Общие сведения о массивах

В предыдущих главах мы рассматривали задачи, в которых использовались скалярные переменные. Однако при обработке однотипных данных (целочисленных значений, строк, дат и т.п.) оказывается удобным использовать массивы. Например, можно создать массив для хранения значений температуры в течение года. Вместо создания множества (365) переменных для хранения каждой температуры, например temperature1, temperature2, temperature3, ... temperature365 можно использовать один массив с именем temperature, где каждому значению будет соответствовать порядковый номер.

№ элемента	1	2	3	4	...	364	365
temperature	-1.5	-3	-6.7	1		2	-3

Таким образом, можно дать следующее определение.
Массив – структурированный тип данных, состоящий из фиксированного числа элементов одного типа.
Массив, представленный на рисунке, имеет 7 элементов, каждый элемент сохраняет число вещественного типа. Элементы в массиве пронумерованы от 1 до 7. Такого рода массив, представляющий собой просто список данных одного и того же типа, называют простым, или одномерным массивом. Для доступа к данным, хранящимся в определенном элементе массива, необходимо указать имя массива и порядковый номер этого элемента, называемый индексом.

1-й элемент массива	2-й элемент массива	3-й элемент массива	4-й элемент массива	5-й элемент массива	6-й элемент массива	7-й элемент массива
-1.5	-3.913	13.672	-1.56	45.89	4.008	-3.61

Если возникает необходимость хранения данных в виде таблиц, в формате строк и столбцов, то необходимо использовать многомерные массивы.
На рисунке приведен пример массива, состоящего из трех строк и четырех столбцов. Это двумерный массив. Строки в нем можно считать первым измерением, а столбцы — вторым. Для доступа к данным, хранящимся в этом массиве, необходимо указать имя массива и два индекса, первый должен соответствовать номеру строки, а второй — номеру столбца, в которых хранится необходимый элемент.

		Номера столбцов			
		1	**2**	**3**	**4**
Номера строк	**1**	6.3	4.3	-1.34	5.02
	2	1.1	4.7	8.12	8.5
	3	-2.4	-6.2	11.23	8.18

После общего знакомства с понятием «массив», рассмотрим работу с массивами в языке Pascal.

Описание массивов

Для описания массива служат служебные слова **array of.** Описать массив можно двумя способами:

Ввести новый тип данных, а потом описать переменные нового типа. В этом случае формат оператора type следующий:

```
type
        имя_типа = array [тип_индекса] of тип_компонентов;
```

В качестве типа_индекса следует использовать перечислимый тип. Тип_компонентов — это любой ранее определенный тип данных, например:

```
type
        massiv=array[0..12] of real;
//Тип данных massiv из 13 элементов,
//элементы нумеруются от 0 до 12.
        dabc=array[-3..6] of integer;
//Тип данных dabc из 10 элементов,
//элементы нумеруются от -3 до 6.
var
        x,y:massiv;
        z: dabc;
```

Можно не вводить новый тип, а просто описать переменную следующим образом:

```
var
        переменная: array [тип_индекса] of тип_переменной;
```

Например:

```
var
        z,x: array[1..25] of word;
//Массивы z и x из 25 значений типа word,
//элементы нумеруются от 1 до 25.
        g:array[-3..7] of real;
//Массив g из 11 значений типа real,
//которые нумеруюся от -3 до 7.
```

Для описания массива можно использовать предварительно определенные константы:

```
const
        n=10;
        m=12;
var
        a: array[1..n] of real;
        b: array[0..m] of byte;
```

Константы должны быть определены до использования, так как массив не может быть переменной длины!

Двумерный массив (матрицу) можно описать, применив в качестве базового типа (типа компонентов) одномерный:

```
type
        massiv=array[1..200] of real;
        matrica=array[1..300] of massiv;
var
        ab:matrica;
```

Такая же структура получается при использовании другой формы записи:

```
type
        matrica = array [1..300,1..200] of real;
var
```

ab:matrica;

или

var ab:array [1..300,1..200] of real;

При всех трех определениях мы получали матрицу вещественных чисел, состоящую из 300 строк и 200 столбцов.

