ДОНСКОЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

УПРАВЛЕНИЕ ДИСТАНЦИОННОГО ОБУЧЕНИЯ И ПОВЫШЕНИЯ КВАЛИФИКАЦИИ

Кафедра «Программное обеспечение вычислительной техники и автоматизированных систем»

Учебно-методическое пособие

по дисциплине

«ОСНОВЫ ПРОГРАММИРОВАНИЯ»

Автор
Кузин А.П.

Ростов-на-Дону, 2018

Аннотация

Учебно-методическое пособие предназначено для студентов очной формы обучения направлений 09.03.04 «Программная инженерия» и 02.03.03 «Математическое обеспечение и администрирование информационных систем»

Авторы

Ст.преподаватель
каф. ПОВТиАС
Кузин А.П.

Оглавление

1. ЛАБОРАТОРНАЯ РАБОТА №1: РАБОТА С ОДНОМЕРНЫМИ МАССИВАМИ

1.1. Цель работы

Научиться использовать одномерные массивы при решении задач. Освоить основные правила работы с одномерными массивами.

1.2. Теоретическая часть

Массив — это пронумерованная последовательность величин одинакового типа, обозначаемая одним именем. Элементы массива располагаются в последовательных ячейках памяти, обозначаются именем массива и индексом. Каждое из значений, составляющих массив, называется его компонентой (или элементом массива).

Массив данных в программе рассматривается как переменная структурированного типа. Массиву присваивается имя, посредством которого можно ссылаться как на массив данных в целом, так и на любую из его компонент.

Если за каждым элементом массива закреплен только один его порядковый номер, то такой массив называется линейным. Вообще количество индексов элементов массива определяет размерность массива. По этом признаку массивы делятся на одномерные (линейные), двумерные, трёхмерные и т.д.

Пример объявления одномерного массива A из 10 элементов целого типа.

```
var a: array[1..10] of integer;
```

или

```
type masInt = array[1..10] of integer;
var a:masInt;
```

Пример обращения к третьему элементу массива **a[3]:=4**;

При работе с массивом часто используются циклы **for** и **foreach**.

Например:

```
var a:array [1..10] of integer;
i:integer;
begin
for i:=1 to 10 do a[i]:=random(100);
foreach x: integer in a do write(x, ' ');
```

end.
Элементы массива можно инициализировать при описании
var a: array of string:= ('Иванов','Петров','Сидорова');

Динамический одномерный массив

Динамическим называется массив, память под который выделяется в процессе работы программы.

Динамические массивы индексируются с нуля, тип индекса - только целое.

Объявление массива

var a: array of integer;

Выделение памяти

SetLength(a,n);

До выделения памяти a равно nil.

Length(a) - возвращает длину массива.

В процессе работы можно изменить память, занимаемую динамическим массивом. Для этого следует повторно вызвать SetLength:

```
var a : array of integer;
SetLength(a,3);
 a[0] := 1;
a[1] := 3;
a[2] := 2;
SetLength(a,4);
```

Пример 1. Заполнение элементов одномерного массива

объявление одномерного массива из 10 целых элементов

```
var a: array [1..10] of integer;
begin
```

цикл со счетчиком для заполнения элементов массива случайными числами в интервале от 0 до 99 и вывод их на экран

```
 for i:integer := 1 to 10 do
 begin
 a[i]:=Random(100);
 write(a[i],' ');
 end;
end.
```

Пример 2. Поиск минимального элемента массива

```
begin
```

цикл для заполнения элементов массива случайными числами

```
for i:integer := 1 to 10 do
 begin
 a[i]:=Random(100);
 write(a[i],' ');
 end;
 writeln;
```

объявление вспомогательных переменных для поиска минимального элемента

```
 var min, num: integer;
 min:=a[1]; num:=1;
```

поиск минимального элемента путем последовательного перебора

```
 for var i:=2 to 10 do
 if a[i]<min then
 begin
 min:=a[i]; num:=i;
 end;

 writeln('минимальный элемент массива A[',num,'] = ',min);
end.
```

Пример 3. Расчет сумы элементов массива

```
var a: array [1..10] of integer;
```

```
begin
for i:integer := 1 to 10 do
begin
 a[i]:=Random(100);
 write(a[i],' ');
end;
writeln;
 var s: integer;
 s:=0;
 for var i:=1 to 10 do
 s:=s+a[i];
 writeln('сумма элементов равна ',s);
end.
```

Пример 4. Подсчет элементов массива

Посчитать число элементов массива равных трем.

```
var a:array[1..10] of integer;
 num:integer;

begin

for var i:=1 to 10 do
begin
 a[i]:=random(100);
 write(a[i],' ');
 if a[i]=3 then num:=num+1;
end;

writeln;
writeln('Количество элементов равных трем ', num);
end.
```

1.3. Последовательность выполнения работы

1. Составить блок-схему (только для первого задания) и написать программу в среде PascalABC.NET (для первого и второго заданий).
2. В отчет включить формулировку задания, блок-схему, программу и скриншоты работы программы.
3. Для каждого задания сделать несколько скриншотов работы программы.

1.4. Варианты заданий

Вариант задания соответствует порядковому номеру в списке группы.

Задание 1

Определить массив целых чисел из 20 элементов. Заполнить массив случайными числами.

1. Найти максимальное значение среди положительных элементов массива.
2. Найти минимальное значение среди положительных элементов массива.
3. Найти максимальное значение среди отрицательных элементов массива.
4. Найти минимальное значение среди отрицательных элементов массива.
5. Найти сумму элементов массива, значения которых больше 5.
6. Найти сумму элементов массива, значения которых меньше 5.
7. Найти сумму элементов массива, значения которых четные числа.
8. Найти сумму элементов массива, значения которых нечетные числа.
9. Найти произведение элементов массива, значения которых больше 5.
10. Найти произведение элементов массива, значения которых меньше 5.
11. Найти произведение элементов массива, значения которых четные числа.
12. Найти произведение элементов массива, значения которых нечетные числа.
13. Найти максимальное значение среди четных элементов массива.
14. Найти минимальное значение среди нечетных элементов массива.
15. Найти максимальный элемент массива и заменить все элементы массива равные
максимальному значению на 0.
16. Найти минимальный элемент массива и заменить все элементы массива равные
минимальному значению на 0.

17. Найти количество положительных элементов массива. Если оно больше половины количества

элементов исходного значения, то заменить знак числа первого элемента массива.

18. Найти количество отрицательных элементов массива. Если оно меньше половины количества

элементов исходного значения, то заменить знак числа последнего элемента массива.

19. Найти количество четных элементов массива. Если оно больше половины количества

элементов исходного значения, то ко всем четным значениям элементов прибавить 1.

20. Найти количество нечетных элементов массива. Если оно меньше половины количества

элементов исходного значения, то ко всем нечетным значениям элементов прибавить 1.

Задание 2

Массив можно заполнить случайными числами или вводит значения с клавиатуры.

1. Даны действительные числа (a1901, a1902...a1950) - количество осадков (в миллиметрах), выпавших в Москве в течение первых 50 лет нашего столетия. Надо вычислить среднее количество осадков и отклонение от среднего для каждого года.

2. Даны действительные числа a1, …,a20. Получить числаb1, …,b20, где bi – среднее арифметическое всех членов последовательностиa1, …,a20, кроме a i (i=1, 2,. .., 20).

3. Даны натуральные числа n1, …,n20, действительные числа x1, …,x20.

$$\frac{n_1x_1 + \ldots + n_{20}x_{20}}{n_1 + \ldots + n_{20}}$$

Вычислить .

4. Даны действительные числа $a1$, …,an,$b1$, …,bn. Вычислить $(a1+bn)(a2+bn-1)...(an+b1)$.

5. Даны действительные числа $a1$, …,an. Если в результате замены отрицательных членов последовательности$a1$, …,an их квадратами члены будут образовывать неубывающую последовательность, то получить сумму членов исходной последовательности; в противном случае получить их произведение.

6. Даны целые числа *a*1, …,*a*99. Получить новую последовательность, выбросив из исходной все члены со значением max (*a*1, …,*a*99).

7. Даны натуральное число *m*, действительные числа *a*1, …, *a*30 (числа*a*1, …,*a*30 попарно различны, *m* ≤ 30). В последовательности *a*1 ,…,*a*30 поменять местами наибольший член и член с номером *m*.

8. Даны действительные числа *x*1 ,…,*x*101, *y*1 ,…,*y*101. Получить действительные *x*1′,...,*x*1′01 ,*y*1′,...,*y*1′01 ,преобразовав для получения *xi*′,*yi*′ члены *xi* ,*yi* по правилу: если они оба отрицательны, то каждый из них увеличить на 0.5; если отрицательно только одно число, то отрицательное число заменить его квадратом; если оба числа неотрицательны, то каждое из них заменить на среднее арифметическое исходных значений.

9. Даны действительные числа *a*1 ,…,*a*30. Получить:

а) max (a1+ a30, a2+ a29,…, a15+ a16);

б) min (a1a16, a2a17,…, a15a30)

10.Даны действительные числа*r*1,…,*r*17, среди которых заведомо есть как отрицательные, так и неотрицательные. Получить*x*1*y*1 + … + *xs ys* , где*x*1 ,…,*x p* – отрицательные члены последовательности*r*1,...,*r*17 , взятые в порядке их следования,*y*1,...,*yq* – неотрицательные члены, взятые в обратном порядке, *s* = min (*p*, *q*).

11.Даны целые числа *a*1 ,…,*a*20. Наименьший член последовательности *a*1 ,…,*a*20 заменить целой частью среднего арифметического всех членов, остальные члены оставить без изменения. Если в последовательности несколько членов со значением min (*a*1,…,*a*20), то заменить последний по порядку.

12.Даны действительные числа *a*1, …,*a*20 (все числа попарно различны). Поменять в этой последовательности местами:

а) наибольший и наименьший члены;

б) наибольший и последний члены.

13.Даны целые числа *a*1,…,*a*100. Получить новую последовательность из 100 целых чисел, заменяя *ai* нулями, если *ai* не равно max (*a*1,…,*a*100), и заменяя *ai* единицей в противном случае (*i*=1, …, 100).

14.Даны действительные числа *a*1, …,*a*26. Требуется домножить все члены последовательности*a*1, …,*a*26 на квадрат ее наименьшего члена, если*a*1 ≥ 0, и на квадрат ее наибольшего члена, если*a*1 < 0.

15.Даны натуральное число *n*, действительные числа*a*1,…,*a*n. Получить*b*1,…,*b*10, где *b*i равно сумме тех членов последовательности*a*1, …*a*n, которые принадлежат полуинтервалу (*i* – 1,*i*] (*i* = 1, …, 10). Если полуинтервал не содержит членов последовательности, то соответствующее *b*i положить равным нулю.

2. ЛАБОРАТОРНАЯ РАБОТА №2: РАБОТА С ДВУМЕРНЫМИ МАССИВАМИ

2.1. Цель работы

Научиться использовать двумерные массивы при решении задач. Освоить основные приемы работы с двумерными массивами.

2.2. Теоретическая часть

Двумерный массив - это одномерный массив, элементами которого являются одномерные массивы.

Другими словами, это набор однотипных данных, имеющий общее имя, доступ к элементам которого осуществляется по двум индексам. Наглядно двумерный массив удобно представлять в виде таблицы, в которой n строк и m столбцов, а под ячейкой таблицы, стоящей в i-й строке и j-м столбце понимают некоторый элемент массива a[i][j].

По-другому двумерный массив также называют матрицей, а в том случае, когда n=m (число строк равно числу столбцов) матрицу называют квадратной.

Для работы с двумерным массивом удобно использовать двойной цикл, где внешний цикл по i будет пробегать по всем строкам, а внутренний цикл по j будет для текущей строки i перебирать все ее элементы.

Пример работы с двумерным массивом:

```
var a: array [1..10, 1..10] of integer;
i,j:integer;
begin
for i:=1 to 10 do
 for j:=1 to 10 do
 a[i,j]:=Random(100);
end.
```

Индекс массива может быть целого, символьного или перечислимого типа. Пример:

```
var a:array [1..10, 'a'..'z'] of integer;
i,j:integer;
begin
a[1,'b']:=Random(100);
 writeln(a[1,'b']);
end.
```

Пример 1. Вычисление количества элементов равных нулю в квадратной матрице

Заполнение квадратной матрицы случайными элементами. Подсчет количества элементов квадратной матрицы равных нулю.

```
объявление двумерного массива размера 10 на 10
var mas:array[1..10, 1..10] of integer;
 i,j,count:integer;
begin
count:=0;
for i:=1 to 10 do
begin
 writeln;
 for j:=1 to 10 do
```

```
 begin
заполнение массива случайными числами
mas[i,j]:=random(100);

вывод на экран элементов массива
write(mas[i,j]:6);

подсчет количества элементов равных нулю
if mas[i,j]=0 then inc(count);

end
end;
writeln; writeln;
writeln('Количество элементов равных нулю - ', count);
end.
```

Пример 2. Вычисление количества элементов равных нулю на главной диагонали в квадратной матрице

Заполнение квадратной матрицы случайными элементами. Подсчет количества элементов на главной диагонали квадратной матрицы равных нулю.

```
var
mas:array[1..10, 1..10] of integer;
i,j,count:integer;
begin
count:=0;
for i:=1 to 10 do
begin
writeln;
for j:=1 to 10 do
begin
mas[i,j]:=random(100);
write(mas[i,j]:6);
end
end;
for i:=1 to 10 do
if mas[i,i]=0 then inc(count);
writeln; writeln;
writeln('Количество элементов на главной диагонали равных нулю - ', count);
end.
```

Для побочной диагонали выделенный красным фрагмент кода преобразуется к виду:

```
for i:=1 to 10 do
if mas[i,10-i+1]=0 then inc(count);
```

Пример 3. Поиск максимального элемента массива

Дан двумерный массив. Найти максимальный элемент массива.

```
var
mas:array[1..10, 1..10] of integer;
i,j,count:integer;
begin
count:=0;
for i:=1 to 10 do
begin
writeln;
for j:=1 to 10 do
begin
mas[i,j]:=random(100);
write(mas[i,j]:6);
end
end;
var max:integer;
max:=mas[1,1];
for i:=1 to 10 do
for j:=1 to 10 do
 if mas[i,j]>max then max:=mas[i,j];
writeln; writeln('max = ', max);
end.
```

2.3. Последовательность выполнения работы

1. Составить блок-схему (только для первого задания) и написать программу в среде PascalABC.NET (для первого и второго заданий).
2. В отчет включить формулировку задания, блок-схему, программу и скриншоты работы программы.
3. Для каждого задания сделать несколько скриншотов работы программы.

Вариант задания соответствует порядковому номеру в списке группы.

2.4. Варианты заданий

Задание 1.

1. Даны целые числа a1, a2... a15. Получить целочисленную матрицу $[b_{ij}]$ i, j = 1, 2, 15, для которой $b_{ij} = a_i - 3a_j$. Определить сумму положительных элементов побочной диагонали матрицы.
2. Даны действительные числа a1…a10, b1…b20. Получить действительную матрицу $|c_{ij}|$ i=1..20; j=1..10, для которой $c_{ij}=a_{ij}/(1+|b_i|)$. Определить произведение всех отрицательных элементов матрицы.
3. Получить $|a_{ij}|$ i=1..10, j=1..12 – целочисленную матрицу, для которой a_{ij} =i+2j. Определить сумму элементов матрицы.
4. Дано натуральное число n. Получить действительную матрицу $|a_{ij}|$, i,j = 1..n, для которой a_{ij}= 1/(i+j). Определить количество положительных элементов на главной диагонали.
5. Дана действительная квадратная матрица $|a_{ij}|_{i,j=1,..,n}$. Получить квадратную матрицу $|b_{ij}|_{i,j=1,..,n}$, для которой $b_{ij} = \begin{cases} a_{ij} \text{ при } j \geq i, \\ a_{ji} \text{ при } j < i. \end{cases}$ Определить сумму элементов, расположенных в нечетных столбцах.

