Основы программирования

Учитель информатики и ИКТ
ГОУ г.Москвы СОШ №310
«У Чистых прудов»
Цыбикова Т.Р.

Тема 6.

МАССИВЫ

Массивы

В рассмотренных ранее примерах программ производилась обработка одиночных данных – значений простых переменных.

При решении практических задач данные объединяются в различные структуры, наиболее простыми из которых являются массивы.

Массив – именованный набор с фиксированным количеством однотипных данных.

- В массивы объединены результаты экспериментов, списки фамилий сотрудников, различные сложные структуры данных.
- Так, список из классного журнала 10 «А» является массивом.
- В массиве могут быть одинаковые данные, поэтому элементы массивы различаются по своим порядковым номерам.

Массивы

- Если каждый элемент имеет один порядковый номер, то такой массив называется одномерным, если два – то это таблица из строк и столбцов.
- Для таблиц первый номер элемента показывает строку, а второй – столбец, на пересечении которых находится элемент.
- Все строки таблицы имеют одинаковую длину.
- Одномерный массив может быть числовой последовательностью с известным количеством членов.
- Так же, как и в последовательности, в массиве можно указать элемент с конкретным номером, например a_5, или записать общий вид элемента, используя в качестве индекса переменную и указывая диапазон ее изменения: a_i, i=1,2,…, n.

Задачи на обработку массивов могут иметь различную формулировку.

Например, начинаться со слов: **«Дано n чисел…»**, а далее говорится, что требуется сделать с этими числами.

Чтобы решить такую задачу на компьютере с использованием языка программирования Паскаль, необходимо выполнить следующее:

1. **определить, какие числа даны**: целые или вещественные (если об этом конкретно не сказано, то лучше считать их вещественными);
2. **назвать весь массив одним именем**, которое будет использоваться для каждого элемента, только к нему добавится номер этого элемента (**индекс**);
3. **описать массив в разделе переменные** **var**, тем самым отведя место в памяти для массива;
4. **Ввести данные в память**.

В описании массива

- В описании массива имеется специальное слово **array** (массив), после которого в квадратных скобках через две точки указывается диапазон изменения номеров элементов, затем слово **of** (из) и пишется тип данных массива.
- Встретив описание массива, транслятор отводит для него столько последовательных ячеек, сколько указано в квадратных скобках, и такого формата, каков тип данных массива.
- Эту память в программе можно использовать целиком или частично, вычисляя значения элементов массива или вводя их с клавиатуры (либо с диска).
- Чаще всего номера элементов меняются от 1 до заданного числа *n*.
- Поместив значение n в разделе констант (**const**), в описании можно указать в качестве переменной *n* последнее значение (верхнюю гран

```
const n=10;
var a: array [1..n] of real;
```

Пример описания:

```
const n=10;
var a: array [1..n] of real;
```

- Это описание означает, что для массива *a* будет отведено десять ячеек оперативной памяти по шесть байтов каждая. Имена ячеек a_1, a_2,…, a_{10}. В паскале эти имена будут записаны следующим образом: *a*[1],…, *a*[10].
- В описании после имени массива a ставится двоеточие, за которым указывается тип данного – массив. Если в программе несколько массивов одного размера и типа, то, как и для простых переменных, их имена можно перечислить через запятую, а потом, после двоеточия, указать описание массива.

Для ввода данных в память необходимо организовать цикл.

- Поскольку число повторений ввода данных известно, удобно использовать цикл **пересчет**. Ввод описанного массива a может иметь вид:

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

- Вводимые значения набираются на клавиатуре **через пробел** и нажимается <**Enter**>.
- Можно ввод прокомментировать и вводить каждое данное на отдельной строке экрана (см. программу E12 modif):

```
for i:=1 to n do
begin
write('a[',I, ']=');
readln (a[i]);
end;
```

Обработка массивов

- При обработке массивов решение многих задач основывается на следующих, более простых, задачах:
  - вычисление суммы (произведения) элементов массива;
  - нахождение наибольшего (наименьшего) элемента;
  - упорядочение элементов по возрастанию или убыванию.
- Рассмотрим эти базовые задачи.