Аналогично можно ввести трехмерный массив, или массив большего числа измерений:

type

abc=array [1..4, 0..6, -7..8, 3..11] of real;

var

b:abc;

Операции над массивами

Для работы с массивом как с единым целым надо использовать имя массива (без указания индекса в квадратных скобках). Для доступа к элементу массива необходимо указать имя массива и в квадратных скобках порядковый номер элемента массива, например **x[1], y[5], c[25], A[8].**

В языке Free Pascal определена операция присваивания для массивов, идентичных по структуре (с одинаковыми типами индексов и компонентов). Например, если массивы C и D описаны как

var C,D: array [0..30] of real;

то можно записать оператор **C:=D;.** Такая операция сразу всем элементам массива C присвоит значения соответствующих им по номерам элементов массива D.

Выполнение любой другой операции над массивом надо организовывать поэлементно, для чего необходимо организовать цикл, в котором последовательно обрабатывать элементы массива, сначала обрабатываем первый элемент массива, затем второй, третий, ...,n-й.

Для обработки элементов массива удобно использовать цикл for.

Ввод-вывод элементов массива

Язык Free Pascal не имеет специальных средств ввода-вывода всего массива, поэтому данную операцию следует организовывать поэлементно. При вводе массива необходимо последовательно вводить 1-й, 2-й, 3-й и т.д. элементы массива, аналогичным образом поступить и при выводе. Следовательно, как для ввода, так и для вывода необходимо организовать стандартный цикл обработки массива.

Для обращения к элементу массива необходимо указать имя массива и в квадратных скобках номер элемента, например **X[5], b[2]** и т. д.

Организация ввода-вывода консольных приложениях

Реализуем эти алгоритмы в консольных приложениях. В режиме визуальное приложение лучше всего воспользоваться компонентом StringGrid: TStringGrid; из вкладки компонентов Additional или продумать другие альтернативы.

/Ввод элементов массива X с помощью цикла while.

```
var
    x: array [1..100] of real;
    i,n: integer;
begin
    writeln ('введите размер массива'); readln(N);
```

```
        i:=1;
        while (i<=N) do
        begin
                write('x(',i,')= ');
                readln(x[i]);
                i:=i+1
        end;
end.
```

//Ввод элементов массива X с помощью цикла for.

```
var
        i,n: integer;
        x: array [1..100] of real;
begin
        readln(N);
        for i:=1 to N do
                begin
                write('x(',i,')= ');
                readln(x[i])
        end;
end.
```

Цикл for ... do удобнее использовать для обработки всего массива, и в дальнейшем при выполнении подобных операций с массивами мы чаще будем применять именно его.
Вывод массива организуется аналогично вводу, только вместо блока ввода элемента массива будет блок вывода.
Предлагаем рассмотреть несколько вариантов вывода массива вещественных чисел a=(1.1, 2.2, 3.3, 4.4, 5.5, 6.6, 7.7, 8.8), самостоятельно разобраться, чем они отличаются друг от друга, и решить, какой из вариантов удобнее в конкретной задаче.

//Вариант 1

```
        for i: = 1 to n do  write (a[i]:3:1);
```

//Вариант 2

```
        for i: = 1 to n do write (a[i]:6:2);
```

// Вариант 3

```
        for i: = 1 to n do write (a[i]:3:1,' ');
```

// Вариант 4

```
        writeln ('массив A');
        for i:=1 to n do writeln(a[i]:6:2);
```

// Вариант 5

```
        for i:=1 to n do write ('a(',i,')=',a[i]:3:1,' ');
```

// Вариант 6

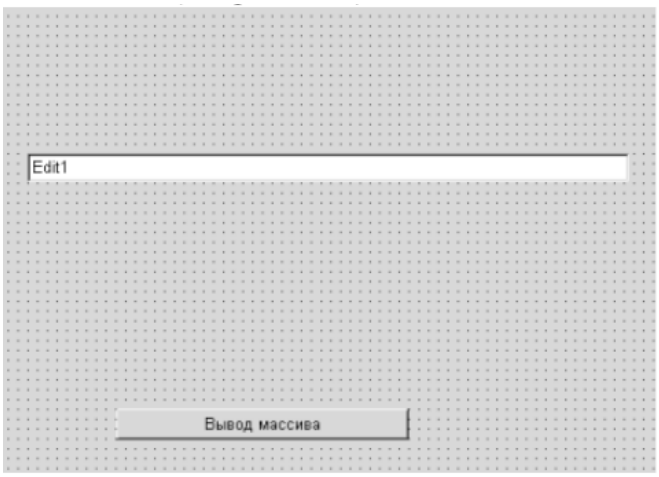
```
        for i:=1 to n do writeln ('a(',i, ')= ',a[i]:6:2);
```

