>=3 then writeln('У студента ',b[t],' оценка по экзамену ',a[t]);
```

- при использовании двумерных массивов применяются вложенные циклы.

```
var m,k:integer;
    x:array[1..10,1..5] of string;
begin
  for m:=1 to 10 do
   for k:=1 to  5 do
   begin
   write('Введите фамилию жильца ',m,'-го подъезда и ',k,'-го этажа');
   readln(x[m,k]);
  end;
  write('Полный список жильцов:');
  for m:=1 to 10 do
    for k:=1 to  5 do
      write(m,'-й подъезд, ',k,'-й этаж',x[m,k])
end.
```

Просмотрите предлагаемые ниже задачи, разберитесь в их работе. При необходимости наберите и изучите программы.

1. Программа "исчезающие звезды".

В цикле for массивы заполняются случайными числами: x – до 640 (координата x), y – до 400 (координата y), c – до 1000000 (цвет). В цикле repeat последовательно проходятся все элементы массива и по указанным координатам сначала окружность (круг) рисуется черным цветом (звезды гаснут), а затем по новым случайным координатам и цвету вспыхивают звезды. Процесс продолжается до нажатия любой клавиши.

```
uses crt, graphabc;
var a,b,d,e,f:integer;
  x,y,c:array[1..100] of integer;    {3 числовых массива по 100 элементов каждый}
begin
floodfill(10,10,clblack);                {заливка фона черным}
   for b:=1 to 100 do begin              {заполнение массивов}
     x[b]:=random(640);
     y[b]:=random(400);
     c[b]:=random(1000000);
   end;
   repeat
     for b:=1 to 100 do begin
       SetPenColor (clblack);              {звезды гаснут}
       circle(x[b],y[b],2);
       floodfill(x[b],y[b],clblack);       {заливка звезды черным}
       x[b]:=random(640);
       y[b]:=random(400);
       c[b]:=random(1000000);
       delay(25);
       setpencolor(c[b]);                  {звезды вспыхивают}
       circle(x[b],y[b],2);
       floodfill(x[b],y[b],c[b]);          {заливка звезды}
     end;
   until keypressed
end.
```

2. Сортировка массива. Массив заполняется случайными числами, которые затем упорядочиваются и печатаются на экране.

```
uses crt;
var a,b,c,d:integer;
 q:array[1..15] of integer;
 begin
 clrscr;
```

```
 writeln('исходные случайные числа:');
 for a:=1 to 15 do
 begin
  q[a]:=random(50);
  write(q[a],'  ');
 end;
 writeln;
 for a:=1 to 15 do
 begin
  for b:=1 to 14 do
  begin
   if q[a]>q[b] then      {Числа располагаются в порядке убывания }
   begin                 {смена знака на < расположит числа по возрастанию}
    c:=q[b];
    q[b]:=q[a];
    q[a]:=c;
   end;
  end;
 end;
 writeln;
 writeln('отсортированные случайные числа:');
 for a:=1 to 15 do
  write(q[a],'  ');
 end.
```

3. Неповторяющиеся случайные числа до 100.