Базовые задачи

Задача 1.

- Вычисление суммы элементов массива ничем не отличается, в принципе, от суммирования значений простых переменных (программа E11).
- Решение задачи состоит из трех основных этапов:
  1) **ввод данных;**
  2) **вычисление суммы;**
  3) **печать результатов.**

Вычисление суммы элементов массива.

```
program E12;
const  n=7;
var a: array [1..n] of real; s:real; i: integer;
begin
 write('вводите элементы массива - ', n,
     'вещественных чисел через
     пробел');
for i:=1 to n do
read (a[i]);
S:=0;
for i:=1 to n do
s:=s+a[i];
writeln;
write ('сумма элементов массива S=',
     S);
end.
```

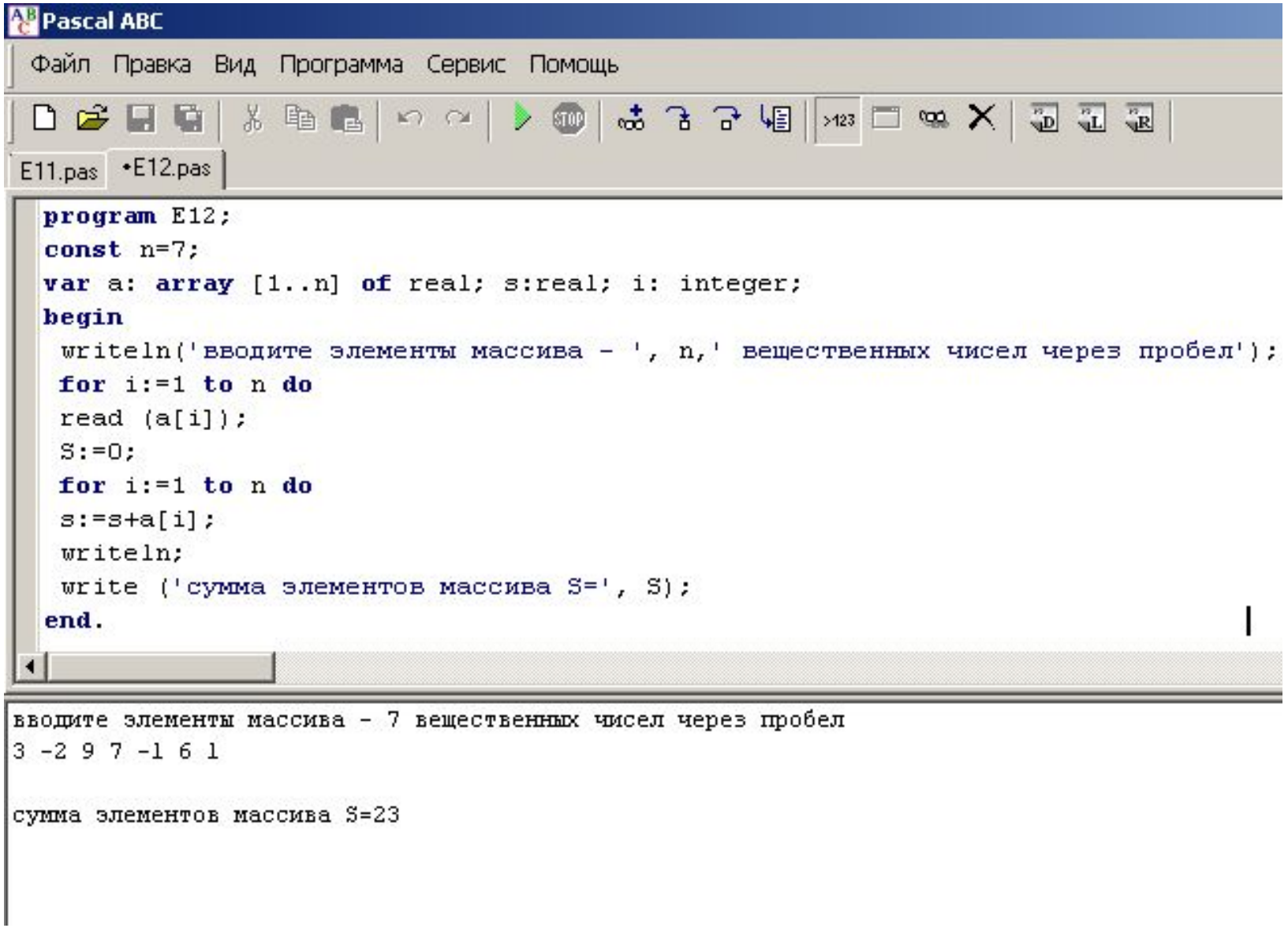
Базовые задачи

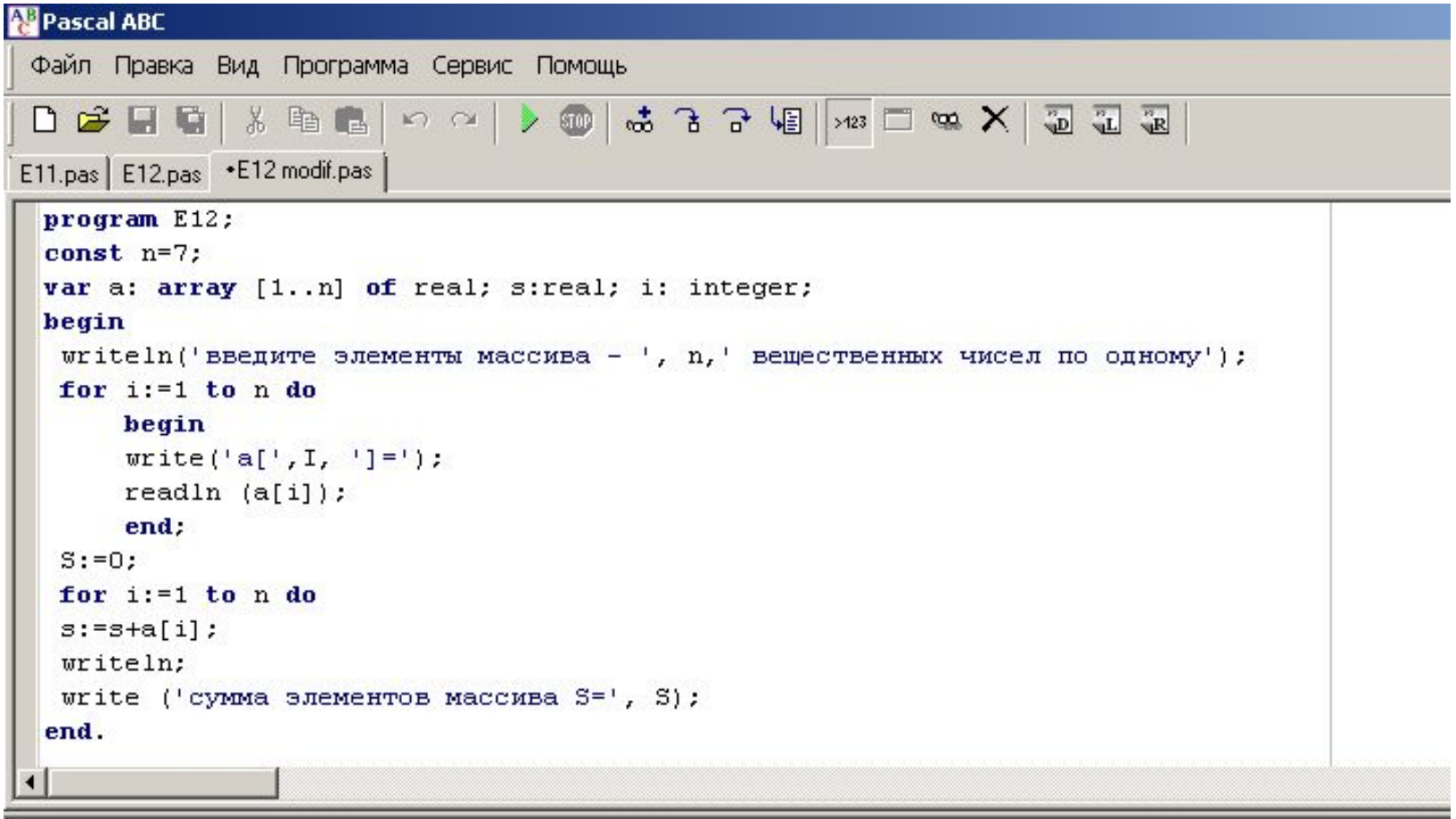
Задача 1.