Ввод-вывод данных в визуальных приложениях

Рассмотрим возможности организации ввода-вывода массивов в визуальных приложениях, для вывода массивов можно использовать стандартный компонент типа **TEdit**.

Рассмотрим эти возможности на примере стандартной задачи вывода массива вещественных чисел **a=(1.1, 2.2, 3.3, 4.4, 5.5, 6.6, 7.7, 8.8)**.

Расположим на форме кнопку (компонент типа TButton) и компонент типа Tedit.



В табл. приведены свойства компонентов типа TEdit и TButton.

Свойство	Name1	Text	Height	Left	Top	Width	ReadOnly
Значение	Edit1	' '	23	103	184	492	True

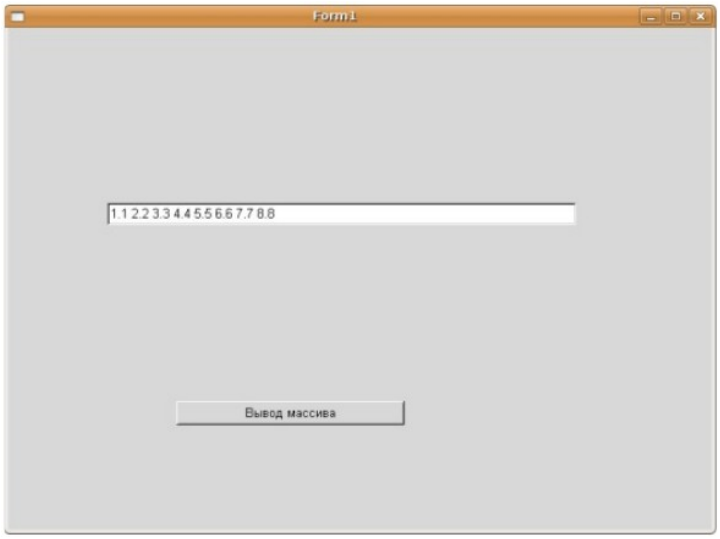
Свойство	Name1	Caption	Height	Left	Top	Width
Значение	label1	Button1	25	177	392	239

Наше приложение по щелчку по кнопке «Вывод массива» будет выводить массив a в поле ввода Edit1. Алгоритм вывода массива заключается в следующем: каждое число переводится в строку с помощью функции FloatToStr, после чего полученная строка добавляется к полю вывода. Текст обработчика события Button1Click с комментариями приведен ниже.

```
procedure TForm1.Button1Click(Sender: TObject);
var
//Исходный массив a.
        a:array [1..8] of real=(1.1,2.2,3.3,4.4, 5.5, 6.6, 7.7, 8.8);
        i:integer;
//В переменной n хранится количество элементов в массиве вещественных чисел.
        n:integer=8;
        S:string='';
begin
        Edit1.Text:='';
//Цикл for для последовательной обработки всех элементов массива a.
        for i:=1 to n do
//Добавление в поле ввода Edit1 строки, которая получена из элемента массива a[i].
                Edit1.Text:=Edit1.Text+FloatToStr(a[i])+' ';
```

end;

После щелчка по кнопке «Вывод массива» окно приложения станет подобным тому, как представлено на рис.