6. Дана действительная квадратная матрица $\|a_{ij}\|_{i,j=1,..,n}$. Получить квадратную матрицу $\|c_{ij}\|_{i,j=1,...,n}$, для которой

$$c_{ij} = \begin{cases} a_{ij} & \text{при } j < i, \\ -a_{ij} & \text{при } j \geq i. \end{cases}$$

Определить сумму элементов, расположенных в четных строках.

7. Получить действительную квадратную матрицу $\|a_{ij}\|$, i,j=1..7, первая строка которой задается формулой a_{1j}=2j+3 (j=1..7) , вторая строка задается формулой a_{2j}=j-3/(2+1/j), (j=1..7) , а каждая следующая строка есть сумма двух предыдущих.

8. Дана действительная матрица размера 5x9. Получить одномерный массив, состоящий из средних арифметических элементов каждого из столбцов. Найти минимальный элемент одномерного массива.

9. Дана действительная матрица размера 5x9. Получить одномерный массив, состоящий из средних арифметических элементов каждого из столбцов, имеющих четные номера. Найти максимальный элемент одномерного массива.

10. Дана действительная матрица 5x6. Нужно все элементы матрицы преобразовать по правилу: отрицательные элементы заменить на -1, положительные – на 1, а нулевые оставить без изменения.

11. Подсчитать число отрицательных элементов в каждом столбце матрицы размера 5x6. Создать одномерный массив, содержащий подсчитанные значения. Определить номер столбца, содержащего наибольшее количество отрицательных элементов.

12. Подсчитать K - число положительных элементов в матрице размера 5x6 и отдельно F - число положительных элементов на главной и побочной диагонали. Найти разность K-F, если она

больше нуля, то у элементов в первом столбце поменять знак на противоположный.
13. Дана действительная матрица размера m*n. Определить числа b1…bm равные соответственно суммам элементов строк. Определить номер строки с минимальной суммой.
14. Найти максимальный элемент матрицы 3x4 и их количество.

Задание 2.

1. Дано натуральное число n. Получить действительную матицу $|a_{ij}|$ i,j=1…n, для которой

$$a_{ij} = \begin{cases} \sin(i+j) & \text{при } i < j \\ 1 & \text{при } i = j \\ \arcsin\dfrac{i+j}{2i+3j} & \text{в остальных случаях.} \end{cases}$$

2. Дано натуральное число n. Выяснить сколько положительных элементов содержит матрица |aij| i,j=1..n, если **a_{ij}=sin(i+j/2)**

3. Дано натуральное число n. Выяснить сколько положительных элементов содержит матрица |aij| i,j=1..n, если **a_{ij}=cos(i^2+n)**
4. Дана действительная матрица размера nxm. Получить новую матрицу путем деления всех элементов данный матрицы на ее наибольший по модулю элемент.
5. Дана действительная квадратная матрица размера 5x5. Заменить нулями все элементы, расположенные на главной диагонали и выше нее.

6. Дана целочисленная матрица размера 6x6. Найти максимальный элемент главной диагонали и наименьший элемент побочной диагонали.
7. Дана действительная матрица размера 9x7. Найти значение наибольшего по модулю элемента матрицы, а также индексы какого-нибудь элемента с найденным значением модуля.
8. Даны две целочисленные матрицы размера 3x2 и 2x3. Получить третью матрицу, равную произведению двух матриц.
9. Дана действительная матрица размера 6x9. Найти среднее арифметическое наибольшего и наименьшего значений ее элементов.
10. В данной действительной матрице размера 6x9 поменять местами строку, содержащую элемент с наибольшим значением, со строкой, содержащей элемент с наименьшим значением. Предполагается, что эти элементы единственны.

3. ЛАБОРАТОРНАЯ РАБОТА №3: ПЕРЕЧИСЛИМЫЙ И ДИАПАЗОННЫЙ ТИПЫ

3.1. Цель работы

Изучить понятия перечислимый и диапазонный типы. Освоить правила работы с переменными данных типов.

3.2. Теоретическая часть

Тип, задаваемый перечислением значений, называется **перечислимым**.

Переменные перечислимого типа хранятся в памяти, как целые числа. Первому значению соответствует 0, второму — 1, и т.д.

Диапазонный тип строится на базе

- целого
- символьного
- или перечислимого типов

type Marks = 2..5; Digits = '0'..'9'; Letters = 'a'..'z'; Summer = Jun..Aug;

Пример 1.

```
type
 Days = (Mon, Tue, Wen, Thi, Fri, Sat, Sun);
 Months = (Jan, Feb, Mar, Apr, May, Jun, Jul, Aug, Sep, Oct, Nov,
Dec);
var
 d: Days;
 m: Months;

begin
 d := Tue;
 m := Months.May;
 writeln(d,m);
end.
```

Пример 2.

```
for m: Months:=Jan to Dec do
 case m of
 Jan: writeln('Январь');
 Feb: writeln('Февраль');
 else writeln('Забыли');
 end;
```

Пример 3.

```
type
 Days = (Mon, Tue, Wen, Thi, Fri, Sat, Sun);
var
 d: Days;
begin
 d := Tue;
{возвращает следующее значение перечислимого типа}
 writeln(Succ(d)); {Wen}
{возвращает предыдущее значение перечислимого типа}
 writeln(Pred(d)); {Mon}
 {возвращает порядковый номер текущего значения перечислимого типа}
 writeln(Ord(d)); {1}
end.
```

Пример 4. Применение диапазонного типа для массива

```
var a:array [1..10, 'a'..'z'] of integer;
i,j:integer;
begin
a[1,'b']:=Random(100);
 writeln(a[1,'b']);
end.
var
 i:'a'..'z';
 j:integer;
begin
 var a:array [1..4, 'a'..'d'] of integer;
 for j:=1 to 4 do
 for i:='a' to 'd' do
 begin
 a[j,i]:=random(10);
 write(i,' = ',a[j,i]:2,' ');
 end
end.
```

Пример 5.Словесное наименование сезона по номеру месяца

```
var Month: integer;
 Season: string;
 begin
 write('Введите номер месяца: ');
 readln(Month);
 case Month of
 1,2,12: Season := 'Зима';
 3..5: Season := 'Весна';
 6..8: Season := 'Лето';
 9..11: Season := 'Осень';
 else Season := 'Вы ввели неверный номер месяца';
 end;
 writeln('Это ',Season)
end.
```

3.3. Варианты заданий

Вариант 1

Определить два перечислимых типа: месяцы и сезоны. Нужно составить программу, определяющую на какой сезон приходится данный месяц.

Вариант 2

Определить два перечислимых типа: страны и столицы. Нужно составить программу, определяющую какой стране относится указанная столица.

Вариант 3

Задать два перечислимых типа: первый фамилии целиком, второй - первые буквы фамилий. Вводится первая буква фамилии нужно вывести фамилию целиком.

Вариант 4

Задать два перечислимых типа: первый коды цветов, второй - названия цветов. Вводится имя цвета, выводится его код.

Вариант 5

Объявить два диапазонных типа: один буквы русского алфавита, второй буквы английского алфавита. Вводится буква с клавиатуры - выводится к какому алфавиту она принадлежит и ее номер в алфавите.

Вариант 6

Объявить одномерный массив из 20 элементов целого типа, в котором в качестве индекса массива выступает диапазонный тип английских букв. Необходимо заполнить элементы массива, вывести на экран как индекс, так и значение. Найти максимальный элемент массива и указать какому индексу он соответствует.

Вариант 7

Символ вводится с клавиатуры. Используя диапазонные типы определить и вывести на экран - это цифра, малая буква анг. алфавита, большая буква анг. алфавита, малая буква русского алфавита, большая буква русского алфавита, не буква.

Вариант 8

Задать два перечислимых типа: первый коды регионов, второй - названия регионов. Вводится код региона, выводится его название.

Вариант 9

Объявить два диапазонных типа: один буквы русского алфавита, второй буквы английского алфавита. Вводится буква с клавиатуры - выводится к какому алфавиту она принадлежит и ее номер в алфавите.

Вариант 10

Объявить одномерный массив из 20 элементов целого типа, в котором в качестве индекса массива выступает диапазонный тип английских букв. Необходимо заполнить элементы массива, вывести на экран как индекс, так и значение. Найти максимальный элемент массива и указать какому индексу он соответствует.

Вариант 11

Символ вводится с клавиатуры. Используя диапазонные типы определить и вывести на экран - это цифра, малая буква анг. алфавита, большая буква анг. алфавита, малая буква русского алфавита, большая буква русского алфавита, не буква.

Вариант 12

Определить два перечислимых типа: месяцы и сезоны. Нужно составить программу, определяющую на какой сезон приходится данный месяц.

Вариант 13

Определить два перечислимых типа: страны и столицы. Нужно составить программу, определяющую какой стране относится указанная столица.

Вариант 14

Задать два перечислимых типа: первый фамилии целиком, второй - первые буквы фамилий. Вводится первая буква фамилии нужно вывести фамилию целиком.

Вариант 15

Задать два перечислимых типа: первый коды цветов, второй - названия цветов. Вводится имя цвета, выводится его код.

Вариант 16

Объявить два диапазонных типа: один буквы русского алфавита, второй буквы английского алфавита. Вводится буква с клавиатуры - выводится к какому алфавиту она принадлежит и ее номер в алфавите.

Вариант 17

Объявить одномерный массив из 20 элементов целого типа, в котором в качестве индекса массива выступает диапазонный тип английских букв. Необходимо заполнить элементы массива, вывести на экран как индекс, так и значение. Найти максимальный элемент массива и указать какому индексу он соответствует.

Вариант 18

Определить два перечислимых типа: месяцы и сезоны. Нужно составить программу, определяющую на какой сезон приходится данный месяц.

4. ЛАБОРАТОРНАЯ РАБОТА №4: СТРОКИ. МНОЖЕСТВА

4.1. Цель работы

Изучить понятия множества и строки. Освоить правила работы с переменными данных типов.

4.2. Задание к лабораторной работе

Символы

c : char

Пример

```
var
 c: char;
 a:byte;
begin
readln(c);
writeln(ord(c)); {возвращает код символа}
a:=120;
writeln(chr(a)); {возвращает символ по коду}
writeln(UpperCase(c)); {преобразует символ в заглавную букву}
```

```
writeln(LowerCase(c)); {преобразует символ в прописную букву}
if (char.IsDigit(c)) then writeln('цифра'); {проверка, что символ - это цифра}
if (char.IsLetter(c)) then writeln('буква'); {проверка, что символ - это буква}
if (char.IsLetterOrDigit(c)) then writeln('буква или цифра'); {проверка, что символ - это буква или цифра}
end.
```

Строки

st : string

Строка - это одномерный массив типа char.

Пример

```
var
 c: char;
 st:string;
begin
 readln(st); {ввод строки}
 c:=st[3]; {переменной с присваивается третья буквы строки st}
 writeln(c);
end.
```

Пример

```
var
 st, st1, st2:string;
begin
 readln(st);
 readln(st1);
 st2:=st+' '+st1; {конкатенация строк}
 writeln(st2);
{сравнение строк}
 if st>st1 then writeln('st>st1') else writeln('st<st1');
end.
```

Подпрограммы работы со строками

- Length(s) - функция, возвращающая длину строки s

- SetLength(s,n) - процедура, устанавливающая длину строки s равной n
- Copy(s,from,len) - функция, возвращающая подстроку s с позиции from длины len
- Insert(subs,s,form) - процедура, вставляющая подстроку subs в строку s с позиции from
- Delete(s,from,len) - процедура, удаляющая из строки s len символов с позиции from
- Pos(subs,s) - функция, возвращающая позицию первого вхождения подстроки subs в строку s или 0, если подстрока не найдена
- PosEx(subs,s,from=1) - функция, возвращающая позицию первого вхождения подстроки subs в строку s, начиная с позиции from, или 0, если подстрока не найдена
- TrimLeft(s) - функция, возвращающая строку s, в которой удалены все начальные пробелы
- TrimRight(s) - функция, возвращающая строку s, в которой удалены все заключительные пробелы
- Trim(s) - функция, возвращающая строку s, в которой удалены все удаляет все начальные и заключительные пробелы
- Преобразование строка ↔число
- Str(x,s) - процедура, преобразующая целое или вещественное выражение x к строковому представлению и записывающая результат в строку s
- Val(s,x,errcode) - процедура, преобразующая строку s к целому или вещественному значению и записывающая результат в целую или вещественную переменную x. Переменная errcode - целая; если преобразование невозможно, то в errcode содержится номер первого символа, вызвавшего ошибку

- IntToStr(i) - функция, преобразующая целое x в строку
- StrToInt(s) - функция, преобразующая строку s к целому; может генерировать исключение
- FloatToStr(i) - функция, преобразующая вещественное x в строку
- StrToFloat(s) - функция, преобразующая строку s к вещественному; может генерировать исключение

Множества

В множестве элементы не могут повторяться.
Объявление множества
```
m : set of integer;
const n=30;
var m:set of 1..n;
 m1: set of 'a'..'z';
 m2:set of (winter, spring, summer, outumn)
```

Добавление элемента в множество
```
Include(m, 5);
```

Исключение элемента из множества
```
Exclude(m,3);
```

Вывод элементов множества
```
foreach i in m do writeln(i);
```

Проверка на принадлежность элемента множеству
```
if (5 in m) then writeln('включено');
```

Объединение множеств
```
m:=m+m1;
```

Пересечение m1*m2
Разность s1-s2
Проверки на вложение множеств
```
m1<m2
m1>m2
```

Пример на множества

```
type num = set of 0 ..9;
digit = set of '0' .. '9';
var s1, s2, s3 : digit;
 s4, s5, s6 : num;
begin
 s1 := ['1', '2', '3'];
 s2 := ['2', '1', '3'];
 s3 := ['1', '3'];
 s4 := [0..3];
 s5 := [4 .. 6];
 s6 := [3, 5 .. 9];
end.
```

4.3. Варианты заданий

Вариант 1

1. Дано слово s1. Получить слово s2, образованное нечетными буквами слова s1.
2. Дано предложение. Найти сумму цифр, встречающихся в этом предложении и вывести их на экран.
3. Дано 2 предложения. Составить два множества: первое содержит слова из первого предложения, а второе - из второго предложения. Вывести элементы каждого множества на экран. Найти их пересечение.

Вариант 2

1. Дано слово s. Получить слово t, получаемое путем прочтения слова s начиная с его конца.
2. Дано предложение. Найти произведение цифр, встречающихся в этом предложении и вывести их на экран.
3. Дано два одномерных массива. Составить два множества: первое множество содержит элементы первого массива, а второе - второго массива. Вывести элементы каждого множества на экран. Найти объединение этих множеств.

Вариант 3

1. Дано слово. Добавить к нему в начале четыре символа "+" и в конце - пять символов "-".
2. Дано предложение. Найти максимальную цифру, среди цифр встречающихся в этом предложении и вывести их на экран.

3. Используя множество посчитать количество различных чисел встречающихся в одномерном массиве.