- Вычисление суммы элементов массива.

Выполнение программы вычисления суммы элементов массива предоставлено в таблице:

Исходные данные: 3, -2, 9, 7, -1, 6, 1							
i	1	2	3	4	5	6	7
a[i]	3	-2	9	7	-1	6	1
S	0	1	10	17	16	22	23


Pascal ABC
Файл Правка Вид Программа Сервис Помощь
E11.pas
•E12.pas
program E12;
const n=7;
var a: array [1..n] of real; s:real; i: integer;
begin
writeln('вводите элементы массива - ', n,' вещественных чисел через пробел');
for i:=1 to n do
read (a[i]);
S:=0;
for i:=1 to n do
s:=s+a[i];
writeln;
write ('сумма элементов массива S=', S);
end.
вводите элементы массива - 7 вещественных чисел через пробел
3 -2 9 7 -1 6 1
сумма элементов массива S=23


Pascal ABC
Файл Правка Вид Программа Сервис Помощь
E11.pas E12.pas *E12 modif.pas
program E12;
const n=7;
var a: array [1..n] of real; s:real; i: integer;
begin
writeln('введите элементы массива - ', n,' вещественных чисел по одному');
for i:=1 to n do
begin
write('a[',I, ']=');
readln (a[i]);
end;
S:=0;
for i:=1 to n do
s:=s+a[i];
writeln;
write ('сумма элементов массива S=', S);
end.
введите элементы массива - 7 вещественных чисел по одному
a[1]=3
a[2]=-2
a[3]=9
a[4]=7
a[5]=-1
a[6]=6
a[7]=1
сумма элементов массива S=23

Базовые задачи

Задача 2.

2. Нахождение наибольшего элемента.