Для ввода массивов в Lazarus в простейшем случае можно воспользоваться функцией InputBox. В качестве примера рассмотрим проект, в котором будет осуществляться ввод массива при щелчке по кнопке. На форме расположим единственную кнопку. Текст обработчик события Button1Click с комментариями приведен ниже.

procedure TForm1.Button1Click(Sender: TObject);

var i,n:byte; X:array [1..20] of real;

begin

//Количество элементов массива.

n:=StrToInt(InputBox('Ввод элементов массива','n=','7'));

for i := 1 to n do

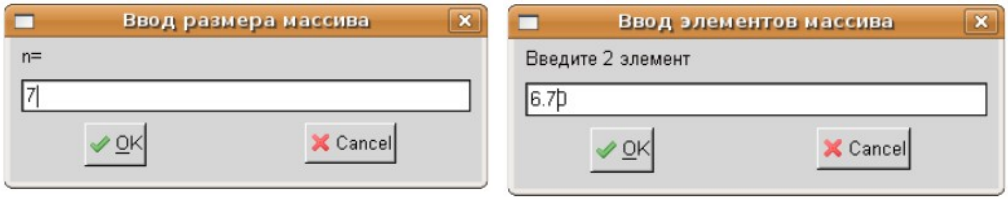
//Поэлементный ввод.

//Ввод очередного элемента массива

X[i]:=StrToFloat(InputBox('Ввод элементов массива','Введите '+IntToStr(i)+ ' элемент','0,00'));

end;

При щелчке по кнопке на экране появится окно для ввода размера массива. После корректного ввода размера массива последовательно будут появляться диалоговые окна для ввода очередного элемента, подобные представленному на рис. 5.17.



Для вывода массива с помощью диалогового окна можно применить функцию **MessageDlg**:

for i:= 1 to n do

MessageDlg('X['+IntToStr(i)+']='+FloatToStr(X[i]), MtInformation, [mbOk],0),

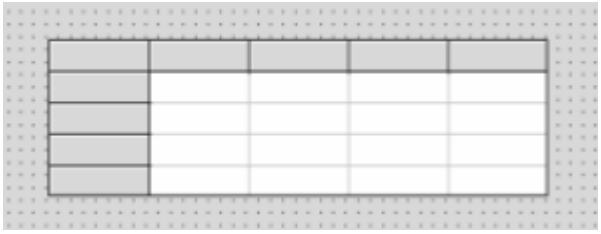
которая будет открывать отдельное окно для каждого элемента.

Чтобы у пользователя была возможность просматривать элементы массива одновременно, можно из них сформировать строку, а затем вывести ее, например, на форму в виде метки или в окне сообщения:

```
var
        i,n:byte;
        X:array [1..20] of real;
        S:string;
begin
//В переменную строкового типа записывается
//пустая строка.
        S:='';
        for i:=1 to n do
//Выполняется слияние элементов массива, преобразованных в строки и символов пробела –
//результат строка, в которой элементы массива перечислены через пробел.
        S:=S+FloatToStrF(X[i],ffFixed,5,2)+' ';
//Вывод строки на форму в виде метки.
        Label2.Caption:=S;
//Аналогично строку можно вывести в виде сообщения, используя функцию
//MessageDlg(S,MtInformation,[mbOk],0);
```

Наиболее универсальным способом ввода-вывода как одномерных, так и двумерных массивов является компонент типа TStringGrid («таблица строк»). Познакомимся с этим компонентом подробнее.

На рис. представлен проект формы, на которой расположен компонент типа TStringGrid (таблица строк). Ячейки таблицы строк предназначены для чтения или редактирования данных. Этот компонент может служить как для ввода, так и для вывода массива.



Основные свойства этого компонента представлены в таблице Основные свойства компонента типа TStringGrid

Свойство	Описание
Name	Имя компонента
ColCount	Количество столбцов таблицы
RowCount	Количество строк таблицы
Cells	Двумерный массив, в котором хранится содержимое таблицы. Ячейка таблицы, находящаяся на пересечении столбца номер col и строки номер row, определяется элементом Cells [col,row]; строки в компоненте нумеруются от 0 до RowCount-1, (столбцы от 0 до ColCount-1)