Вариант 4

1. Дано слово. Добавить к нему в начале и конце столько звездочек, сколько букв в этом слове.
2. Дано предложение. Найти минимальную цифру, среди цифр встречающихся в этом предложении и вывести их на экран.
3. Организовать ввод символов с клавиатуры, до тех пор пока не будет нажат символ "#". Вводимые символы добавляются в множество. Вывести на экран из данного множества только цифры и посчитать их количество. Сформировать второе множество, в которое попадут все четные цифры из первого множества, вывести его на экран и посчитать количество.

Вариант 5

1. Дано предложение. Составить программу, которая печатает столбиком все вхождения в предложение некоторого символа и считает их количество.
2. Дан текст, имеющий вид "d1+d2+d3+...+dn", где di - цифры. Вычислить записанную в тексте сумму и количество слагаемых.
3. Дано множество целых чисел. На его основе построить два множества: в первое множество входят числа которые делятся на 4, а во второе - на 2. Найти пересечение полученных множеств.

Вариант 6

1. Дано предложение. Определить долю (в %) букв "a" в нем.
2. Дан текст, имеющий вид "d1-d2+d3-...+dn", где di - цифры. Вычислить записанную в тексте сумму и количество слагаемых.
3. Организовать ввод символов с клавиатуры, до тех пор пока не будет нажат символ "%". Вводимые символы добавляются в множество. Вывести на экран из данного множества только буквы английского алфавита и посчитать их количество. Сформировать второе множество, в которое попадут все гласные буквы английского алфавита из первого множества, вывести его на экран и посчитать количество.

Вариант 7

1. Дано предложение. Определить сколько в нем одинаковых соседних букв и вывести какие это буквы.
2. Дано предложение. Найти наибольшее количество идущих подряд цифр и вывести их на экран.
3. Дано слово. С помощью множеств посчитать количество гласных и согласных букв русского алфавита (повторяющиеся буквы считать за одну).

Вариант 8

1. Дано предложение. Вывести столбиком все его буквы "и", стоящие на четных местах и посчитать их количество.
2. Дано предложение. Найти наименьшее количество идущих подряд цифр и вывести их на экран.
3. Используя множества, удалить из слова повторяющиеся символы.

Вариант 9

1. Дано предложение. Определить сколько в нем гласных букв и вывести их на экран.
2. Дан текст, имеющий вид "d1*d2*d3*...*dn", где di - цифры. Вычислить записанное в тексте произведение и количество множителей.
3. Символы вводятся с клавиатуры до тех пор пока не будет нажата клавиша ".". С помощью множеств необходимо проверить какие и сколько символов из введенных принадлежат диапазону от "a" до "f".

Вариант 10

1. Дано предложение. Найти порядковый номер первой и последней буквы "е" в нем. Посчитать количество букв "е".
2. Дан текст, имеющий вид "x+d1+d2+..+dn=y", где где di b y - цифры. Найти x.
3. Символы вводятся с клавиатуры до тех пор пока не будет введен символ "+". С помощью множеств вывести из данной последовательности на экран символы, которые не являются буквами и цифрами, посчитать их количество.

Вариант 11

1. Дано слово. Определить является он перевертышем или нет (перевертыш - слово которое одинаково читается в двух направлениях).

2. Дан текст, имеющий вид "x-d1-d2-..-dn=y", где где di b y - цифры. Найти x.
3. Дано 2 предложения. Составить два множества: первое содержит слова из первого предложения, а второе - из второго предложения. Вывести элементы каждого множества на экран. Найти их пересечение.

Вариант 12

1. Дано предложение. Определить количество букв "и", предшествующих первой запятой предложения.
2. Дан текст, имеющий вид "x*d1*d2*..*dn=y", где где di b y - цифры. Найти x.
3. Дано два одномерных массива. Составить два множества: первое множество содержит элементы первого массива, а второе - второго массива. Вывести элементы каждого множества на экран. Найти объединение этих множеств.

Вариант 13

1. Дано предложение. Напечатать все символы, расположенные между первой и второй запятой, посчитать их количество. Если второй запятой нет, то должны быть напечатаны все символы, расположенные после первой запятой и до конца предложения.
2. Дано предложение. Найти длину самого короткого слова.
3. Дано множество целых чисел. На его основе построить два множества: в первое множество входят числа которые делятся на 4, а во второе - на 2. Найти пересечение полученных множеств.

Вариант 14

1. Дано предложение. Заменить в нем все вхождения буквосочетания "да" на "не".
2. Дано предложение. Найти длину самого длинного слова.
3. Организовать ввод символов с клавиатуры, до тех пор пока не будет нажат символ "%". Вводимые символы добавляются в множество. Вывести на экран из данного множества только буквы английского алфавита и посчитать их количество. Сформировать второе множество, в которое попадут все гласные буквы английского алфавита из первого множества, вывести его на экран и посчитать количество.

Вариант 15

1. Дано предложение. Удалить из него все буквы "с". Посчитать количество удаленных букв.
2. Дано предложение. Напечатать в столбик все слова из этого предложения и посчитать их количество.
3. Дано слово. С помощью множеств посчитать количество гласных и согласных букв русского алфавита (повторяющиеся буквы считать за одну).

Вариант 16

1. Дано предложение. Удалить из него все буквы "о", стоящие начетных местах. Посчитать количество удаленных букв.
2. Дано предложение. Найти какое-нибудь его слова начинающиеся на букву "к" и напечатать это слово.
3. Используя множества, удалить из слова повторяющиеся символы.

Вариант 17

1. Дано слово. Вставить заданную букву после первой буквой "и".
2. Дано предложение. Определить количество слов в предложении, начинающихся с буквы "а".
3. Символы вводятся с клавиатуры до тех пор пока не будет нажата клавиша ".". С помощью множеств необходимо проверить какие и сколько символов из введенных принадлежат диапазону от "a" до "f".

Вариант 18

1. Дано слово. Вставить заданную букву перед последней буквой "и".
2. Дано предложение. Напечатать все слова этого предложения, длина которых не более 6 символов. Посчитать их количество.
3. Дано 2 предложения. Составить два множества: первое содержит слова из первого предложения, а второе - из второго предложения. Вывести элементы каждого множества на экран. Найти их пересечение.

5. ЛАБОРАТОРНАЯ РАБОТА №5: ПОДПРОГРАММЫ

5.1. Цель работы

Изучить понятие подпрограмм, их назначение. Уметь структурировать программу с помощью выделения множества подпрограмм. Понимать отличия функций от процедур.

5.2. Теоретическая часть

Подпрограмма - это вспомогательный алгоритм, который используется для реализации другого алгоритма.

Вспомогательные алгоритмы имеют:

- имя
- список параметров (некоторые параметры являются входными, а некоторые — выходными).

Если один из выходных параметров возвращается особым образом, так что алгоритм можно использовать в выражении, то такой алгоритм называется алгоритмом-функцией.

В программировании вспомогательные алгоритмы называются подпрограммами и делятся на:

- процедуры
- функции

Процедуры

```
procedure NameProc(a,b:integer);
begin
тело процедуры
end;
```

Вызов процедуры

```
Begin
 var x,y:integer;
 NameProc(3,2);
 NameProc(x,y);
end.
```

Если необходимо, чтобы в теле процедуры переменная изменила свое значение, то необходимо ее описать с ключевым словом var.

Пример

```
var x,y:integer;
procedure nameProc(x:integer; var y:integer);
begin
```

```
 x:=x+2; y:=y+3;
end;
begin
 x:=3; y:=5;
nameProc(x,y);
 writeln('x= ',x); writeln('y= ',y);
end.
```

Результат работы программы

```
x = 3
y = 8
```

Функции являются другой разновидностью подпрограмм.

Это подпрограмма, возвращающая одно значение особым образом так, что ее вызов можно использовать в выражении.

Пример

```
function Stepen(x,y:integer):integer;
var i,p:integer;
begin
 p:=1;
 for i:=1 to y do
 p:=p*x;
 Result:=p;
end;
begin
writeln(Stepen(3,4));
end.
```

Переменные, объявленные внутри подпрограммы, считаются локальными и не будут видны после выхода из подпрограммы.

Досрочное завершение подпрограммы - оператор exit

В программе можно описать несколько подпрограмм с одним именем, но разными типами или количеством формальных параметров.

Пример

```
procedure Swap(var x,y:integer);
var d:integer;
begin
 d:=x;
 x:=y;
 y:=d;
end;
```

```
procedure Swap(var x,y:real);
var d:real;
begin
 d:=x;
 x:=y;
 y:=d;
end;
begin
 var a,b:integer;
 var n,m:real;
 a:=2; b:=5;
Swap(a,b);
 writeln('a = ',a,' b = ',b);
 n:=2.1; m:=5.4;
 Swap(n,m);
 writeln('n = ',n,' m = ',m);
end.
```

Передача в подпрограмму массива

```
const n=10;
type
 mas=array[1..n] of integer;
var
 a:mas;

procedure InitMas(var x:mas);
begin
 for var i:=1 to n do
 x[i]:=Random(30)-10;
end;

procedure PrintMas(x:mas);
begin
 writeln;
 for var i:=1 to n do
 write(x[i]:4);
 writeln;
end;

function NumZero(x:mas):integer;
begin
```

```
 Result:=0;
 foreach var i:integer in x do
 if i=0 then inc(Result);
end;

begin
InitMas(a);
 PrintMas(a);
 writeln('Количество элементов = ',NumZero(a));
end.
```

5.3. Варианты заданий

При решении задач реализовать текстовое меню, которое всегда будет выводиться на экран. В программе обязательно должны быть использованы как процедуры, так и функции, у которых должны быть передаваемые параметры.

1. Четырехугольник задан своими сторонами a,b,c,d и диагональю r. Вычислить площадь четырехугольника с использованием функции нахождения площади треугольника.
2. Треугольник задан своими координатами своих вершин. Вычислить площадь треугольника с использованием функции нахождения расстояния между двумя точками.
3. Найти минимальное среди 4-х чисел, используя функцию нахождения минимального среди двух чисел.
4. Составить подпрограмму, которая проверяет заданный массив из N чисел, на упорядоченность по возрастанию(k=1 - если массив упорядочен, k=0 - иначе). С её помощью в основной программе обработать два массива и тот, который из них не упорядочен - обнулить(с помощью другой подпрограммы).
5. Четыре точки заданы своими координатами X(x1, x2), Y(y1, y2), Z(z1, z2), P(p1, p2). Выяснить, какие из них находятся на максимальном расстоянии друг от друга и вывести на печать значение этого расстояния. Вычисление расстояния между двумя точками оформить в виде функции.
6. Задана окружность $(x-a)^2 + (y-b)^2 = R^2$ и точки P(p1, p2), F(f1, f1), L(l1,l2). Выяснить и напечатать, сколько точек лежит внутри окружности. Проверку, лежит ли точка внутри окружности, оформить в виде функции.
7. Заданы матрицы A и B третьего порядка. Переменной S присвоить -1, если максимальный элемент матрицы A больше максимального элемента матрицы B; 0 если

максимальные элементы матрицы равны; 1, если максимальный элемент матрицы A меньше максимального элемента матрицы B. Поиск максимального элемента оформить в виде функции.

8. Даны отрезки a, b, c и d. Для каждой тройки этих отрезков, из которой можно построить треугольник, вывести на экран площадь данного треугольника. Проверку существования треугольника оформить в виде функции.
9. Даны две матрицы A и B. Написать программу, меняющую местами максимальные элементы этих матриц. Нахождение максимального элемента матрицы оформить в виде функции.
10. Вычислить $C_n^k = \frac{n!}{k!\,(n-k)!}$. Вычисление факториала числа оформите в виде подпрограммы.
11. Даны действительные числа a, b. Получить r = max (a, b + a), d = max (ab, a + b), s = max (r + d2, 3.14).
12. Даны действительные числа a, b. Получить u = min(a, b-a), y = min(ab, a+b), k = min(u+v2, 3.14). Поиск минимального элемента оформить в виде функции.
13. Даны натуральные числа a, b, c. Определить функцию bin(x), переводящую число x из десятичной системы счисления в двоичную. Найти bin(a + b), bin(ab + c).
14. Даны два натуральных числа a, b. Вычислить $\frac{a!!-ab}{a!!+ab}$. Функция x!! Определяется следующим образом:
    x!! = 1*3*5*...*x, если x нечетно,
    x!! = 2*4*6*...*x, если x четно.
15. Даны действительные числа a, b, c. Получить $\frac{\min(a,a+b)+\min(a,b+c)}{1+\min(a+bc,b)}$
16. Даны действительные числа a, b. Получить r = max (a, b + a), d = max (ab, a + b), s = max (r + d^2, 3.14).
17. Даны натуральные числа a, b, c. Определить функцию bin (x), переводящую число x из десятичной системы счисления в двоичную. Найти двоичное представление эти чисел.

6. ЛАБОРАТОРНАЯ РАБОТА №6: ЗАПИСИ

6.1. Цель работы

Изучить понятие записи. Освоить правила работы с записями и их назначение.

6.2. Теоретическая часть

Запись позволяет объединить разнородные элементы.

Объявление типа запись

```
type <имя записи>= record
 <поля записи>
end;
```

В качестве полей записей могут выступать как переменные, так и методы.

```
type
 SexType = (male, female);
 Person = record
 Name: string;
 Age, Weight: integer;
 Sex: SexType;
 end;
```

Инициализация записи:

```
var p: Person := (Name: 'Иванов'; Age: 20; Weight: 80; Sex: male);
```

или

```
var p: Person;
begin
p.Age:=20;
 p.Name:='Иванов';
end.
```

Операция присваивания для записей:

```
var p,p1: Person;
begin
 p.Age:=20;
 p.Name:='Иванов';
p1:=p;
end.
```

Для того чтобы при обращении к полям записи не писать каждый раз префикс (p.) можно использовать оператор with.

```
begin
 with p do
```

```
begin
 write('Введите имя '); readln(Name);
 write('Введите возраст '); readln(Age);
end;
 writeln('Вас зовут ',p.Name,'. Вам ',p.Age,'года\лет');
end.
```

Также можно объявить массив записей. Например:

```
var list:array[1..10] of Person;
 i:integer;
begin
for i:=1 to 10 do
 begin
 readln(list[i].Name);
 readln(list[i].Age);
 end;
end.
```

или

```
var list:array[1..10] of Person;
 i:integer;
begin
for i:=1 to 10 do
 with list[i] do
 begin
 readln(Name);
 readln(Age);
 end;
end.
```

Записи в качестве параметров можно передавать как в процедуры, так и функции. Функции также могут возвращать запись. Например:

```
type
 SexType = (male, female);
 Person = record
 Name: string;
 Age, Weight: integer;
 Sex: SexType;
```

```
 end;

var list:array[1..10] of Person;
 i:integer;

procedure add(var p:Person);
begin
 with p do
 begin
 readln(Name);
 readln(Age);
 end;
end;

procedure print(p:Person);
begin
 with p do
 writeln(Name,' - ',Age);
end;

begin
 for i:=1 to 1 do
 add(list[i]);
 for i:=1 to 1 do
 print(list[i]);
end.
```

Пример 1. Организация меню для консольного приложения

```
procedure menu; forward;

procedure add;
var i:byte;
begin
 writeln;
 writeln('add');
 sleep(2000);
 writeln;
 menu;
end;

procedure menu;
```

```
var x:byte;
begin
writeln('1-add');
writeln('2-exit');
readln(x);
case x of
1:add;
2:exit;
end;
end;

begin
menu;
end.
```

Пример 2. Удаление элемента, выделенного в ListBox

```
uses FormsABC;
var l:listbox;
 a:array[1..10] of integer;
 b:button;

procedure del;
var i:byte;
begin
for i:=l.SelectedIndex+1 to 10-1 do
 a[i]:=a[i+1];
a[10]:=0;
l.Items.Clear;
for i:=1 to 10 do l.Items.Add(a[i]);
end;

begin
MainForm.SetSize(300,300); //устанавливаем размер формы
MainForm.CenterOnScreen; // устанавливаем, чтобы форма выводилась по центру
l:=new ListBox();
for var i:=1 to 10 do
begin
 a[i]:=random(50);
 l.Items.Add(a[i]);
end;
```

```
b:=new Button('Del');
b.Click+=del;
end.
```

6.3. Последовательность выполнения работы

Создать программу, которая хранит базу данных в виде одномерного массива. Тип элементов массива - запись.