```
uses crt;
var a,b,c,d:integer;
 q:array[1..10] of integer;
 begin
  clrscr;
  a:=1;
  while a<11 do
  begin
   q[a]:=random(100)+1;
   if a>1 then
   for b:=1 to a-1 do
   begin
    if q[a]=q[b] then a:=a-1;
   end;
   a:=a+1;
  end;
  for a:=1 to 10 do writeln(q[a]);
 end.
```

Задачи для самостоятельного решения

1. Описать числовой массив на 5 элементов и заполнить его присваиванием любыми числами, распечатать содержимое элементов массива
2. а) в столбик
3. б) в строку.
4. Описать числовой массив на 5 элементов и заполнить его в цикле с клавиатуры любыми числами, распечатать содержимое элементов массива.
5. Описать символьный массив на 5 элементов и заполнить его присваиванием именами, распечатать содержимое элементов массива.
6. Описать символьный массив на 5 элементов и заполнить его в цикле с клавиатуры именами, распечатать содержимое элементов массива.
7. Описать числовой массив на 25 элементов и заполнить его случайными целыми числами, каждое из которых лежит в пределах от 10 до 50, распечатать содержимое элементов массива в строку.

8. Описать числовой массив на 15 элементов и заполнить его случайными целыми числами, каждое из которых лежит в пределах от 10 до 100, распечатать содержимое элементов массива в строку, рассчитать среднее арифметическое элементов и вывести его на экран с поясняющим текстом, распечатать содержимое тех элементов массива, значение которых больше среднего арифметического.
9. Найти сумму элементов массива с четными номерами, содержащего 10 чисел.
10. Найти сумму положительных элементов заданного массива, содержащего 5 чисел.
11. Задано 2 массива, содержащих по 5 чисел. Сформировать новый массив, включая в него сначала все элементы первого массива, затем все элементы второго массива.
12. Задан массив, содержащий 10 чисел. Найти значение и индекс максимального (минимального) элемента.
13. Информация о температуре воздуха за месяц задана в виде массива. Определить, сколько раз температура опускалась ниже 0^0C.
14. Занести в массив карту расположения кораблей в игре "Морской бой" и смоделировать игру.
15. Задан массив, содержащий 10 чисел. Сформировать 2 массива, включая в массив первый четные (по номеру) элементы массива в порядке их следования, а во второй массив – нечетные.
16. Запросить с клавиатуры 5 слов, занести их в массив, определить количество в каждом из них знаков, занести их в другой массив, напечатать на экране содержимое обоих массивов в табличной форме.
17. Запросить с клавиатуры 5 слов, занести в массив только те слова, количество букв в которых равно четырем. Распечатать на экране содержимое массива.
18. Запросить с клавиатуры слово, определить количество в нем знаков, разрезать слово на отдельные буквы, которые занести в другой массив, распечатать на экране слово *справа налево*.
19. В заданном двумерном массиве поменять местами следующие два элемента:
    a. Минимальный и максимальный.
    b. Два наименьших.
    c. Первый и последний положительные.
    d. Два наибольших.
    e. Два первых положительных.
    f. Два последних отрицательных.
    g. Два первых четных.
    h. Два последних нечетных.
    i. Минимальный положительных и максимальный отрицательный.

Упражнения.
Выполните задания с *Array1* по *Array10* по электронному задачнику (*стр.49*).
Выполните задания с *Array51* по *Array60* по электронному задачнику (*стр.54*).
Выполните задания с *Array65* по *Array69* по электронному задачнику (*стр.55*).
Выполните задания с *Matrix1* по *Matrix4,* с *Matrix7* по *Matrix10* по электронному задачнику (*стр.64*).
Открыть его можно в режиме *Помощь – Электронный задачник РТ*.

ТЕМА 9. Работа с файлами

Работа с файлами применяется для хранения в них дополнительной и изменяющейся информации. В файлах размещаются данные, предназначенные для длительного хранения. Каждому файлу присваивается используемое при обращении к нему уникальное имя. Файлы широко используются при решении различных задач.

При работе с файлами на Паскале следует учитывать такую особенность: работа с файлом по его имени невозможна, а для доступа к файлу необходимо сначала связать его с некоторой файловой переменной особого типа, и в будущем использовать эту переменную вместо имени файла.

Для работы с внешними файлами в блоке описания необходимо определить файловую переменную, которая будет представлять файл внутри программы *var f:text;*
в качестве типа файловой переменной указывается стандартное имя *text*.

В данном случае файловая пременная типа *text* имеет имя *f*.

Для работы с файлами можно использовать следующие операции:

- *assign* для установления связи между файловой переменной и внешним именем файла.
  Например: assign (имя_файловой_переменной, имя_файла);
- *erase* для удаления файла с диска.
  Например: erase (имя_файловой_переменной);
- *reset* открытие существующего файла для ввода информации.
  Например: reset (имя_файловой_переменной);
- *rewrite* создание и открытие нового файла для записи в него информации.

Например : rewrite (имя_файловой_переменной).

- *close* закрытие файла по окончании работы с ним.

Например: close (имя_файловой_переменной).

Для работы с файлами используются стандартные функции:

- *eof* нахождение конца файла,
- *eoln* нахождение конца строки данных.

Считывание информации из файла и занесение информации в файл организуется стандартными операторами:

write (имя_файловой_переменной, имя_записываемой_переменной) записывает символ в файл, связанный с файловой переменной,

readln (имя_файловой_переменной, имя_читаемой_переменной) чтение из текстового файла строки.

Содержимое файла можно выводить на экран, на принтер, в файл.

Примеры программ работы с файлами:

1. Для выполнения этой задачи сначала необходимо создать дайл для последующего удаления. Это можно сделать набрав и сохранив в своей папке файл с любым текстом из Паскаль АВС. Файл должен распологаться его в том же каталоге, где и ваши программы.

Удаление с жесткого диска (винчестера) файлов с помощью программы, созданной на языке Паскаль.