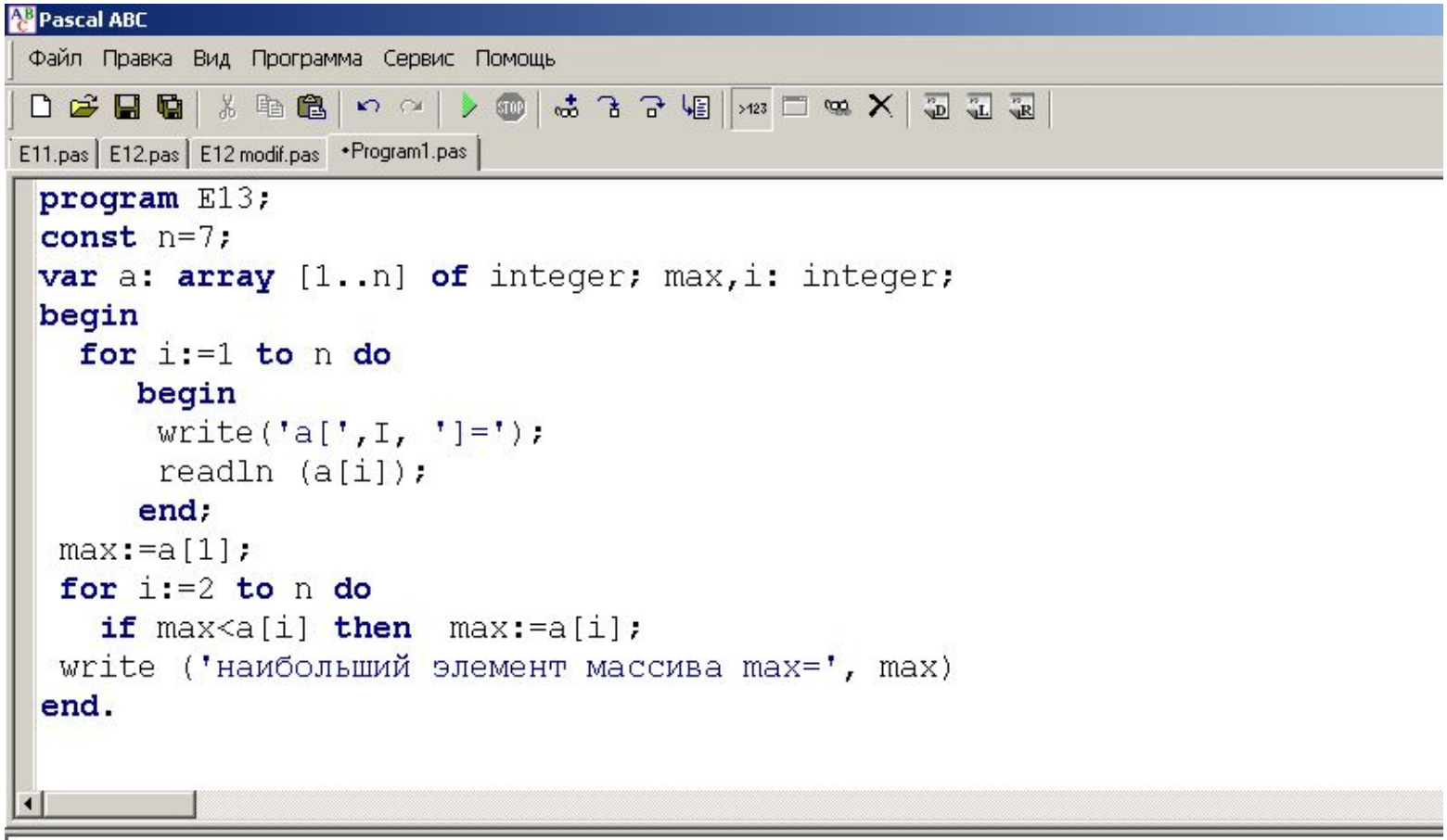
- В предыдущем примере производились вычисления, переменная *S* меняла свои значения в процессе решения задачи.
- Однако большинство задач, решаемых с помощью компьютера, являются **невычислительными**.
- К ним относится **задача поиска наибольшего элемента в массиве.**
- Трудность при разработке алгоритма решения заключается в том, что надо описать в виде команд компьютеру привычные для человека действия: **выделение большего из последовательности чисел.**

Базовые задачи

Задача 2.

2. Нахождение наибольшего элемента.

- Чтобы лучше представить себе, как последовательно просматривать и сравнивать между собой числа, записанные в памяти, вообразим, что каждое число написано на отдельной карточке и карточки сложены стопкой.
- В таком случае мы **первое число запомним сразу как наибольшее** и перевернем карточку.
- Теперь в нашем распоряжении два числа: одно видим, другое – помним. Сравнивая их между собой, запомним большее, т.е. если первое было больше, то запоминать новое не придется и надо смотреть следующую карточку.
- Если второе больше первого, то первое в дальнейшем помнить нет смысла и мы запомним второе.
- Таким образом, на каждом этапе сравнения мы будем помнить большее из просмотренных чисел и в конце решим задачу. Записав приведенные рассуждения в виде операторов, получим программу нахождения наибольшего значения. Промежуточные значения и ответ содержит переменная **max**.




```
a[1]=3
a[2]=-2
a[3]=9
a[4]=7
a[5]=-1
a[6]=6
a[7]=1
наибольший элемент массива max=9
```

Базовые задачи

Задача 3.