Для данной базы данных необходимо реализовать подпрограммы:

- добавить запись в базу
- удалить запись из базы
- вывод всех элементов базы
- поиск по ключевому полю.

В программе обязательно должны быть подпрограммы в виде функций и процедур.

Для получения не более 7 балов за лаб. раб. достаточно создать консольное приложение, организовав диалог с пользователем с помощью операторов read и write.

Для получения от 8 до 10 балов необходимо создать windows приложение, где интерфейс реализован в виде формы и управляющих элементов.

6.4. Варианты заданий

При поиске предусмотреть, что в базе могут быт одинаковые записи.

Поиск должен происходить по каждому критерию отдельно.

1. База студентов. О каждом студенте хранится: номер его зачетки, фамилия, имя, группа, год рождения. Организовать поиск по фамилии и номеру зачетки.
2. База автомобилей. О каждой машине хранится: номер машины, марка, цвет, пробег, год выпуска. Организовать поиск по марке и году выпуска .
3. База книг. О каждой книге хранится: номер книги, автор, название, год издания, количество страниц. Организовать поиск по автору и году издания.
4. База лекарств. О каждом лекарстве: название, год выпуска, назначение, вес, тип (таблетки, микстура, ...). Организовать поиск по названию и назначению.
5. База вин. О каждом вине хранится тип вина (сухое, полусладкое, ...), название, цвет, год урожая. Организовать поиск по названию и типу.

6. База оборудования. Об оборудовании тип, наименование, стоимость, состояние (отличное, хор, удовл). Организовать поиск по типу и наименованию.
7. База сотрудников. О каждом сотруднике фамилия, имя, пол, паспорт, год рождения. Поиск по фамилии и году рождения.
8. База фильмов. О каждом фильме: название, год выпуска, страна, продолжительность. Поиск по названию и году выпуска.
9. База программ. О каждой программе: название, назначение, тип (системная, прикладная, инструментальная), год выпуска. Поиск по названию и году выпуска.
10. База продуктов. О каждом продукте: наименование, тип (молочный, мясной...), стоимость, вес. Поиск по наименованию и стоимости.
11. База кафе (ресторанов). О каждом кафе: название, адрес (название улицы), уровень, год открытия. Поиск по названию и адресу.
12. База писем. О каждом письме: фамилия получателя, фамилия отправителя, адрес отправителя, адрес получателя, дата получения. Поиск по фамилии отправителя и дате получения.
13. База пользователей форума. О каждом пользователе: логин, пароль, фамилия, дата регистрации, дата последнего входа. Поиск по фамилии и дате регистрации.

Вы можете придумать свою предметную область, если Вас не устраивает приведенный перечень.

7. ЛАБОРАТОРНАЯ РАБОТА №7: РАЗРАБОТКА ПРОГРАММЫ С ГРАФИЧЕСКИМ ИНТЕРФЕЙСОМ С ИСПОЛЬЗОВАНИЕМ ПРОЦЕДУР И ФУНКЦИЙ

7.1. Цель работы

Научиться создавать программы с интуитивно понятным интерфейсом.

7.2. Теоретическая часть

Для создания элементов интерфейса необходимо подключить модуль

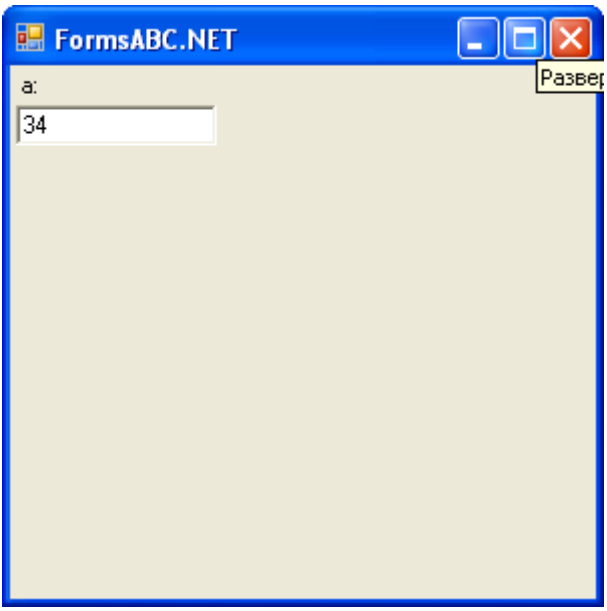
uses FormsABC;

Создание элементов интерфейса

Поле, в которое можно вводить только целые числа - IntegerField

```
uses FormsABC;
var a:IntegerField;
begin
 a := new IntegerField('a:');
end.
```

Результат работы программы



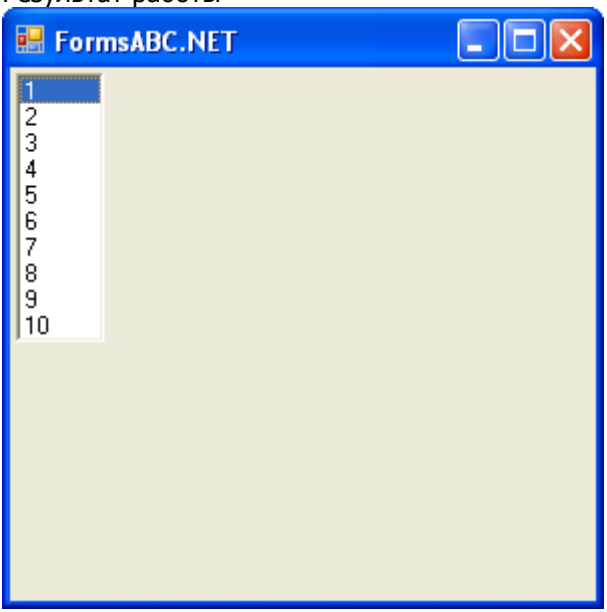
- Поле, в которое можно вводить только целые и дробные числа - RealField
- Поле, в которое можно вводить строку - Field
- Кнопка - Button

Создание списка и работа с ним

```
var l:Listbox;
begin
```

```
l:=new ListBox(); //создание списка
l.Items.Clear(); // удаление всех элементов списка
for var i:=1 to 10 do l.Items.Add(i); // добавление в список элементов
end.
```

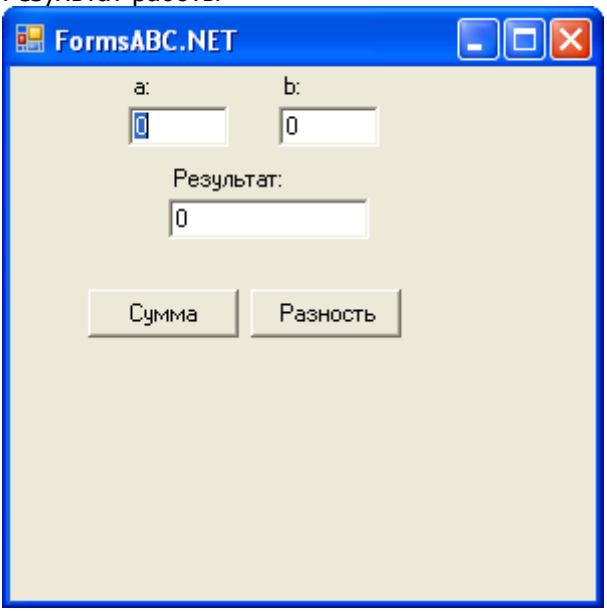
Результат работы



Пример. Создание простого калькулятора, позволяющего складывать и вычитать только целые числа

```
uses FormsABC;
var a,b,c:IntegerField;
 bt1, bt2:Button;
begin
 EmptySpace(50); a := new IntegerField('a:',50);
 EmptySpace(10); b := new IntegerField('b:',50);
 LineBreak;
 EmptySpace(70);  c := new IntegerField('Результат:',100);
 LineBreak;
 EmptyLine(10);
 EmptySpace(30);  bt1:=new  Button('Сумма');  bt2:=new  Button('Разность');
end.
```

Результат работы



Комментарий к примеру
Функция EmptySpace - добавляет пустое пространство
Фукнция LineBreak - переход на новую строку
Функция EmptyLine - добавляет пустую строку

Для того чтобы калькулятор стал считать необходимо привязать обработчики событий на нажатие кнопки "Сумма" и "Разность". Для этого необходимо написать две процедуры: Add, Diff.

```
procedure Add;
begin
 c.Value:=a.Value+b.Value;
end;

procedure Diff;
begin
 c.Value:=a.Value-b.Value;
end;
```

Запись a.Value означает получить значение, которое пользователь ввел в поле "a:".

Запись c.Value означает поместить результат в поле "Результат:", чтобы он стал виден пользователю.

Для того чтобы, эти функции вызывались их необходимо привязать к кнопкам:

после строчек создания кнопок
```
bt1:=new Button('Сумма');
bt2:=new Button('Разность');
```

необходимо добавить
```
bt1.Click+=Add;
bt2.Click+=Diff;
```

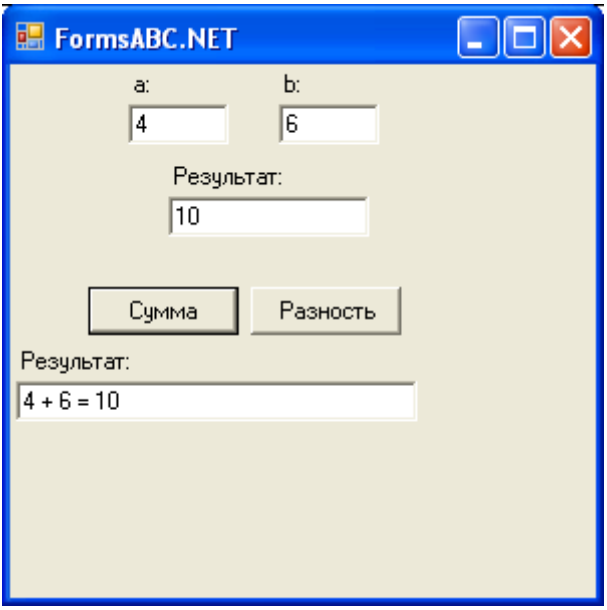
Доработаем калькулятор таким образом, чтобы он не просто выводит ответ в краткой форме, а выводил его в полной форме - в виде "4+5=9". Для этого:

Объявим переменную
```
txt:Field;
```

В основной программе создадим поле для строки:
```
txt := new Field('Результат:', 200);
```

В процедуре Add добавим строку:
```
txt.Text:=a.Text+' + '+b.Text+' = '+c.Text;
```

Аналогичную строчку необходимо добавить в процедуру Diff.
После данных изменений результат работы программы будет следующий:



7.3. Варианты заданий

Вариант 1

1. Напишите функции вычисления длины окружности (CircleLength) и площади круга (CircleSquare) по заданному радиусу. Создайте форму с полями ввода и кнопками. Организуйте работу формы: при нажатии на кнопку "Длина" выводится результат длины окружности, а на кнопку "Площадь" соответственно площадь круга. Вывод результата организовать как в краткой, так и в полной форме.

Вариант 2

1. Напишите функции вычисления расстояния между двумя точками, заданными своими координатами (Distance) и нахождения координаты середины между двумя точками(Middle). Создайте форму с полям ввода и кнопками. Организуйте работу формы: при нажатии на кнопку "Расстояние" выводится результат расстояния между точками, а на кнопку "Середина" соответственно координаты середины. Вывод результата организовать как в краткой, так и в полной форме.

Вариант 3

1. Напишите функцию, вычисляющую площадь треугольника, заданного координатами своих вершин (TriangleSquare), используя формулу Герона. Для вычисления напишите вспомогательную функцию Distance, которая находит расстояние между двумя точками. Создайте форму с полям ввода и кнопками. Организуйте работу формы: при нажатии на кнопку "Расстояние" выводится результат расстояния между точками, а на кнопку "Площадь" соответственно площадь треугольника. Вывод результата организовать как в краткой, так и в полной форме.

Вариант 4

1. Напишите функции, вычисляющую возраст человека в годах по заданному возрасту в месяцах и определяющую пора человеку получать паспорт или нет. Создайте форму с полям ввода и кнопками. Организуйте работу формы: при нажатии на кнопку "Возраст" выводится возраст в годах, а на кнопку "Паспорт" соответственно пора или нет получать паспорт. Вывод результата организовать как в краткой, так и в полной форме.

Вариант 5

1. Напишите функции, определяющую, сколько месяцев осталось до дня рождения человека, если известен его возраст в месяцах, и определяющую исполнилось человеку уже 18 лет или нет. Создайте форму с полям ввода и кнопками. Организуйте работу формы: при нажатии на кнопку "ДР" выводится сколько месяцев осталось до ДР, а на кнопку "Совершеннолетие" соответственно есть 18 лет или нет. Вывод результата организовать как в краткой, так и в полной форме.

Вариант 6

1. Напишите функции, вычисляющие первую и вторую цифры заданного двузначного числа. Создайте форму с полям ввода и кнопками. Организуйте работу формы: при нажатии на кнопку "Первая" выводится первая цифра, а на кнопку "Вторая" соответственно вторая. Вывод результата организовать как в краткой, так и в полной форме.

Вариант 7

1. Напишите функции, вычисляющую последнюю цифру заданного числа и определяющую четное число или нет. Создайте форму с полям ввода и кнопками. Организуйте работу формы: при нажатии на кнопку "Цифра" выводится последняя цифра, а на

кнопку "Четность" соответственно четная или нет. Вывод результата организовать как в краткой, так и в полной форме.

Вариант 8

1. Напишите функции, возвращающую трехзначное число по известным цифрам и определяющую полученное число кратно трем или нет. Создайте форму с полям ввода и кнопками. Организуйте работу формы: при нажатии на кнопку "Число" выводится полученное число, а на кнопку "Кратность" соответственно кратно трем или нет. Вывод результата организовать как в краткой, так и в полной форме.

Вариант 9

1. Напишите функции, вычисляющие температуру Фаренгейта по заданной температуре Цельсия и наоборот (Для перевода температуры из шкалы Фаренгейта в шкалу Цельсия нужно от исходного числа отнять 32 и умножить результат на 5/9. Для перевода температуры из шкалы Цельсия в шкалу Фаренгейта нужно умножить исходное число на 9/5 и прибавить 32.).

Создайте форму с полям ввода и кнопками. Организуйте работу формы: при нажатии на кнопку "Фаренгейт" выводится результат в Фаренгейтах, а на кнопку "Цельсий" соответственно в цельсиях. Вывод результата организовать как в краткой, так и в полной форме.

Вариант 10

1. Напишите функции вычисления площади внешней окружности (Square1), площади внутренней окружности (Square2) и площади кольца (RingSquare), ограниченного двумя окружностями заданных радиусов. Создайте форму с полям ввода и кнопками. Организуйте работу формы: при нажатии на кнопку "Окружности" выводятся площади окружностей, а на кнопку "Кольцо" соответственно площадь кольца. Вывод результата организовать как в краткой, так и в полной форме.