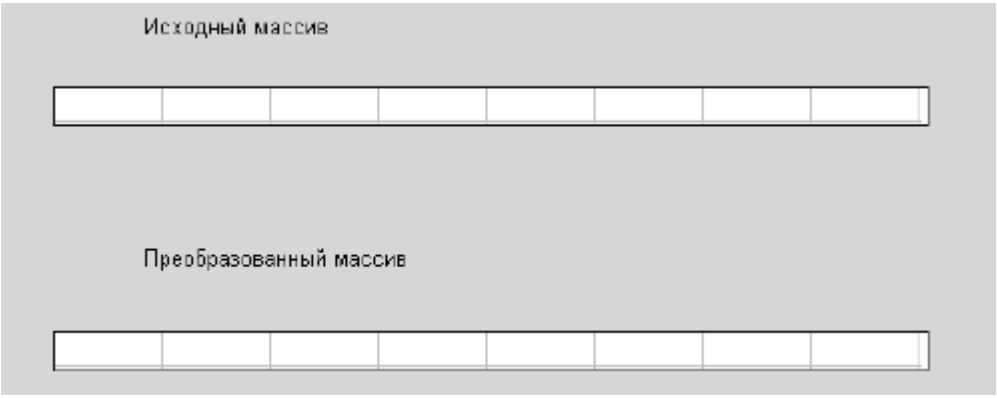
Свойство	Описание
FixedCols	Количество зафиксированных слева столбцов таблицы, которые выделяются цветом и при горизонтальной прокрутке таблицы остаются на месте
FixedRows	Количество зафиксированных сверху строк таблицы, которые выделяются цветом и при вертикальной прокрутке таблицы остаются на месте
ScrollBars	Параметр определяет наличие полос прокрутки, возможны следующие значения параметра: •ssNone — отсутствие полос прокурутки (пользователь может в этом случае перемещаться только с помощью курсора) •ssHorizontal, ssVertical или ssBoth — наличие горизонтальной, вертикальной или обеих полос прокрутки; •ssAutoHorizontal, ssAutoVertical или ssAutoBoth — появление горизонтальной, вертикальной или обеих полос прокрутки по мере необходимости.
Options.goEditing	Логическая переменная, которая определяет, может пользователь (True) или нет (False) редактировать содержимое ячеек таблицы
Options.goTab	Логическая переменная, которая разрешает (True) или запрещает (False) использование клавиши Tab для перемещения курсора в следующую ячейку таблицы
DefaultColWidth	Ширина столбцов таблицы
DefaultRowHeight	Высота строк таблицы
GridLineWidth	Ширина разграничительных линий между ячейками таблицы
Left	Расстояние от таблицы до левой границы формы
Top	Расстояние от таблицы до верхней границы формы
DefaultColWidth	Ширина столбцов таблицы
DefaultRowHeight	Высота строк таблицы
Height	Высота компонента типа TStringGrid
Width	Ширина компонента типа TStringGrid
Font	Шрифт, которым отображается содержимое ячеек таблицы

Рассмотрим использование компонента для ввода-вывода массивов на примере программы, с помощью которой можно осуществить ввод массива из восьми вещественных чисел, а затем вывести его в обратном порядке.

Разместим на форме две метки, два компонента типа TStringGrid и одну кнопку. Свойства компонентов StringGrid1 и StringGrid2 можно задать такими же, как показано в табл

Свойство	StringGrid1	StringGrid2	Описание
ColCount	8	8	Количество столбцов таблицы
RowCount	1	1	Количество строк
FixedCols	0	0	Количество зафиксированных слева столбцов таблицы

FixedRows	0	0	Количество зафиксированных сверху строк таблицы
Options.goEditing	True	False	Логическая переменная, которая определяет возможность пользователю редактировать содержимое ячеек таблицы
Left	186	186	Расстояние от таблицы до левой границы формы
Top	72	216	Расстояние от таблицы до верхней границы формы
Height	24	24	Высота компонента
Width	518	518	Ширина компонента

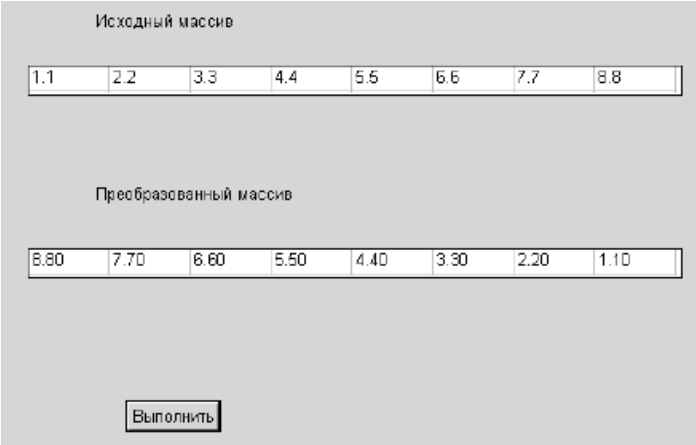


Не забудьте поставить кнопку «Выполнить».