```
uses crt;
var f:text;
   a:string;
 begin
   write('Введите имя удаляемого файла с расширением ');
   readln(a);
   writeln('Удаляется  файл  ');
   assign(f,a);          {Связывание существующего файла с файловой переменной}
   erase(f);             {Удаление файла}
   writeln('Файл удален');
 end.
```

Наберите, сохраните программу в своей папке. Убедитесь, что файл действительно удален.

2. Программа создает новый текстовый файл и записывает в него информацию.

При создании файла в программе дайте ему расширение *pas*, тогда Вы сможете обратиться к файлу, не выходя из Паскаль АВС.

```
uses crt; var f:text;
   g:string;
   c:char;
 begin
   write('Введите имя создаваемого файла (латинскими буквами)');
   read(g);
   assign(f,g);
   rewrite(f);          {Открытие нового текстового файла (старый затирается) }
   writeln('Введите текст, который нужно записать в файл: ');
   writeln(' По окончании нажми Esc');
   repeat
     write(f,c);        {Записывает символ в файл, связанный с файловой переменной}
     write(c);          {Печать введенного символа на экране}
     if c=#13 then writeln;   {Если нажат Enter, то перевод строки}
   until c=#27;               {код клавиши Esc}
   close(f);
 end.
```

После набора и проверки программы убедитесь, что ваш файл действительно создан и информация записана.

3. Программа читает из текстового файла и выводит содержимое на экран.

```
uses crt;
var f:text;
 a,g:string;
   q:integer;
```

```
   c:char;
begin
   clrscr;
   write('Введите имя файла (латинскими буквами) из которого нужно прочесть данные ');
   read(g);
   assign(f,g);
   reset(f);                  {Открытие текстового файла для чтения из него информации}
    repeat
    readln(f,a);              {Чтение из текстового файла строки}
    for q:=1 to length(a) do
      begin
      c:=a[q];
      c:=chr(ord(c));         {Обработка очередного символа}
      write(c);
      if c=#13 then writeln;
      end;
    until eof(f);             {Проверка, не кончился ли файл}
   close(f);
writeln('   чтение информации завершено.');
end.
```

Задачи для самостоятельного решения

1. Составить программу, чтобы она создавала файл, записывала в него любой текст, а затем считывала и выводила на экран.
2. В любую имеющуюся программу добавить блок запроса пароля и сравнивать его с хранящимся в файле, если пароль не совпадает, то программу не запускать.
3. Задачу 2 изменить так, чтобы пароль шифровался по любой схеме, а при проверке программа его самостоятельно расшифровывала.

ТЕМА 10. Работа с процедурами и функциями

Процедуры и функции, написанные программистом, предназначены для оптимизации программ.

Основным их преимуществом является возможность многократного использования, более легкого и удобного тестирования и отладки независимо от других модулей.

В Паскале главная программа начинается с декларативной части, в которую, в частности, входят и описания всех процедур и функций. Обычно, в Паскале декларативная часть состоит из раздела определения констант, за которым следует раздел объявления переменных, а затем все процедуры и функции. Главное правило – любой объект должен быть описан до его первого использования.

Пример задачи: написать программу, которая определяет сумму квадратов n первых натуральных чисел. Сумма определяется в функции пользователя.