Вариант 11

1. Напишите функции, вычисляющую сумму цифр заданного двузначного числа и определяющую кратно оно пяти или нет. Создайте форму с полям ввода и кнопками. Организуйте работу формы: при нажатии на кнопку "Сумма" выводится полученная сумма, а на кнопку "Кратность" соответственно кратно пяти или

нет. Вывод результата организовать как в краткой, так и в полной форме.

Вариант 12

1. Напишите функции, возвращающую двухзначное число по известным цифрам и корень квадратный из полученного числа. Создайте форму с полям ввода и кнопками. Организуйте работу формы: при нажатии на кнопку "Число" выводится полученное число, а на кнопку "Корень" соответственно корень из этого числа. Вывод результата организовать как в краткой, так и в полной форме.

Задание 2 для всех вариантов

Необходимо взять задание №2 из лабораторной работы «Работа с одномерными массивами». И написать программу, решающую данную задачу, но с использованием формы и подпрограмм.

Реализовать следующие подпрограммы:

- заполнения одномерного массива случайными числами, в которую передается какой массив надо заполнять.
- вывода элементов массива в ListBox, также передается какой массив надо выводить
- выполнения задания и вывода результата

возможна реализация и других дополнительных подпрограмм.

Для программы должна быть создана форма, которая будет содержать кнопки, такие как заполнение массива случайными числами, вывода элементов массива на экран в виде ListBox, выполнения задания. Возможно наличие других кнопок и полей ввода.

8. ЛАБОРАТОРНАЯ РАБОТА №8: ПОДПРОГРАММЫ

8.1. Цель работы

Научиться работать с файлами, как способом хранения информации на винчестере. Освоить основные операции над файлами.

8.2. Теоретическая часть

Файл — именованная область на диске, содержащая некоторую информацию.

Преимущества:

Хранят данные в промежутках между запусками программ.

Размер данных в файле может существенно превышать оперативную память компьютера.

Файлы обычно классифицируют по двум признакам:

1. По типу компонент:

- текстовые;
- двоичные:
- типизированные;
- бестиповые.

2. По способу доступа:

- последовательный;
- произвольный.

Текстовые файлы

Тип text. Состоят из строк переменной длины, в конце каждой из которых находится символ перехода на новую строку (#13#10 в Windows, #10 в Linux). В PascalABC.NET это константа NewLine.

Двоичные файлы

Информация хранится в виде двоичного кода.

Типизированные файлы

Тип file of <type>. Содержат данные фиксированного типа <type>.

Бестиповые файлы

Тип file. Могут хранить данные различных типов.

В двоичных файлах информация хранится в том виде, как она хранится в оперативной памяти, а в текстовых числовая информация преобразуется к строковому виду и обратно, что занимает больше времени.

Файл называется **файлом произвольного доступа**, если можно перейти к его i-му элементу за время, не зависящее от размеров файла («за константное время»),

файлом последовательного доступа, если переход к i-му элементу требует количество операций, пропорциональное i («требует линейного времени»).

Все файлы, содержащие элементы разного размера могут иметь только последовательный доступ к элементам. Таковыми являются: текстовые, бестиповые файлы.

Типизированные файлы имеют произвольный доступ

Понятие файловой переменной, файлового указателя

Перед тем, как работать с информацией в файле, его надо открыть. После этого можно выполнять операции чтения из файла и записи в файл. По окончании работы его нужно закрыть.

Закрытый файл можно: переименовывать, перемещать, копировать, удалять.

С каждым открытым файлом связан так называемый файловый указатель, который указывает на текущую позицию в файле.

Файловый указатель создается при открытии файла, и, как правило, устанавливается на 1й элемент файла.

После каждой операции чтения или записи файловый указатель продвигается вперед на размер считанных элементов.

2 способа открытия файла

- Reset(f) — открытие текстового файла на чтение, а двоичного — на чтение и запись; файл должен существовать; файловый указатель — на начало файла;

- Rewrite(f) — создание нового файла (если такого файла не существовало) или обнуление существующего; файловый указатель — в начало; текстовые файлы при этом открываются только на запись, а двоичные — на чтение и запись;

Функция Eof(f) (расшифровывается как End Of File) возвращает true, если файловый указатель находился за концом файла.

После работы с данными в файле его необходимо закрыть с помощью Close(f).

Если, не закрывая, выполнить Reset(f), то файловый указатель просто перейдет к началу.

Подпрограммы для работы с типизированными файлами

procedure Truncate(f)

//Усекает типизированный файл f, отбрасывая все элементы с позиции файлового указателя

function FileSize(f)

//Возвращает количество элементов в типизированном файле f

function FilePos(f)

//Возвращает текущую позицию файлового указателя в типизированном файле f

procedure Seek(f, i);

//Устанавливает текущую позицию файлового указателя в типизированном файле f на элемент с номером i

Перейти на конец файла для добавления элементов: Seek(f, FileSize(f));

Перейти на предыдущую позицию: Seek(f, FilePos(f) - 1);

Пример 1. Чтение и запись типизированного файла

```
var f: file of integer;
begin
 Assign(f, 'a.dat');
 {связывает файловую переменную f с файлом на диске 'a.dat'
 (файл на диске может отсутствовать)}
 //файл состоит из элементов (1, 2, 3)
Reset(f); //файловый указатель — на элемент "1"
var x: integer;
 read(f, x); //x = 1
 //файловый указатель — на элемент "2"
write(f, x+1, x+2); //файловый указатель — за концом файла
 {выполнить read(f, x) НЕЛЬЗЯ, т.к. произойдет
 ошибка времени выполнения
 "чтение за концом файла"}
write(f, 7); //можно записать число в конец

Close(f);
```

```
end.
```

Пример 2. Переход в типизированном файле

Добавить в конец файла 'a.dat' элемент 0 (при необходимости создать файл).

```
const
  fileName = 'a.dat';
 var
  f: file of integer;
 begin
  Assign(f, fileName);
  if FileExists(fileName) then
  begin
    Reset(f);
    Seek(f, FileSize(f));
  end
  else
    Rewrite(f);
   write(f, 0);
  Close(f);
end.
```

Возведение всех элементов файла в квадрат.

```
const fileName = 'a.dat';
var f: file of real;
begin
  Assign(f, fileName);
  Reset(f);
  for var i:=0 to FileSize(f)-1 do
  begin
    var x: real;
    read(f, x);
    Seek(f, i);
    write(f, x * x);
  end;
  Close(f);
  end.
```

8.3. Последовательность выполнения работы

Реализовать программу с текстовым или графическим интерфейсом. В программе должно быть несколько процедур и как минимум одна функция (желательно каждый пункт меню реализовать в виде отдельной подпрограммы).

В программе обязательно должен быть пункт меню "Выход".

Обязательно необходимо обрабатывать исключения.

Во всех вариантах сделать пункт меню "Запись в позицию" и "Чтение позиции" - пользователь вводит номер элемента файла на место которого он хочет записать значение или прочитать значение.

Также обязательно добавить пункт меню "Печать файла" - происходит вывод содержимого файла.

8.4. Варианты заданий

Вариант 1

Дан одномерный массив целых чисел из 10 элементов. При запуске программы выводится меню: 1 - заполнить массив случайными числами, 2- ввести элементы массива с клавиатуры, 3-найти минимальный элемент, 4 - записать элементы массива и минимальный элемент в файл.

Вариант 2

Дан одномерный массив вещественных чисел из 10 элементов. При запуске программы выводится меню: 1 - заполнить массив случайными числами, 2- ввести элементы массива с клавиатуры, 3-найти сумму элементов массива, 4 - записать элементы массива и сумму в файл.

Вариант 3

Дан одномерный массив целых чисел из 10 элементов. При запуске программы выводится меню: 1 - заполнить массив случайными числами, 2- ввести элементы массива с клавиатуры, 3-найти сумму положительных элементов массива, 4 - записать элементы массива и сумму в файл.

Вариант 4

Дан одномерный массив вещественных чисел из 10 элементов. При запуске программы выводится меню: 1 - заполнить массив случайными числами, 2- ввести элементы массива с клавиатуры, 3-найти сумму отрицательных элементов массива, 4 - записать элементы массива и сумму в файл.

Вариант 5

Дан одномерный массив целых чисел из 10 элементов. При запуске программы выводится меню: 1 - заполнить массив случайными числами, 2- ввести элементы массива с клавиатуры, 3-найти количество положительных элементов массива, 4 - записать положительные элементы массива и их количество в файл.

Вариант 6

Дан одномерный массив целых чисел из 10 элементов. При запуске программы выводится меню: 1 - заполнить массив случайными числами, 2- ввести элементы массива с клавиатуры, 3-найти количество отрицательных элементов массива, 4 - записать отрицательные элементы массива и их количество в файл.

Вариант 7

Дан одномерный массив целых чисел из 10 элементов. При запуске программы выводится меню: 1 - заполнить массив случайными числами, 2- ввести элементы массива с клавиатуры, 3-найти максимальный элемент, 4 - записать элементы массива и максимальный элемент в файл.

Вариант 8

Дан типизированный файл, содержащий 10 целых значений. При запуске программы выводится меню: 1 - заполнить одномерный массив значениями из файла, 2- вывести элементы массива на экран, 3-найти максимальный элемент.

Вариант 9

Дан типизированный файл, содержащий 10 дробных значений. При запуске программы выводится меню: 1 - заполнить одномерный массив значениями из файла, 2- вывести элементы массива на экран, 3-найти минимальный элемент.

Вариант 10

Дан типизированный файл, содержащий 10 целых значений. При запуске программы выводится меню: 1 - заполнить одномерный массив значениями из файла, 2- вывести элементы массива на экран, 3-найти максимальный положительный элемент массива.

Вариант 11

Дан типизированный файл, содержащий 10 дробных значений. При запуске программы выводится меню: 1 - заполнить одномерный массив значениями из файла, 2- вывести элементы массива на экран, 3-найти минимальный отрицательный элемент массива.

Вариант 12

Дан одномерный массив целых чисел из 10 элементов. При запуске программы выводится меню: 1 - заполнить массив случайными числами, 2- ввести элементы массива с клавиатуры, 3-найти количество четных элементов массива, 4 - записать четные элементы массива и их количество в файл.

Вариант 13

Дан одномерный массив целых чисел из 10 элементов. При запуске программы выводится меню: 1 - заполнить массив случайными числами, 2- ввести элементы массива с клавиатуры, 3-найти количество нечетных элементов массива, 4 - записать нечетные элементы массива и их количество в файл.

Вариант 14

Дан двумерный массив целых чисел размера 3x3. При запуске программы выводится меню: 1 - заполнить массив случайными числами, 2- ввести элементы массива с клавиатуры, 3-найти сумму элементов главной диагонали, 4 - записать элементы главной диагонали и их сумму в файл.

Вариант 15

Дан двумерный массив целых чисел размера 3x3. При запуске программы выводится меню: 1 - заполнить массив случайными числами, 2- ввести элементы массива с клавиатуры, 3-найти

произведение элементов побочной диагонали, 4 - записать элементы побочной диагонали и их произведение в файл.

Вариант 16

Дан двумерный массив целых чисел размера 4x4. При запуске программы выводится меню: 1 - заполнить массив случайными числами, 2- ввести элементы массива с клавиатуры, 3-найти количество положительных элементов массива, 4 - записать положительные элементы и их количество в файл.

Вариант 17

Дан двумерный массив целых чисел размера 4x4. При запуске программы выводится меню: 1 - заполнить массив случайными числами, 2- ввести элементы массива с клавиатуры, 3-найти количество четных элементов массива, 4 - записать четные элементы и их количество в файл.

Вариант 18

Дан двумерный массив целых чисел размера 4x4. При запуске программы выводится меню: 1 - заполнить массив случайными числами, 2- ввести элементы массива с клавиатуры, 3-найти сумму нечетных элементов массива, 4 - записать нечетные элементы и их сумму в файл.

Вариант 19

Дан типизированный файл, содержащий 12 целых значений. При запуске программы выводится меню: 1 - заполнить двумерный массив размера 3x4 значениями из файла, 2- вывести элементы массива на экран, 3-найти максимальный элемент массива.

Вариант 20

Дан типизированный файл, содержащий 12 целых значений. При запуске программы выводится меню: 1 - заполнить двумерный массив размера 4x3 значениями из файла, 2- вывести элементы массива на экран, 3-найти сумму элементов массива.

ЛАБОРАТОРНЫЕ РАБОТЫ СЛЕДУЮЩЕГО СЕМЕСТРА

1. ЛАБОРАТОРНАЯ РАБОТА №1: РАБОТА С ТЕКСТОВЫМИ СИМВОЛЬНЫМИ ФАЙЛАМИ

1.1. Цель работы

Научиться работать с символьными и текстовыми файлами. Понимать отличия данных видов файлов. Освоить принципы работы с текстовыми и символьными файлами

1.2. Теоретическая часть

Подпрограммы для работы с текстовыми файлами

procedure Reset(f)

//Открывает текстовый файл f ТОЛЬКО на чтение.

procedure Rewrite(f)

//Открывает текстовый файл f ТОЛЬКО на запись, обнуляя его содержимое

procedure Append(f)

//Открывает текстовый файл f на дополнение

//Файловый указатель устанавливается за концом файла

function Eoln(f)

//Возвращает True, если файловый указатель находится на символе конца строки в текстовом файле f

function SeekEof(f)

//Пропускает пробельные символы, после чего возвращает True, если достигнут конец текстового файла f

function SeekEoln(f)

//Пропускает пробельные символы, после чего возвращает True, если достигнут конец строки в текстовом файле f

procedure read(f, a, b, c)

//Считывает значения в переменные a, b, c;

//Переменные могут быть числовых типов, символьного, строкового

procedure write(f, a, b, c)

//Записывает значения переменных a, b, c в текстовый файл f;

//Переменные могут быть числовых типов, символьного, строкового

```
procedure readln(f, a, b, c);
procedure writeln(f, a, b, c);
procedure readln(f);
procedure writeln(f);
```

Пример 1. Работа с текстовыми файлами

```
Uses formsAbc;
Var
 f:text;
 op: openFileDialog;
function CountEmptyLine(f:text):integer;
var stroka:string;
 summa:integer;
begin
      Reset(f);
     Summa := 0;
      While not(EOF(f)) do
      Begin
         Readln(f,stroka);
         If stroka='' Then
              Inc(summa);
      End;
      Close(f);
      Result:=summa;
end;
procedure CnELine;
begin
op:=new OpenFileDialog();
   if op.ShowDialog()=DialogResult.OK then
   begin
      Assign(f, op.FileName);
      a.value:=CountEmptyLine(f);
   end;

end;
procedure main;
Begin
End.
```

Пример 2. Работа с файловыми диалогами

```
uses Formsabc;
```

```
var
x:integerfield;
 op:openfiledialog;
 sf:savefiledialog;
 b1, b2:button;
 f:file of char;
 l:listbox;
procedure opf;
var c:char;
begin
op:= new OpenFileDialog();
 if op.ShowDialog()=DialogResult.OK then
 begin
assign(f, op.FileName);
 reset(f);
 l.Items.Clear();
 while not eof(f) do
 begin
read(f,c);
 l.Items.Add(c);
end;
 close(f);
end;
end;
procedure svf;
begin
sf:=new SaveFileDialog();
 if sf.ShowDialog()=DialogResult.OK then
 begin
assign(f,sf.FileName);
 rewrite(f);
 write(f, '3');
 close(f);
end;
end;

procedure main;
begin
b1:=new Button('open');
 b1.Click+=opf;
 b2:=new Button('save');
 b2.Click+=svf;
```

```
 l:=new ListBox();
end;
begin
main;
end.
```

1.3. Варианты заданий

При открытии и сохранении файла необходимо использовать файловые диалоги.