В первую таблицу будем вводить элементы массива, во вторую выводить преобразованный массив. Щелчок по кнопке Выполнить вызовет следующую подпрограмму:

```
procedure Tform1.Button1Click( Sender: TObject) ;
var
        n,i:integer;
        a:array [0..7] of real;
begin
        for i:=0 to 7 do
 //Ввод массива.
//Из поля таблицы считывается элемент, преобразовывается в число и
//присваивается элементу массива.
        a[i]:=StrToFloat(StringGrid1.Cells[i,0]);
        for i:=0 to 7 do
 //Вывод массива.
//Элемент массива преобразовывается в строку и помещается в поле таблицы.
        StringGrid2.Cells[i,0]:=FloatToStrF(a[7-i],ffFixed,5,2);
end;
```

При запуске программы на выполнение появляется окно приложения, подобное представленному на рис. 5.20, пользователь вводит исходный массив, щелкаем по кнопке Выполнить, после чего окно приложения принимает вид, подобный представленному на рис.



В этом параграфе были рассмотрены различные способы ввода-вывода как в консольных, так и в визуальных приложениях. В дальнейшем мы будем использовать те из них, которые удобнее при решении конкретной задачи.
Теперь перейдем к рассмотрению основных алгоритмов обработки одномерных массивов, многие из которых аналогичны соответствующим алгоритмам обработки последовательностей (вычисление суммы, произведения, поиск элементов по определенному признаку, выборки и т. д.). Отличие заключается в том, что в массиве одновременно доступны все его компоненты, поэтому становятся возможными более сложные действия с массивами (например, сортировка элементов массива, удаление и вставка элементов и т.д.).

Вычисление суммы и произведения элементов массива

Нахождение суммы и произведения элементов массива аналогично подобным алгоритмам нахождения суммы и произведения элементов последовательности.

Дан массив X, состоящий из n элементов. Найти сумму элементов этого массива. Переменной S присваивается значение, равное нулю, затем последовательно к переменной S добавляются элементы массива X.

Соответствующий алгоритму фрагмент программы будет иметь вид:

```
s:=0;
for i:=1 to n do s:=s+x[i];
writeln('s=',s:7:3); //визуальном приложении надо изменить на соответствующее см. выше.
```

Найдем произведение элементов массива X. Решение задачи сводится к тому, что значение переменной P, в которую предварительно была записана единица, последовательно умножается на значение i–го элемента массива.

Соответствующий фрагмент программы будет иметь вид:

```
p:=1;
for i:=1 to n do p:=p*x[i];
```

writeln('P=',P:7:3); //визуальном приложении надо изменить на соответствующее см. выше.

Поиск максимального элемента в массиве и его номера

Рассмотрим задачу поиска максимального элемента (Max) и его номера (Nmax) в массиве X, состоящем из n элементов.

Алгоритм решения задачи следующий. Предположим, что первый элемент массива является максимальным, и запишем его в переменную Max, а в Nmax – его номер (число 1).

Затем все элементы, начиная со второго, сравниваем в цикле с максимальным. Если текущий

элемент массива оказывается больше максимального, то записываем его в переменную Max, а в переменную Nmax – текущее значение индекса i.

Соответствующий фрагмент программы будет иметь вид:

```
Max:=X[1]; Nmax:=1;
for i:=2 to n do
    if X[i]>Max then
        begin
            Max:=X[i];
            Nmax:=i;
        end;
write(' Max=',Max:1:3,' Nmax=',Nmax);
```

Алгоритм поиска минимального элемента в массиве будет отличаться от приведенного выше лишь тем, что в условном блоке и, соответственно, в конструкции if текста программы знак поменяется с > на <.