```
   program summa_kv;
   uses crt;
   var e, f:integer;
   function powers (n: integer) : integer;  {объявление функции}
   var i, sum:integer;
   begin
   clrscr;
   sum:=0;
   for  i:=1 to n do
     sum:=sum + sqr(i);
   powers:= sum
   end;              {конец функции}
   begin             {основная программа}
   write ('Сколько чисел нужно сложить?');
   readln (e);
   f:= powers (e);     {вызов функции, значение e передается в n}
   writeln ('сумма квадратов  ',e,' чисел');
   writeln ('Число членов = ',e,'. Сумма = ',f);
   end.                  {конец основной программы}
```

Функции обычно применяются в расчетных задачах, когда необходимо выполнить ряд вычислений и передать их значения основной программе.

Первая строка описания функции называется заголовком. Она включает в себя имя функции, имя и тип каждого формального параметра, а также тип результата.

Например: function powers (n:integer):real;

Здесь имя функции – powers, у нее один формальный параметр – n, принадлежит типу integer. Функция возвращает результат типа integer.

За заголовком функции следует объявление локальных переменных подпрограммы.

Например: var i, sum:integer;

Локальные переменные, объявленные в подпрограмме, никак не связаны с объектами главной программы, даже если у них одинаковые имена.

За объявлениями, если они есть, следует пара ограничителей begin-end, окружающих набор предложений, составляющих саму функцию. При этом самым последним должно быть выполнено предложение, которым имени функции назначается некоторое значение, оно будет возвращено в главную программу.

Процедуры применяются, когда приходится решать задачи, в которых интересует не вычисление какого-то конкретного значения, а выполнение некоторой совокупности действий, например, отпечатать список чисел. Процедура обычно не возвращает в основную программу никакого значения. А если и передает данные, то только через глобальные переменные. Но существуют, так называемы, процедуры с параметрами, в которые (и даже из которых) можно передавать данные. В этой теме будут рассмотрены и примеры таких процедур. В Электронном задачнике даются задачи именно на такие процедуры.

Между функцией и процедурой имеется несколько различий. Наиболее существенное состоит в том, что функция всегда возвращает одно конкретное значение, тогда как процедура нет. Это различие отражается в особенности объявления процедур и функций.

Заголовок процедуры оформляется почти так же, как заголовок функции с той разницей, что в нем отсутствует указание о типе возвращаемого значения.

Пример: procedure test (a: real; var b: integer);

где test – заголовок процедуры, параметр – a типа real, параметр b типа integer.

Вызов процедуры из главной программы представляет собой просто имя процедуры со списком аргументов. Например: test (x,y). Процедуры вообще могут не иметь параметров.

Например, сумма n первых натуральных чисел:

```
program primer;    // процедура без параметров
uses crt;
var a,k,sum : integer;
procedure add;          {объявление процедуры}
begin
writeln(a ,'  чисел,  сумма  ',sum);
end;                    {конец процедуры}
begin                   {начало основной программы}
  clrscr;
writeln('последовательное сложение натуральных чисел');
write('сколько первых натуральных чисел сложить?   ');
readln(k);
  sum:=0;
  for a:=1 to k do
  begin
  sum:=sum+a;
  add;                  {вызов процедуры}
  end;
end.
```

Пример решения задач из Электронного задачника:

```
Program Proc1;      // процедура с параметрами
uses crt;
var i: integer;
   t ,c:real;
procedure powerA3(a:real;var b:real); {формальные параметры процедуры}
```

```
begin
  b:=power(a,3);       {a - число, c - степень}
  end;
begin                        {основная программа}
  writeln('Вычисление 3-й степени числа:');
  for i:=1 to 5 do
  begin
    write('введите число  ');
    readln(t);
    powerA3(t,c);       {фактические параметры процедуры}
  writeln(3-я степень числа:',c:2:1);
  end;
end.