Вариант 1

1. Даны символьные файлы f1 и f2. Переписать с сохранением порядка следования компоненты файлы f1 в файл f2, а компоненты файла f2 - в файл f1. Использовать вспомогательный файл.

2. Дан текстовый файл f. Переписать в файл g все строки из файла f, которые начинаются с буквы 'c'.

Вариант 2

1. Дан символьный файл f. В файле f не менее двух компонент. Определить, являются ли два первых символа файла цифрами. Если да, то установить, является ли число, образованное этими цифрами, четным.

2. Дан текстовый файл f. Вывести все его строки, содержащие более 60 символов.

Вариант 3

1. Дан символьный файл f. Получить файл g, образованный из файла f заменой всех его прописных (больших) букв одноименными строчными (малыми).

2. Дан текстовый файл f. Вывести самую длинную строку файла. Если в файле имеется несколько строк с наибольшей длиной, то получить одну из них.

Вариант 4

1. Дан символьный файл f. Подсчитать число вхождений в файл буквы 'a' и 'b'.

2. Дан текстовый файл f. Посчитать количество символов в каждой строке и записать в файл g.

Вариант 5

1. Даны символьные файлы f и g. Определить, совпадают ли компоненты файла f с компонентами файла g. Если нет, то получить номер первой компоненты, в которой файлы f и g отличаются между собой.

2. Дан текстовый файл f. Переписать в файл g все строки из файла f, которые заканчиваются на вопросительный знак.

Вариант 6

1. Даны символьные файлы f и g. Записать в файл h все совпадающие компоненты файлов f и g.

2. Имеется текстовый файл f. Переписать строки из файла f в файл g, добавляя после каждой строки, начинающейся с буквы 'a' строку '++++++++++'.

Вариант 7

1. Дан символьный файл f. Записать в файл g с сохранением порядка следования те символы файла f, которым в этом файле предшествует буква 'a'.

2. Дан текстовый файл f. Вывести все строки файла длина которых больше 30 символов.

Вариант 8

1. Даны символьные файлы f1 и f2. Переписать в файл f посмивольно из файлов f1и f2 (один символ из файла f1, один символ из файла f2).

2. Дан текстовый файл f. Вывести строки файла, в которых есть не менее пяти букв 'и'.

Вариант 9

1. Дан символьный файл f. Переписать в файл g все цифры из файла f.

2. Дан текстовый файл f. Вывести самую короткую строку файла. Если в файле имеется несколько строк с наименьшей длиной, то получить одну из них.

Вариант 10

1. Дан символьный файл f. Переписать в файл g все гласные буквы из файла f.

2. Дан текстовый файл f. Записать в файл g все пятые символы из каждой строки.

Вариант 11

1. Дан символьный файл f. Записать в файл g символы имеющиеся в файле f в обратном порядке.

2. Дан текстовый файл f. Вывести строки из файла, номер которых кратен 3.

2. ЛАБОРАТОРНАЯ РАБОТА №2: СОЗДАНИЕ ИЗОБРАЖЕНИЙ ИЗ ГРАФИЧЕСКИХ ПРИМИТИВОВ

2.1. Цель работы

Научиться составлять сложные изображения из графических примитивов.

2.2. Теоретическая часть

Графические примитивы представляют собой процедуры, осуществляющие рисование в графическом окне.

Рисование осуществляется текущим пером (линии), текущей кистью (заливка замкнутых областей) и текущим шрифтом (вывод строк).

procedure SetPixel(x,y: integer; c: Color);

Закрашивает пиксел с координатами (x,y) цветом c

procedure PutPixel(x,y: integer; c: Color);

Закрашивает пиксел с координатами (x,y) цветом c

function GetPixel(x,y: integer): Color;

Возвращает цвет пиксела с координатами (x,y)

procedure MoveTo(x,y: integer);

Устанавливает текущую позицию рисования в точку (x,y)

procedure LineTo(x,y: integer);

Рисует отрезок от текущей позиции до точки (x,y). Текущая позиция переносится в точку (x,y)

procedure LineTo(x,y: integer; c: Color);

Рисует отрезок от текущей позиции до точки (x,y) цветом c. Текущая позиция переносится в точку (x,y)

procedure Line(x1,y1,x2,y2: integer);

Рисует отрезок от точки (x1,y1) до точки (x2,y2)

procedure Line(x1,y1,x2,y2: integer; c: Color);

Рисует отрезок от точки (x1,y1) до точки (x2,y2) цветом c

procedure FillCircle(x,y,r: integer);

Заполняет внутренность окружности с центром (x,y) и радиусом r

procedure DrawCircle(x,y,r: integer);

Рисует окружность с центром (x,y) и радиусом r

procedure FillEllipse(x1,y1,x2,y2: integer);

Заполняет внутренность эллипса, ограниченного прямоугольником, заданным координатами противоположных вершин (x1,y1) и (x2,y2)

procedure DrawEllipse(x1,y1,x2,y2: integer);

Рисует границу эллипса, ограниченного прямоугольником, заданным координатами противоположных вершин (x1,y1) и (x2,y2)

procedure FillRectangle(x1,y1,x2,y2: integer);

Заполняет внутренность прямоугольника, заданного координатами противоположных вершин (x1,y1) и (x2,y2)

procedure FillRect(x1,y1,x2,y2: integer);

Заполняет внутренность прямоугольника, заданного координатами противоположных вершин (x1,y1) и (x2,y2)

procedure DrawRectangle(x1,y1,x2,y2: integer);

Рисует границу прямоугольника, заданного координатами противоположных вершин (x1,y1) и (x2,y2)

procedure FillRoundRect(x1,y1,x2,y2,w,h: integer);

Заполняет внутренность прямоугольника со скругленными краями; (x1,y1) и (x2,y2) задают пару противоположных вершин, а w и h – ширину и высоту эллипса, используемого для скругления краев

procedure DrawRoundRect(x1,y1,x2,y2,w,h: integer);

Рисует границу прямоугольника со скругленными краями; (x1,y1) и (x2,y2) задают пару противоположных вершин, а w и h – ширину и высоту эллипса, используемого для скругления краев

procedure Circle(x,y,r: integer);

Рисует заполненную окружность с центром (x,y) и радиусом r

procedure Ellipse(x1,y1,x2,y2: integer);

Рисует заполненный эллипс, ограниченный прямоугольником, заданным координатами противоположных вершин (x1,y1) и (x2,y2)

procedure Rectangle(x1,y1,x2,y2: integer);

Рисует заполненный прямоугольник, заданный координатами противоположных вершин (x1,y1) и (x2,y2)

procedure RoundRect(x1,y1,x2,y2,w,h: integer);

Рисует заполненный прямоугольник со скругленными краями; (x1,y1) и (x2,y2) задают пару противоположных вершин, а w и h – ширину и высоту эллипса, используемого для скругления краев

procedure Arc(x,y,r,a1,a2: integer);

Рисует дугу окружности с центром в точке (x,y) и радиусом r, заключенной между двумя лучами, образующими углы a1 и a2 с осью OX (a1 и a2 – вещественные, задаются в градусах и отсчитываются против часовой стрелки)

procedure FillPie(x,y,r,a1,a2: integer);

Заполняет внутренность сектора окружности, ограниченного дугой с центром в точке (x,y) и радиусом r, заключенной между двумя лучами, образующими углы a1 и a2 с осью OX (a1 и a2 – вещественные, задаются в градусах и отсчитываются против часовой стрелки)

procedure DrawPie(x,y,r,a1,a2: integer);

Рисует сектор окружности, ограниченный дугой с центром в точке (x,y) и радиусом r, заключенной между двумя лучами, образующими углы a1 и a2 с осью OX (a1 и a2 – вещественные, задаются в градусах и отсчитываются против часовой стрелки)

procedure Pie(x,y,r,a1,a2: integer);

Рисует заполненный сектор окружности, ограниченный дугой с центром в точке (x,y) и радиусом r, заключенной между двумя лучами, образующими углы a1 и a2 с осью OX (a1 и a2 – вещественные, задаются в градусах и отсчитываются против часовой стрелки)

procedure DrawPolygon(points: array of Point);

Рисует замкнутую ломаную по точкам, координаты которых заданы в массиве points

procedure FillPolygon(points: array of Point);

Заполняет многоугольник, координаты вершин которого заданы в массиве points

procedure Polygon(points: array of Point);

Рисует заполненный многоугольник, координаты вершин которого заданы в массиве points

procedure Polyline(points: array of Point);

Рисует ломаную по точкам, координаты которых заданы в массиве points

procedure TextOut(x,y: integer; s: string);

Выводит строку s в прямоугольник к координатами левого верхнего угла (x,y)

procedure FloodFill(x,y: integer; c: Color);

Заливает область одного цвета цветом c, начиная с точки (x,y).

procedure Curve(points: array of Point);

Рисует кривую по точкам, координаты которых заданы в массиве points

procedure DrawClosedCurve(points: array of Point);

Рисует замкнутую кривую по точкам, координаты которых заданы в массиве points

2.3. Последовательность выполнения работы

Написать программу, которая при нажатии на кнопку F1 начинает рисовать в соответствии с заданием, приводимым ниже, а при нажатии на кнопку F2 - рисует график указанной функции, при нажатии правой кнопки мыши - графическое окно очищается.

Ваш ответ необходимо загрузить в данном разделе

2.4. Варианты заданий

1. Нарисуйте окружность, центр которой расположен в центре графического окна, а диаметр равен высоте окна.

Указание. Следует воспользоваться свойствами Window.Width, Window.Height и процедурой Circle.

Нарисуйте график функции - sin(x)

2. Нарисуйте диагонали графического окна (процедура Line). Нарисуйте график функции - cos(x)

3. Нарисуйте два прямоугольника — в левом верхнем и правом нижнем углах графического окна (процедура Rectangle).

Нарисуйте график функции - sin(2x)

4. Напишите процедуру, которая рисует прямоугольник с диагоналями:

procedure RectWithDiags(x, y, width, height: integer);

Здесь x, y – координаты верхнего левого угла прямоугольника; width, height – ширина и высота прямоугольника.

Нарисуйте график функции - cos(2x)

5. Напишите процедуру, которая рисует окружность с красным крестом в центре:

procedure CircleWithCross(x, y, r: integer);

Здесь x, y – координаты центра окружности; r – радиус окружности.

Указание. Для установки цвета пера воспользуйтесь свойством Pen.Color, а для установки его ширины — свойством Pen.Width:

```
Pen.Color := Color.Red;
Pen.Width := 5;
```

Нарисуйте график функции - x*x

6. Нарисуйте горизонтальный ряд из одинаковых квадратов. Попытайтесь предварительно определить, как изменяются координаты левых верхних углов квадратов, и записать рисование квадратов в цикл:

procedure SquaresRow(x0, y0, N, width, dist:nteger);

Здесь x0, y0 – координаты левого верхнего угла первого квадрата; N – количество квадратов; width – ширина одного квадрата; dist – расстояние между двумя соседними квадратами.

Нарисуйте график функции - x*x*x

7. Напишите процедуру, рисующую N концентрических окружностей (окружностей с общим центром и разными радиусами):

procedure InnerCircles(x, y, minR, step, N: integer);

Здесь x, y – координаты центра всех окружностей; minR – радиус наименьшей окружности; step – величина, на которую отличаются радиусы двух соседних окружностей; N – количество окружностей.

Указание. В решении установить стиль прозрачной кисти, чтобы каждая новая окружность не закрашивала предыдущую:

Brush.Style := bsClear;

Можно также рисовать окружности от внешней к внутренней. Нарисуйте график функции - x*x+2

8. Напишите процедуру FramedTextOut, которая выводит текст в рамке.

Нарисуйте график функции - (x+2)*(x+2)

9. Напишите процедуру TextOutRightBottomCorner, которая выводит заданный текст в правый нижний угол графического окна. Текст должен располагаться в указанном месте независимо от своего размера.

Нарисуйте график функции - tg(x)

10. Напишите процедуру
CharInCircleOut(x,y: integer; c: char),
которая выводит переданный ей символ в центре круга.
Нарисуйте график функции - (2-x)*(2-x)

11. Нарисуйте циферблат с делениями в виде отрезков.
Нарисуйте график функции - 1/x

12. Нарисуйте черный прямоугольник у левого края графического окна и переместите его к правому краю.

Нарисуйте график функции - 1/(x+3)

13. Напишите процедуру, которая выводит заданный текст в центре графического окна.

Нарисуйте график функции - 2*x*x+3

14. Напишите процедуру, которая выводит заданный текст в каждом из четырех углов графического окна.

Нарисуйте график функции - cos(3x)

15. Нарисуйте круг в левом верхнем углу графического окна и переместите его в правый нижний угол. Предварительно рассчитайте, как должны изменяться координаты центра круга, чтобы движение было равномерным.

Нарисуйте график функции - x*x-2

3. ЛАБОРАТОРНАЯ РАБОТА №3: ДИНАМИЧЕСКИЙ ДВУМЕРНЫЙ МАССИВ

3.1. Цель работы

Изучить правила работы с двумерным динамическим массивом и понять его преимущества по сравнению со статическим массивом.

3.2. Теоретическая часть

Динамический массив объявляется без задания фиксированной длины. Такое объявление создает лишь указатель. Даже многомерный динамический массив создается в виде одного неинициализированного указателя.

Процедура SetLength изменяет размер строки, одномерного динамического массива или многомерного динамического массива

Пример

```
type
        a=array of integer;
var
        x:array of a;
begin
       setlength(x,1);
       setlength(x[0],2);
       x[0,0] := 5; x[0,1]:=3;
       writeln(x[0,0], ' ', x[0,1]);
end.
```

Особенности динамических массивов

1. Динамические массивы можно инициализировать при описании:

```
var a: array of integer := (1,3,5);
```

2. Выделять память под динамическе массивовы можно с помощью операции new:

```
a := new integer[5];
```

Такой способ хорош тем, что он подчеркивает, что динамический массив в .NET является классом. Плох же он тем, что при повторном выделении памяти таким способом старое содержимое теряется.

3. Как мы упомянули, динамический массив в PascalABC.NET является классом, а значит, он имеет методы и свойства:

a.Length - свойство, возвращающее длину массива

System.Array.Sort(a) - статический метод, сортирующий массив a по возрастанию

System.Array.Reverse(a) - статический метод, инвертирующий данные в массиве a

и многие другие.

4. Для динамических массивов в PascalABC.NET имеет место структурная эквивалентность типов (в Delphi - именная). Поэтому следующий код в PascalABC.NET будет работать, а в Delphi вызовет ошибку компиляции:

```
var
 b1: array of integer;
 b2: array of integer;
 ...
b1 := b2;
```

5. Ввиду структурной эквивалентности типов для динамических массивов их можно передавать в подпрограмму следующим образом:

```
procedure print(a: array of integer);
begin
  for var i:=0 to a.Length-1 do
    write(a[i],' ');
end;
```

Напомним, что открытые массивы в PascalABC.NET отсутствуют!

6. Для динамических массивов (в отличие от статических) можно использовать цикл foreach (при условии, что мы осуществляем доступ к элементам только на чтение):

```
foreach x: integer in a do
  write(x,' ');
```

И, наконец, скажем несколько слов про двумерные динамические массивы. Они моделируются как массивы массивов.

Следующий код иллюстрирует создание двумерного динамического массива размера m на n:

```
var
  c: array of array of integer;
  m,n: integer;
 ...
read(m,n);
SetLength(c,m);
for var i:=0 to m-1 do
  SetLength(c[i],n);
```

3.3. Последовательность выполнения работы

Реализовать на основе динамического двумерного массива задание. Количество строк и столбцов вводит пользователь.