Сортировка элементов в массиве

Сортировка представляет собой процесс упорядочения элементов в массиве в порядке возрастания или убывания их значений. Например, массив X из n элементов будет отсортирован в порядке возрастания значений его элементов, если

$$X[1] \le X[2] \le ... \le X[n],$$

и в порядке убывания, если

$$X[1] \ge X[2] \ge ... \ge X[n].$$

Многие алгоритмы сортировки основаны на том факте, что надо переставлять два элемента таким образом, чтобы после перестановки они были правильно расположены друг относительно друга. При сортировке по возрастанию после перестановки элемент с меньшим индексом должен быть не больше элемента с большим индексом.

Рассмотрим некоторые из алгоритмов.

Сортировка методом «пузырька»(консольное приложение)

```
var
        i,j,n: byte; b:real;
        X: array [1..100] of real;
begin
        writeln ('введите размер массива ');
        readln (n);
        for i:=1 to n do
                begin
                        write('X[',i,']=');
                        readln (X[i]);
                end;
        writeln ('массив X ');
        for i:=1 to n do write (x[i]:5:2,' ');
        writeln;
        for j:=1 to n-1 do
        for i:=1 to n-j do
                if X[i] > X[i+1] then
                { Если текущий элемент больше следующего, то }
                begin { поменять их местами. }
                        b:=X[i];  { Сохранить значение текущего элемента. }
                        X[i]:=X[i+1];{Заменить текущий элемент следующим. }
                        X[i+1]:=b; {Заменить следующий элемент переменной b.}
                end;
        writeln('упорядоченный массив');
        for i:=1 to n do write (X[i]:5:2,' ');
        writeln;
end.
```

Сортировка выбором

```
//Сортировка выбором.
repeat
    max:=x[1]; nom:=1;
    for i:=2 to k do
        if max < X[i] then
        begin
            max:=X[i]; nom:=i;
        end;
    b:=x[nom]; x[nom]:=x[k]; x[k]:=b;
    k:=k-1;
until k=1;
```

Удаление элемента из массива

Удалить из массива X(n) отрицательные элементы.

```
program upor_massiv;
var
    i,n,j:byte;
    X: array [1..100] of real;
begin
    writeln ('введите размер массива ');
    readln (n);
    {Ввод массива.}
    for i:=1 to n do
        begin
            write('X[',i,']=');
            readln (X[i]);
        end;
    writeln ('массив X ');
    for i:=1 to n do
        write (x[i]:5:2,' ');
    writeln;
```

```
        i:=1;
        while(i<=n)do
        {Если очередной элемент массива X[i] отрицателен, то }
        if x[i]<0 then
                begin
                        {Удаляем элемент массива с номером i.}
                        for j:=i to n-1 do x[j]:=x[j+1];
                        {Уменьшаем размер массива.}
                        {Не переходим к следующему элементу массива.}
                        n:=n-1;
                end
        else
        {Если элемент не удалялся, то переходим к следующему элементу массива.}
        i:=i+1;
        writeln('Измененный массив:');
        for i:=1 to n do write (X[i]:5:2,' ');
        writeln;
end.
```

Вставка элемента в массив

```
var i,n,m:byte; X: array [1..100] of real;
        b:real;
begin
        writeln ('N='); readln (n);
        for i:=1 to n do
        begin
                write('X[',i,']='); readln (X[i]);
        end;
        writeln ('Массив X');
        for i:=1 to n do write (x[i]:5:2,' ');
        writeln;
        writeln ('m='); readln (m);
```

```
        writeln ('b='); readln(b);
        for i:=n downto m+1 do
                x[i+1]:=x[i];
        x[m+1]:=b;
        n:=n+1;
        writeln('Измененный массив');
        for i:=1 to n do write (X[i]:5:2,' ');
        writeln;
end.
```

Для визуального приложения необходимо изменить программы в соответствии концепцией визуального программирования