Упорядочение массива по возрастанию.

- Упорядочения массивов по какому-либо признаку называется также **сортировками**.
- Существуют различные методы сортировок, различающиеся, в основном, **по скорости получения результата**.
- Рассмотрим один из них – **«метод пузырька».**
- Пусть имеется последовательность чисел **a1, a2,…, an**, которую необходимо упорядочить по возрастанию.
- **Зафиксируем первый элемент** и будем последовательно сравнивать его со стоящими справа.

Базовые задачи

Задача 3.

Упорядочение массива по возрастанию.

- Если какой-то из элементов справа окажется меньше первого, то мы поменяем местами этот элемент с первым и продолжим сравнение уже нового элемента, стоящего на первом месте, с оставшимися справа числами.
- Если снова выявится элемент, меньшей зафиксированного, то повторим перестановку.
- В результате первого просмотра последовательности на первом месте окажется наименьший из всех элементов, т. е. он, как более «легкий», как бы всплывает наверх (отсюда и название метода – «метод пузырька»).
- Теперь зафиксируем второй элемент и повторим просмотр, выполняя при необходимости перестановки элементов, и т.д.

Базовые задачи

Задача 3.

Упорядочение массива по возрастанию.

- Уяснив идею решения, остановимся на двух вопросах: **каким образом фиксировать элементы и как осуществить перестановку двух элементов?**
- Чтобы при переборе элементов, стоящих справа от проверяемого, не менялся индекс последнего, индексы фиксируемого и стоящих правее него элементов должны быть различными: ***i* и *j*.**
- Индекс ***i*** изменяется **от 1 до *n*–1**, индекс ***j*** всегда больше ***i*** и пробегает все значения **от *i*+1 до *n***.
- Для каждого значения ***i*** индекс ***j*** должен последовательно принять все допустимые значения, *следовательно*, конструкция программы, отражающая полный перебор всех элементов и их упорядочение по возрастанию, представляет **двойной цикл**.
- **При перестановке двух элементов местами используется третья переменная.**
- **Перестановка местами (обмен значениями в памяти) двух переменных *a* и *b* выглядит следующим образом: 1) *c*: = *a*; 2) *a:* = *b*; 3) *b*: = *c*.**

Программа сортировки методом пузырька имеет вид:

```
program E14;
const n=7;
var a: array [1..n] of real; i,j: integer; c:real;
begin
 for i:=1 to n do
  begin
   write('a[',I, ']=');
   readln (a[i]);
  end;
 for i:=1 to n-1 do
  for j:=i+1 to n do
   if a[i]>a[j]
    then  begin
            c:=a[i];
            a[i]:=a[j];
            a[j]:=c
          end;
writeln ('упорядоченный по возрастанию массив');
for i:=1 to n-1 do
write ('  ',a[i]);
end.
```

```
a[1]=3
a[2]=-2
a[3]=9
a[4]=7
a[5]=-1
a[6]=6
a[7]=1
упорядоченный по возрастанию массив
  -2  -1  1  3  6  7
```

Базовые задачи

Задача 4. **Поиск элемента в массиве.**

- Одна из важных невычислительных задач – **поиск данного значения среди элементов массива.**
- Такой поиск называется также поиском по ключу.
- На практике поиск осуществляется в упорядоченном массиве, причем имеются различные алгоритмы поиска.
- В данном примере осуществим **поиск путем сплошного перебора.**
- Если элемент найден, то напечатаем его номер, если нет, то выдадим соответствующее сообщение.
- Существенным является то, **какой из одинаковых элементов массива, равных данному нас интересует: первый встретившийся при поиске или последний.**

Базовые задачи

Задача 4. **Поиск элемента в массиве.**

- В этом примере **будем искать первый**.
- Поиск осуществляется в **цикле**, и как только элемент найден, надо выйти из цикла.
- Для досрочного выхода из цикла **for** используем оператор **goto**.
- Если досрочный выход не произойдет, то значит, элемент, равный данному, в массиве отсутствует.
- В таком случае выдача сообщения об отсутствии элемента происходит сразу после цикла поиска.
- Следовательно, чтобы обойти печать номера элемента, надо использовать еще один оператор **goto** и еще одну **метку** в программе.