Интерфейс программы оформить с помощью модуля GraphABC;

Прорисовать в окне значения горячих клавиш.

Двумерный массив выводить в виде таблицы с очерченными контурами. Элементы главной и побочной диагоналей при выводе выделить цветом.

Предусмотреть понятную форму вывода результата задания.

Организовать возможность добавления строк и столбцов в процессе работы программы (закрепить за данными операциями горячие клавиши, например F5, F6).

3.4. Варианты заданий

1. Дано натуральное число n. Получить действительную матицу $|a_{ij}|$ i,j=1…n, для которой

$$a_{ij} = \begin{cases} \sin(i+j) & \text{при } i < j \\ 1 & \text{при } i = j \\ \arcsin\dfrac{i+j}{2i+3j} & \text{в остальных случаях.} \end{cases}$$

2. Дано натуральное число n. Выяснить сколько положительных элементов содержит матрица |aij| i,j=1..n, если **$a_{ij}=\sin(i+j/2)$**

3. Дано натуральное число n. Выяснить сколько положительных элементов содержит матрица |aij| i,j=1..n, если **$a_{ij}=\cos(i^2+n)$**
4. Дана действительная матрица размера nxm. Получить новую матрицу путем деления всех элементов данный матрицы на ее наибольший по модулю элемент.
5. Дана действительная квадратная матрица размера 5x5. Заменить нулями все элементы, расположенные на главной диагонали и выше нее.
6. Дана целочисленная матрица размера 6x6. Найти максимальный элемент главной диагонали и наименьший элемент побочной диагонали.
7. Дана действительная матрица размера 9x7. Найти значение наибольшего по модулю элемента матрицы, а также индексы какого-нибудь элемента с найденным значением модуля.
8. Даны две целочисленные матрицы размера 3x2 и 2x3. Получить третью матрицу, равную произведению двух матриц.
9. Дана действительная матрица размера 6x9. Найти среднее арифметическое наибольшего и наименьшего значений ее элементов.

10. В данной действительной матрице размера 6x9 поменять местами строку, содержащую элемент с наибольшим значением, со строкой, содержащей элемент с наименьшим значением. Предполагается, что эти элементы единственны.

4. ЛАБОРАТОРНАЯ РАБОТА №4: ДИНАМИЧЕСКИЕ СТРУКТУРЫ ДАННЫХ: СТЕК, ОЧЕРЕДЬ, ЛИНЕЙНЫЙ СПИСОК

4.1. Цель работы

Понять назначение динамических структур данных. Освоить правила работы с динамическими структурами.

4.2. Теоретическая часть

Если до начала работы с данными невозможно определить, сколько памяти потребуется для их хранения, память следует распределять во время выполнения программы по мере необходимости отдельными блоками. Блоки связываются друг с другом с помощью указателей. Такой способ организации данных называется динамической структурой данных, поскольку она размещается в динамической памяти и ее размер изменяется во время выполнения программы.

Из динамических структур в программах чаще всего используются линейные списки, стеки, очереди и бинарные деревья. Они различаются способами связи отдельных элементов и допустимыми операциями. Динамическая структура, в отличие от массива или записи, может занимать несмежные участки оперативной памяти.

Элемент любой динамической структуры состоит из двух частей: информационной, ради хранения которой и создается структура, и указателей, обеспечивающих связь элементов друг с другом.

Элемент динамической структуры описывается в виде записи, например:

```
type
pnode =  ^node;
node = record
d : word;                    { информационная }
```

```
s : string;          { часть }
p : pnode;           { указатель на следующий элемент }
end;
```

Линейные списки

В линейном списке каждый элемент связан со следующим и, возможно, с предыдущим. В первом случае список называется односвязным, во втором — двусвязным. Если последний элемент связать указателем с первым, получится кольцевой список.

Каждый элемент списка содержит ключ, идентифицирующий этот элемент. Ключ обычно бывает либо целым числом, либо строкой и является частью поля данных. В качестве ключа в процессе работы со списком могут выступать разные части поля данных. Ключи разных элементов списка могут совпадать.

Над списками можно выполнять следующие операции:

начальное формирование списка (создание первого элемента);

добавление элемента в конец списка;

чтение элемента с заданным ключом;

вставка элемента в заданное место списка (до или после элемента с заданным ключом);

удаление элемента с заданным ключом;

упорядочивание списка по ключу.

Стеки

Стек является простейшей динамической структурой. Добавление элементов в стек и выборка из него выполняются из одного конца, называемого вершиной стека. Другие операции со стеком не определены. При выборке элемент исключается из стека.

Говорят, что стек реализует принцип обслуживания LIFO (last in — first out, последним пришел — первым обслужен). Кстати, сегмент стека назван так именно потому, что память под локальные переменные выделяется по принципу LIFO. Стеки широко применяются в системном программном обеспечении, компиляторах, в различных рекурсивных алгоритмах.

Очереди

Очередь — это динамическая структура данных, добавление элементов в которую выполняется в один конец, а выборка — из другого конца. Другие операции с очередью не определены. При выборке элемент исключается из очереди. Говорят, что очередь

реализует принцип обслуживания FIFO (first in — first out, первым пришел — первым обслужен). В программировании очереди применяются очень широко — например, при моделировании, буферизованном вводе-выводе или диспетчеризации задач в операционной системе.

Рассмотрим основные операции с линейными односвязными списками.

1. Предварительные описания

```
type
  PNode = ^Node;
  Node = record
    data: integer;
    next: PNode;
end;

function NewNode(data: integer; next: PNode): PNode;
begin
  New(Result);
  Result^.data := data;
  Result^.next := next;
end;
```

2. Вставка элемента со значением x в начало списка, на который указывает p.

```
p := NewNode(x,p);
```

3. Удаление элемента из начала непустого списка, на который указывает p.

```
 var t := p;
 p := t^.next;
 Dispose(t);
```

4. Вставка элемента со значением x после текущего, на который указывает p.

```
p^.next := NewNode(x,p^.next);
```

5. Удаление элемента, следующего за текущим, на который указывает p.

```
var t := p^.next;
if t<>nil then
begin
```

```
  p^.next := t^.next;
  Dispose(t);
end;
```

6. Вставка элемента со значением x перед текущим, на который указывает p.

```
 p^.next := NewNode(p^.data,p^.next);
 p^.data := x;
 p := p^.next;
```

7. Удаление текущего элемента, на который указывает p.

```
var t := p^.next;
 p^:= t^;
 Dispose(t);
```

Элемент, следующий за текущим, должен существовать.

8. Вывод списка, на первый элемент которого указывает p.

```
while p<>nil do
begin
  write(p^.data,' ');
  p := p^.next;
end;
```

9. Поиск элемента со значением x. На первый элемент списка указывает p.

```
while (p<>nil) and (p^.data<>x) do
  p := p^.next;
```

10. Разрушение списка.

```
while p<>nil do
begin
  var t := p;
  p := p^.next;
  Dispose(t);
end;
```

4.3. Последовательность выполнения работы

Программа должна иметь интерфейс на основе формы (FormsABC). Выводить содержимое списка (очереди, стека) в ListBox.

При реализации линейного списка предусмотреть возможность добавления и удаления элемента:

а) добавлять в список, соблюдая указанную упорядоченность;
б) удаление любого элемента из списка;
в) возможность редактирования любого элемента списка.

При реализации очереди предусмотреть возможность:
а) добавления в конец очереди;
б) удаление элемента из начала очереди;
в) возможность редактирования первого и последнего элементов очереди.

При реализации стека предусмотреть возможность:

а) добавления в конец стека;
б) удаление элемента из конца стека;
в) возможность редактирования последнего элемента стека.

4.4. Варианты заданий

1. Организовать линейный список цветов: хранить название цвета и его числовой код. Упорядочивать по первой букве названия, если первые буквы совпадают, то по второй и т.д.

2. Организовать линейный список показаний температуры: хранить дату (в текстовом виде), температуру. Упорядочивать по температуре.

3. Организовать линейный список студентов: хранить ФИО и номер зачетки. Упорядочивать по первой букве фамилии, если первые буквы совпадают, то по второй и т.д.

4. Организовать линейный список дисциплин: хранить название дисциплины и количество часов. Упорядочивать по длине названия дисциплины.

5. Организовать очередь цветов: хранить название цвета и его числовой код.

6. Организовать очередь показаний температуры: хранить дату (в текстовом виде), температуру.

7. Организовать очередь студентов: хранить ФИО и номер зачетки.

8. Организовать очередь дисциплин: хранить название дисциплины и количество часов.

9. Организовать стек цветов: хранить название цвета и его числовой код.

10. Организовать стек показаний температуры: хранить дату (в текстовом виде), температуру.

11. Организовать стек студентов: хранить ФИО и номер зачетки.

12. Организовать стек дисциплин: хранить название дисциплины и количество часов.

13. Организовать линейный список автомобилей: хранить марку и год выпуска. Упорядочивать по году выпуска.

14. Организовать очередь автомобилей: хранить марку и год выпуска.

15. Организовать стек автомобилей: хранить марку и год выпуска.

5. ЛАБОРАТОРНАЯ РАБОТА №5: СЛОЖНАЯ ДИНАМИЧЕСКАЯ СТРУКТУРА ДАННЫХ

5.1. Цель работы

Изучить принципы использования указателей при реализации сложных динамических структур.

5.2. Теоретическая часть

Двусвязный список.

Двусвязный список, отличается от односвязного только тем, что каждый узел списка имеет указатель не только на следующий элемент, но и на предыдущий.

Все основные операции для двусвязных списков, похожи на операции с односвязными, но за счет добавления еще одного указателя, в код вносятся некоторые изменения.

```
Type
    Telem = integer;

    TList = ^TElement;
    TElement = Record
     Data : Telem;
     Next, Prev : TList;
    End;
```

Кольцевые (циклические) двухсвязные списки

Во многом эти списки похожи на обычные двухсвязные, за одним исключением: они закольцованы, т.е., последний элемент в таком списке ссылается на первый. Это вносит свои коррективы в алгоритм добавления элемента к кольцевому списку.

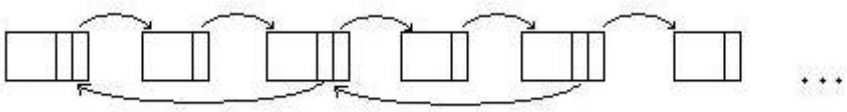
5.3. Последовательность выполнения работы

Построить структуру в динамической памяти. Элементы с двумя связями содержат строковые данные, с одной - символьные или целочисленные. Выполнить просмотр структуры и поиск в структуре: по заданному строковому элементу найти связанные с ним символьные или целочисленные элементы.

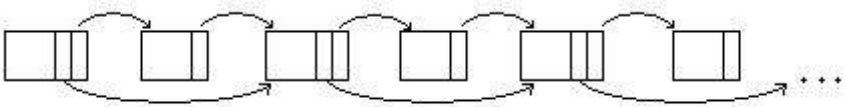
Для организации интерфейса использовать модуль GraphABC.

Вывод элементов списка организовать в графическом виде. Результат поиска выделить цветом на выведенном списке.

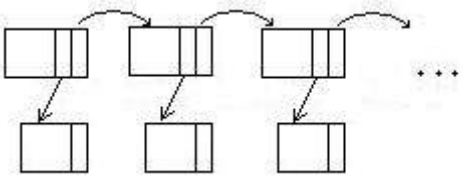
5.4. Варианты заданий

1. 

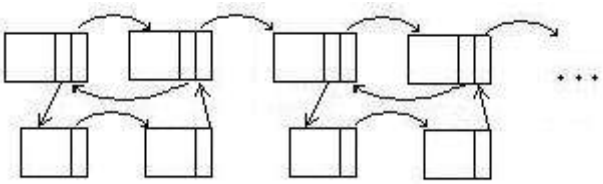
2.



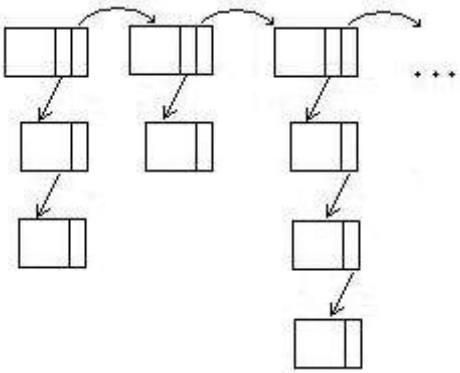
3.



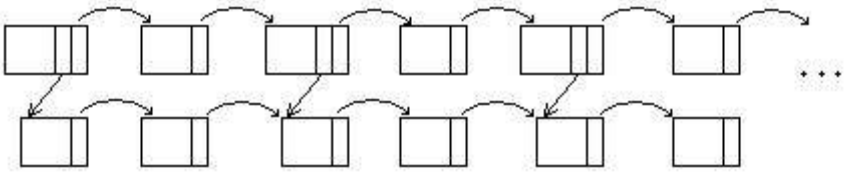
4.

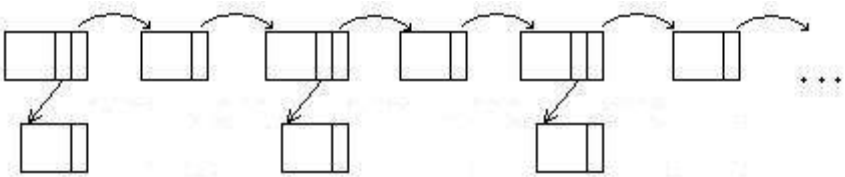


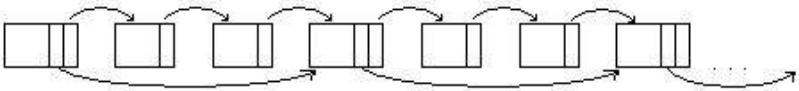
5.

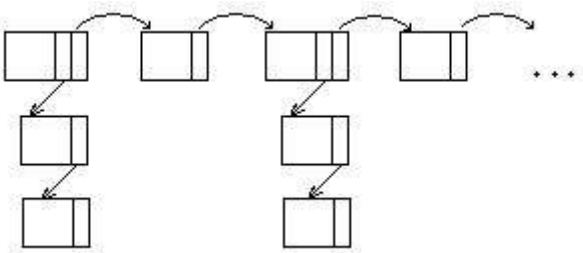


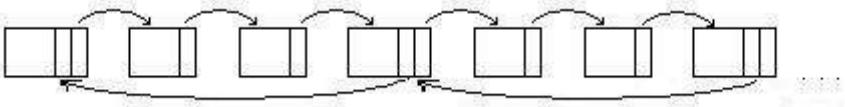
6.

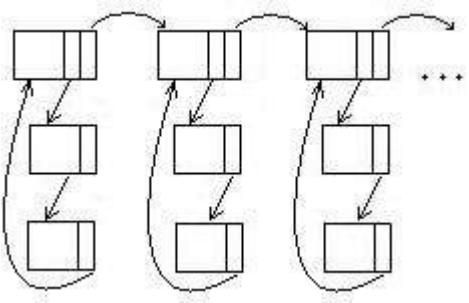


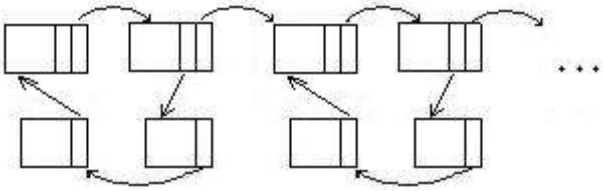
7. 