Program Proc3;          // процедура с параметрами
uses crt;
var a,b,c,d,Ar,Ge:real;
procedure Mean(x,y:real;var AMean,GMean:real);
begin
  AMean:=(x+y)/2;
  GMean:=sqrt(x*y);
end;
begin
  writeln('Введите числа a,b,c,d');
  readln(a,b,c,d);
  Mean(a,b,Ar,Ge);
  writeln('ср-ар',Ar:2:1,'ср-геом',Ge:2:1);
  Mean(a,c,Ar,Ge);
  writeln('ср-ар',Ar:2:1,'ср-геом',Ge:2:1);
  Mean(a,d,Ar,Ge);
  writeln('ср-ар',Ar:2:1,'ср-геом',Ge:2:1);
end.


program Proc16;         // функция
      uses crt;
      var a, b:real;
      function sign(x:real):integer;
      begin
      if x<0 then sign:=-1;
      if x=0 then sign:=0;
      if x>0 then sign:=1;
      end;
      begin
      writeln ('vv a i b');
      readln (a,b);
      writeln ('sign(a)+sign(b) = ',sign(a)+sign(b));
      end.

program Proc17;             // функция
      uses crt;
      var a1, b1,c1:real;
          i:integer;
      function RootsCount(a,b,c:real):string;
      var d:real;
      begin
        d:=sqr(b)-4*a*c;
        if d<0 then RootsCount:='нет корней';
```

```
    if d=0 then RootsCount:='один корень';
    if d>0 then RootsCount:='два корня';
  end;
  begin
  for i:=1 to 3 do
  begin
    writeln ('введи коэффициенты квадратного уравнения a, b,c');
    readln (a1,b1,c1);
    writeln (RootsCount(a1,b1,c1));
  end;
  end.
```

Наберите и отладьте следующие программы, разберитесь в их работе:

1. Программа "Неповторяющиеся случайные числа" с использованием массива и процедуры без параметров.

```
uses crt;
var a,b,c,d:integer;
   q:array[1..15] of integer;
procedure qw;
begin
 for b:=1 to 5 do
 begin
   if a=b then continue;
   if q[a]=q[b] then begin q[a]:=random(20); qw; end;
 end;
end;
begin
clrscr;
for a:=1 to 5 do
 q[a]:=random(20);
for a:=1 to 5 do
 qw;
for a:=1 to 5 do
 writeln (q[a]);
end.
```

2. Проверка целого числа на четность (использование функции).

```
  var m:integer;
  function Chet(n:integer):boolean;  { Функция проверки целого числа на четность }
  begin
    if (n mod 2)= 0 then Chet:=TRUE else Chet:=FALSE;
  end;
  begin
    writeln('Введите целое число и нажмите Enter');
    writeln('Для завершения введите 100');
    repeat
      readln ( m );
      if chet(m)
        then writeln( m,' - четное число')
        else writeln( m,' - нечетное число   ');
    until m = 100;
  end.
```

3. Вычисление длины и площади окружности (использование процедуры с параметрами).

```
  uses crt;
  var t,l,s:real;         {глобальные переменные радиус, длина и площадь окружности}
  procedure SqLeOkr(r:real);          {процедура, r формальный параметр процедуры}
  begin
    s:=pi*r*r;                        {r - радиус, s - площадь круга, l - длина окружности}
    l:=2*pi*r;
```

```
end;
begin                                {основная программа}
  writeln('Вычисление длины окружности и площади круга:');
  write('Задайте радиус и нажмите Enter  ');
  readln(t);
  l:=0;
  s:=0;
  SqLeOkr(t);                        {фактический параметр процедуры}
  writeln('Радиус окружности: ',t:3:1);
  writeln('Длина окружности: ',l:3:1,',площадь: ',s:3:1);
end.
```

Упражнения

Выполните задания с *Proc2* по *Proc15* по электронному задачнику (*стр.37*) и с *Proc18* по *Proc22* по электронному задачнику (*стр.39*). Открыть его можно в режиме *Помощь – Электронный задачник PT*.