Задачи для решения

1. Записать положительные элементы массива X подряд в массив Y. Вычислить сумму элементов массива X и произведение элементов массива Y. Из массива Y удалить элементы, расположенные между максимальным и минимальным элементами.
2. Сформировать массив B, записав в него элементы массива A с нечетными индексами. Вычислить среднее арифметическое элементов массива B и удалить из него максимальный, минимальный и пятый элементы.
3. Дан массив целых чисел X. Переписать пять первых положительных элементов массива и последних два простых элемента в массив Y. Найти максимальный отрицательный элемент массива X.
4. Записать элементы массива X, удовлетворяющие условию $1 \le x_i \le 2$, подряд в массив Y. Поменять местами максимальный и минимальный элементы в массиве Y.
5. Переписать элементы массива целых чисел X в обратном порядке в массив Y. Вычислить количество четных, нечетных и нулевых элементов массива Y.
6. Определить максимальный и минимальный элементы среди положительных нечетных элементов целочисленного массива X. Удалить из массива все нулевые элементы.
7. Переписать элементы целочисленного массива $X=(x_1, x_2, ...,x_{12})$ в массив $Y = (y_1, y_2, ..., y_{12})$, сдвинув элементы массива X вправо на три позиции. При этом три элемента с конца массива X перемещаются в начало: $(y_1, y_2, ..., y_{12}) = (x_{10}, x_{11}, x_{12}, x_1 , x_2, ..., x_9)$. Определить номера максимального простого и минимального положительного элементов в массивах X и Y.
8. Записать элементы массива $X = (x_1 , x_2, ..., x_{15})$ в массив $Y = (y_1,y_2 , ..., y_{15})$, сдвинув элементы массива X влево на четыре позиции. При этом четыре элемента, стоящие в начале массива X, перемещаются в конец: $(y_1, y_2, ..., y_{15}) = (x_5, x_6, ..., x_{15}, x_1, x_2, x_3, x_4)$. Поменять местами минимальный и максимальный элементы массива Y.
9. В массиве X определить количество элементов меньших среднего арифметического значения. Удалить из массива положительные элементы, расположенные между

максимальным и минимальным.
10. Вычислить среднее арифметическое элементов массива X, расположенных между его минимальным и максимальным значениями. Если минимальный элемент размещается в массиве раньше максимального, то упорядочить массив на данном промежутке по возрастанию его элементов.
11. Определить, содержит ли заданный массив группы элементов, расположенные в порядке возрастания их значений. Если да, то определить количество таких групп.
12. В заданном массиве целых чисел найти самую маленькую серию подряд стоящих нечетных элементов.
13. Удалить из массива целых чисел все простые числа, расположенные до максимального значения.
14. Удалить из массива предпоследнюю группу элементов, представляющих собой знакочередующийся ряд.
15. Задан массив целых положительных чисел X. Определить количество совершенных чисел в массиве. Удалить из массива последних два отрицательных числа. Сформировать массив Y, куда записать номера элементов массива X, являющихся простыми числами.
16. Переписать положительные элементы массива целых чисел X в обратном порядке в массив Y. Вычислить процент четных, нечетных и нулевых элементов массива Y. Перевести элементы массива Y в двоичную систему счисления.
17. Определить максимальный и минимальный элементы среди положительных четных элементов целочисленного массива X. Удалить из массива X совершенные числа, расположенные после максимального значения.
18. Заданы массивы вещественных чисел X и Y. Сформировать массив Z, куда записать положительные элементы массивов Y и Z в семеричной системе счисления. Определить номера максимального и минимального элементов в массиве Z.
19. Записать четные положительные элементы целочисленных массивов X и Y в массив Z. Поменять местами минимальный и максимальный элементы массива Z. Вывести элементы массива Z в четверичной системе счисления.
20. Из целочисленного массива X удалить все числа, превышающие среднее арифметическое простых элементов массива.
21. В массивах вещественных чисел X и Y записаны координаты точек на плоскости. Найти две точки, расстояние между которыми наименьшее.
22. Определить, содержит ли заданный массив вещественных чисел группы элементов, расположенные в порядке убывания их значений. Если да, то определить группу наименьшей длины.
23. В заданном массиве целых чисел найти самую большую серию подряд стоящих четных элементов.
24. Удалить из массива целых чисел все элементы, которые в пятеричном представлении не содержат нулей.
25. Из массивов вещественных чисел A и B сформировать массив C, записав в него элементы массивов A и B, которые не содержат «семерок» в восьмеричном представлении.