Программа поиска данного элемента в массиве:

```
program E15;
label 1,2;
const n=7;
var a: array [1..n] of real; x:real; i: integer;
begin
 writeln('введите элементы массива');
 for i:=1 to n do
 read(a[i]);
 writeln;
 write('введите число для поиска в массиве, x = ');
 readln (x);
 for i:=1 to n do
   if a[i]=x then goto 1;
 writeln ('такого числа в массиве нет');
 goto 2;
1:write ('номер элемента массива, равного данному, равен  ',i);
2:end.
```

```
введите элементы массива
3 -2 9 7 -1 6 1

введите число для поиска в массиве, x = 1
номер элемента массива, равного данному, равен  7введите элементы массива
3 -2 9 7 -1 6 1

введите число для поиска в массиве, x = 5
такого числа в массиве нет
```

Если искать не первый по порядку равный ключу элемент, а последний, то надо использовать цикл обратного пересчета: **for** i: = n **downto** 1 **do**.

```
program E15;
label 1,2;
const n=7;
var a: array [1..n] of real; x:real; i: integer;
begin
 writeln('введите элементы массива');
 for i:=1 to n do
 read(a[i]);
 writeln;
 write('введите число для поиска в массиве, x = ');
 readln (x);
 for i:=n downto 1 do
   if a[i]=x then goto 1;
 writeln ('такого числа в массиве нет');
 goto 2;
1:write ('номер элемента массива, равного данному, равен  ',i);
2:end.
```

Вопросы и задания

1. Чем отличается массив от файла?
2. Для чего необходимо описание массива?
3. Что надо сделать, чтобы начать решать на компьютере задачу, формулировка которой начинается со слов: «Дано *n* чисел…»?
4. Может ли массив содержать разнородные данные?
5. Можно ли в примере программы E12 ограничиться одним оператором цикла?
6. Что надо изменить в программе E13, чтобы осуществлялся поиск не наибольшего, а наименьшего элемента массива?
7. Какие изменения в программу E13 надо внести, чтобы одновременно со значением наибольшего числа определялся его порядковый номер?
8. Объясните работу двойного цикла в программе E14.
9. Измените программу E15 так, чтобы вместо цикла **пересчет** при поиске элемента использовался цикл **пока**. Примените переменную-флажок, которая до цикла имела бы нулевое значение, а в случае нахождения необходимого элемента изменила бы значение на 1. Как при этом обойтись без операторов **goto**?

Вопросы и задания

10. В заданной последовательности целых чисел определите количество и сумму элементов, кратных 10.
11. Дано *n* чисел. Найдите сумму чисел, больших заданного числа *a*.
12. В заданном массиве замените нулем наибольший элемент.
13. Найдите полупроизведение всех положительных элементов массива.
14. Найдите сумму квадратов неотрицательных элементов и количество положительных чисел в заданном целочисленном одномерном массиве.
15. В заданной вещественной последовательности поменяйте местами первый и наименьший элементы.
16. Дано *n* чисел. Замените все отрицательные числа их модулями.
17. Вычислите среднее арифметическое наибольшего и наименьшего из *n* чисел.

Литература

- **А.А.Кузнецов, Н.В.Ипатова**
  «Основы информатики», 8-9 кл.:
  - Раздел 3. ОСНОВЫ ПРОГРАММИРОВАНИЯ, С.108-114