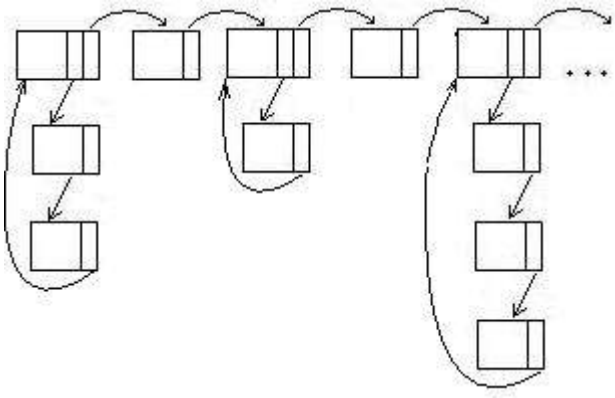
8. 

9. 

10. 

11. 

12. 

13. 

6. ЛАБОРАТОРНАЯ РАБОТА №6: РАЗРАБОТКА ПРОГРАММЫ НА ОСНОВЕ ТРЕХ ВЗАИМОСВЯЗАННЫХ КЛАССОВ

6.1. Цель работы

Освоить отличия объектно-ориентированного подхода от процедурного. Изучить понятие класса и его элементов. Научиться составлять простые программы на основе взаимосвязанных классов.

6.2. Теоретическая часть

Класс представляет собой составной тип, состоящий из полей (переменных), методов (процедур и функций) и свойств. Переменные типа класс называются объектами или экземплярами класса.

Описание класса имеет вид:

```
type
  имя класса = class
    описания полей
        объявления или описания методов и описания свойств
  end;
```

Переменная типа класс хранит в действительности указатель на объект. Однако, при обращении к полям, методам или свойствам объекта разыменование такого указателя не требуется; указывается имя объекта и затем, после разделителя-точки, указывается имя поля, метода или свойства:

```
var p: Person := new Person('Иванов',20);
 p.Print;
 p.NextYear;
 p.Print;
```

PascalABC.NET существуют четыре типа атрибутов видимости: public (открытый), private (закрытый), protected (защищенный) и internal (внутренний). К члену класса, имеющему атрибут public, можно обратиться из любого места программы, члены класса с атрибутом private доступны только внутри методов этого класса, члены класса с атрибутом protected доступны внутри методов этого класса и всех его подклассов, члены класса с атрибутом internal доступны внутри сборки (термин .NET, сборка в нашем понимании - это множество файлов, необходимых для генерации .exe или .dll-файла). Кроме того, private и protected члены видны отовсюду в пределах модуля, в котором определен класс.

Пример. Работа с двумя классами

Пояснение к примеру

Методы делятся на классовые и экземплярные. Классовые методы в .NET называются статическими.

Объявление классового метода начинается с ключевого слова class.

Экземплярные методы можно вызывать только через переменную-объект класса.

Классовые же методы не связаны с конкретным экземпляром класса; их следует вызывать в виде: имя класса.имя метода(параметры).

```
uses FormsABC;
type
    Model = class
        class procedure Calc(x,y: integer; var sum,prod: integer);
        begin
            sum := x + y;
            prod := x * y;
        end;
    end;

    View = class
    private
        a,b,sum,prod: IntegerField;
        procedure MyClick;
        begin
            var s,p: integer;
            Model.Calc(a.Value,b.Value,s,p);
            sum.Value := s;
            prod.Value := p;
        end;
    public
        constructor Create;
        begin
            a := new IntegerField('a:');
            b := new IntegerField('b:');
            LineBreak;
            sum := new IntegerField('Сумма:',220);
            LineBreak;
            prod := new IntegerField('Произведение:',220);
            LineBreak;
```

```
            EmptyLine(20);
            var d := new Button('Вычислить');
            d.Click += MyClick;
        end;
    end;

begin
 var v := new View;
end.
```

6.3. Последовательность выполнения работы

Написать программу, используя модель объектно-ориентированного программирования.

В программе должно быть реализовано не менее трех классов.

Например: Программа вычисляет периметр квадрата.

```
Uses FormsABC;
type

Square=class
private
 side:integer;

public
 constructor(x:integer);
 begin
 side:=x;
 end;

function GetSide:integer;
 begin
 Result:=side;
 end;
end;

Model=class
private
 sq:Square;

public
constructor(val:integer);
```

```
 begin
 sq:=new Square(val);
 end;
function perimeter:integer;
 begin
 Result:=4*sq.GetSide();
 end;
end;

View=class
private
 side:IntegerField;
 ans:IntegerField;
 sqr:Model;
 b:button;

procedure ClickPer;
 begin
 sqr:=new Model(side.Value);
 ans.Value:=sqr.perimeter();
 end;

public

constructor Create;
 begin
 side:=new IntegerField('сторона');
 ans:=new IntegerField('ответ');
 b:=new Button('периметр');
 b.Click+=ClickPer;
 end;
end;

begin
 var v:=new View;
end.
```

6.4. Варианты заданий

Вариант 1

1. Напишите функции вычисления длины окружности (CircleLength) и площади круга (CircleSquare) по заданному радиусу. Создайте форму с полями ввода и кнопками. Организуйте работу

формы: при нажатии на кнопку "Длина" выводится результат длины окружности, а на кнопку "Площадь" соответственно площадь круга. Вывод результата организовать как в краткой, так и в полной форме.

Вариант 2

1. Напишите функции вычисления расстояния между двумя точками, заданными своими координатами (Distance) и нахождения координаты середины между двумя точками(Middle). Создайте форму с полям ввода и кнопками. Организуйте работу формы: при нажатии на кнопку "Расстояние" выводится результат расстояния между точками, а на кнопку "Середина" соответственно координаты середины. Вывод результата организовать как в краткой, так и в полной форме.

Вариант 3

1. Напишите функцию, вычисляющую площадь треугольника, заданного координатами своих вершин (TriangleSquare), используя формулу Герона. Для вычисления напишите вспомогательную функцию Distance, которая находит расстояние между двумя точками. Создайте форму с полям ввода и кнопками. Организуйте работу формы: при нажатии на кнопку "Расстояние" выводится результат расстояния между точками, а на кнопку "Площадь" соответственно площадь треугольника. Вывод результата организовать как в краткой, так и в полной форме.

Вариант 4

1. Напишите функции, вычисляющую возраст человека в годах по заданному возрасту в месяцах и определяющую пора человеку получать паспорт или нет. Создайте форму с полям ввода и кнопками. Организуйте работу формы: при нажатии на кнопку "Возраст" выводится возраст в годах, а на кнопку "Паспорт" соответственно пора или нет получать паспорт. Вывод результата организовать как в краткой, так и в полной форме.

Вариант 5

1. Напишите функции, определяющую, сколько месяцев осталось до дня рождения человека, если известен его возраст в месяцах, и определяющую исполнилось человеку уже 18 лет или нет. Создайте форму с полям ввода и кнопками. Организуйте работу формы: при нажатии на кнопку "ДР" выводится сколько меся-

цев осталось до ДР, а на кнопку "Совершеннолетие" соответственно есть 18 лет или нет. Вывод результата организовать как в краткой, так и в полной форме.

Вариант 6

1. Напишите функции, вычисляющие первую и вторую цифры заданного двузначного числа. Создайте форму с полям ввода и кнопками. Организуйте работу формы: при нажатии на кнопку "Первая" выводится первая цифра, а на кнопку "Вторая" соответственно вторая. Вывод результата организовать как в краткой, так и в полной форме.

Вариант 7

1. Напишите функции, вычисляющую последнюю цифру заданного числа и определяющую четное число или нет. Создайте форму с полям ввода и кнопками. Организуйте работу формы: при нажатии на кнопку "Цифра" выводится последняя цифра, а на кнопку "Четность" соответственно четная или нет. Вывод результата организовать как в краткой, так и в полной форме.

Вариант 8

1. Напишите функции, возвращающую трехзначное число по известным цифрам и определяющую полученное число кратно трем или нет. Создайте форму с полям ввода и кнопками. Организуйте работу формы: при нажатии на кнопку "Число" выводится полученное число, а на кнопку "Кратность" соответственно кратно трем или нет. Вывод результата организовать как в краткой, так и в полной форме.

Вариант 9

1. Напишите функции, вычисляющие температуру Фаренгейта по заданной температуре Цельсия и наоборот (Для перевода температуры из шкалы Фаренгейта в шкалу Цельсия нужно от исходного числа отнять 32 и умножить результат на 5/9. Для перевода температуры из шкалы Цельсия в шкалу Фаренгейта нужно умножить исходное число на 9/5 и прибавить 32.).

Создайте форму с полям ввода и кнопками. Организуйте работу формы: при нажатии на кнопку "Фаренгейт" выводится результат в Фаренгейтах, а на кнопку "Цельсий" соответственно в цельсиях. Вывод результата организовать как в краткой, так и в полной форме.

Вариант 10

1. Напишите функции вычисления площади внешней окружности (Square1), площади внутренней окружности (Square2) и площади кольца (RingSquare), ограниченного двумя окружностями заданных радиусов. Создайте форму с полям ввода и кнопками. Организуйте работу формы: при нажатии на кнопку "Окружности" выводятся площади окружностей, а на кнопку "Кольцо" соответственно площадь кольца. Вывод результата организовать как в краткой, так и в полной форме.

Вариант 11

1. Напишите функции, вычисляющую сумму цифр заданного двузначного числа и определяющую кратно оно пяти или нет. Создайте форму с полям ввода и кнопками. Организуйте работу формы: при нажатии на кнопку "Сумма" выводится полученная сумма, а на кнопку "Кратность" соответственно кратно пяти или нет. Вывод результата организовать как в краткой, так и в полной форме.

Вариант 12

1. Напишите функции, возвращающую двухзначное число по известным цифрам и корень квадратный из полученного числа. Создайте форму с полям ввода и кнопками. Организуйте работу формы: при нажатии на кнопку "Число" выводится полученное число, а на кнопку "Корень" соответственно корень из этого числа. Вывод результата организовать как в краткой, так и в полной форме.

7. ЛАБОРАТОРНАЯ РАБОТА №7: НАСЛЕДОВАНИЕ И ПОЛИМОРФИЗМ

7.1. Цель работы

Освоить понятие наследования классов. Изучить принципы работы с классом потомком и классом предком.

7.2. Теоретическая часть

Класс может быть унаследован от другого класса. Класс, от которого наследуют, называют базовым классом (надклассом, предком), а класс, который наследуется, называется производным классом (подклассом, потомком). При наследовании все поля, методы и свойства базового класса переходят в производный класс,

кроме этого, могут быть добавлены новые поля, методы и свойства и переопределены (замещены) старые методы.

При описании класса его базовый класс указывается в скобках после слова class.

Например:

```
type
  BaseClass = class
    procedure p;
   procedure q(r: real);
  end;
  MyClass = class(BaseClass)
   procedure p;
    procedure r(i: integer);
 end;
```

В данном примере процедура p переопределяется, а процедура r добавляется в класс MyClass.

Если не указать имя базового класса, то считается, что класс наследуется от класса Object - предка всех классов. Например, BaseClass наследуется от Object.

Перед словом class может быть указано ключевое слово final – в этом случае от класса запрещено наследовать.

Пример.

```
type
parent=class
protected
    x:integer;
public
  constructor;
  begin
      x:=5;
  end;

  constructor(z:integer);
  begin
    x:=z;
  end;

  procedure show; virtual;
```

```
  begin
    writeln('parent ',x);
  end;
end;

child=class(Parent)
private
  y:integer;

  function GetY:integer;
  begin
    Result:=y;
  end;

  procedure SetY(yy:integer);
  begin
    y:=yy;
  end;

public
  constructor;
  begin
    inherited create;
    y:=6;
  end;

  constructor(t,q:integer);
  begin
    inherited create(t);
    y:=q;
  end;

  procedure show; override;
  begin
    writeln('child ',y,' ', self.x);
  end;


  property PY: integer read GetY write SetY;

end;
```

```
var mas:array[1..3] of Parent;
 c:Child;

BEGIN
   mas[1] := new parent(4);
   mas[2] := new child;
   c:=mas[2] as Child;
   c.PY:=33;
   mas[3] := new child(10,8);
   for var i:=1 to 3 do mas[i].show();
END.
```

Пример. Использование наследования классов для работы с несколькими формами

```
uses
   System,
   System.Windows.Forms;

type

   student=class
   private
      fio:string[20];
   public
     constructor Create();
     begin

     end;

     procedure SetName(st:string);
     begin
        fio:=st;
     end;

     function GetName:string;
     begin
         Result:=fio;
     end;
   end;

   miniForm=class(Form)
```

```
  private
     tx:TextBox;
     bt:Button;
     stud:student;
  public
     procedure MyButtonClick(sender: Object; e: EventArgs);
     begin
         stud.SetName(tx.Text);
         self.Close;
    end;

    constructor Create(st:string; s:student);
    begin
        inherited Create();
        stud:=s;
        self.Text:=st;
        tx:=new TextBox();
        self.Controls.Add(tx);
        bt:=new Button();
        bt.Text:='Add';
        bt.Top:=50;
        bt.Click+=MyButtonClick;
        self.Controls.Add(bt);
    end;
  end;

  myForm=class(Form)
  private
     stud:student;
     b1:Button;
     l:listBox;
     procedure MyButtonClick(sender: Object; e: EventArgs);
     begin
       var mn:miniForm;
       mn:=new miniForm('222', stud);
       mn.ShowDialog();
       l.Items.Add(stud.GetName);
    end;
 public
    constructor Create(st:string);
    begin
        inherited Create();
```

```
            stud:=new student;
            self.Text:=st;
            b1:= new Button();
            b1.Text:='Add';
            b1.Click+=MyButtonClick;
            self.Controls.Add(b1);
            l:=new ListBox();
            l.Top:=50;
            self.Controls.Add(l);
        end;
    end;

var m :myForm;
begin
 m := new myForm('111');
 Application.Run(m);
end.
```

7.3. Последовательность выполнения работы

Для интерфейса использовать модуль FormsABC. Реализовать три класса (один - класс родитель, второй и третий классы - дети). Один из методов базового класса должен быть виртуальным, а в наследниках - он должен быть перегружен. Каждый класс должен иметь не менее двух конструкторов, класс-ребенок обращается к конструктору базового класса.

Классы наследники также должны иметь собственные методы, которые не определены в родители.

Создать одномерный массив из 5 элементов. Елементы которого это указтели на класс родителя. Инициализировать первый элемент массива объектом класса родителя, а другие элементы объектами класса детей.

В форме создать три кнопки и обеспечить их функционал: Инициализация массива объектов. Вывод элементов массива в Listbox. Третья кнопка выполняет какой-нибудь метод для выбранного в listbox-е объекта класса.

Для некоторых скрытых полей класса реализовать свойства.

7.4. Варианты заданий

1. Животное, лошадь, собака.
2. Средства передвижения, автомобиль, поезд.
3. Мебель, стул, диван.

4. Электрическая техника, холодильник, утюг.
5. Домашняя утварь, ложка, кружка.
6. Компьютерная техника, монитор, принтер.
7. Запоминающее устройство, жесткий диск, съемный диск.
8. Человек, преподаватель, студент.
9. Сотрудник, библиотекарь, охранник.
10. Книга, учебник, словарь.
11. Бумага, плакат, картина.
12. Комната, кухня, спальня.
13. Канцелярия, ручка, маркер.
14. Летательные аппараты, самолет, ракета.
15. Плоская фигура, квадрат, круг.
16. Объемная фигура, куб, сфера.