

2. Добавьте в таблицу свои значения n и a .
 3. Попробуйте подобрать такие значения элементов массива, чтобы $S = P$, для $n = 2, 5$.
 4. Для $n = 10$ ввели все элементы массива, равные 9. Какой результат получили? Почему? Что нужно исправить в программе для получения правильного результата?
- 2 Для задачи из примера 4.3 добавьте вывод среднего балла.
 - 3 В ходе хоккейного матча удалялись игроки обеих команд. Для каждого удаленного игрока известно время его отсутствия на поле. Определите, какая из команд провела больше времени на скамейке штрафников.
 - 4 Для задачи из примера 4.5 выполните следующее задание:
Введите число 5557. Почему появилась ошибка? Дополните массив констант простыми числами так, чтобы программа могла выдавать ответ для чисел, меньших 10 000. (Для этого можно воспользоваться самой программой или таблицей простых чисел.)
 - 5 Для задачи из примера 4.6 выполните перечисленные задания.
 1. Внесите в программу изменения так, чтобы цвет сектора выбирался из массива констант.
 - 2*. Измените программу так, чтобы диаграмма всегда строилась в центре графического окна. Диаметр круга определяется меньшей из двух величин — шириной или высотой окна.
 - 6* В массивах x и y хранятся координаты точек. Постройте многоугольник, заданный этими координатами. Запросите у пользователя номера двух точек и постройте диагональ многоугольника, соединяющую эти точки.

§ 5. Поиск элементов с заданными свойствами

Человек постоянно сталкивается с задачами поиска требуемой информации. Типичным примером может служить работа со справочниками или библиотечной картотекой. В современном мире информацию ищут с использованием сети Интернет.

Чтобы поиск был результативным и быстрым, разрабатывают эффективные алгоритмы поиска. Важную роль в процессе поиска информации играет способ хранения данных. Одной из самых простых структур для этого является массив.

5.1. Линейный поиск

Рассмотрим, как осуществляется поиск для данных, хранящихся в массиве.

Среди разновидностей простейших задач поиска, встречающихся на практике, можно выделить следующие типы:

1. Найти хотя бы один элемент, равный заданному элементу X . В результате необходимо получить i —

индекс (номер) элемента массива, такой, что $a[i] = X$.

2. Найти все элементы, равные заданному X . В результате необходимо получить количество таких элементов и (или) их индексы.

Иногда поиск организуется не по совпадению с элементом X , а по выполнению некоторых условий. Примером может служить поиск элементов, удовлетворяющих условию: $X_1 \leq a[i] \leq X_2$, где X_1 и X_2 заданы.

Если нет никакой добавочной информации о разыскиваемых данных, то самый простой подход — последовательный просмотр элементов массива.

Алгоритм, при котором для поиска нужного элемента последовательно просматривают все элементы массива в порядке их записи, называется **линейным** или **последовательным** поиском.

5.2. Поиск одного элемента, удовлетворяющего условию поиска

Пример 5.1. Задан одномерный массив из n чисел. Определить, есть ли в нем хотя бы один элемент, равный x (значение x вводится).

I. Исходные данные: массив a , количество чисел n , искомое число x .

II. Результат: вывод сообщения «Элемент найден» или «Элемент не найден».

III. Алгоритм решения задачи.

1. Ввод исходных данных.

2. Пусть p — переменная логического типа, которая имеет значение «истина», если элемент

Алгоритмы поиска можно разделить на алгоритмы, использующие упорядоченные наборы данных, и на алгоритмы, работающие с предварительно упорядоченным набором данных.

Примером поиска в неупорядоченном наборе данных может служить поиск тетради конкретного учащегося в стопке тетрадей, сданных на проверку. Чтобы найти нужную тетрадь, возможно, придется пересмотреть все. Поиск в словаре — поиск в упорядоченном наборе данных, т. к. все слова расположены в алфавитном порядке.

Пример 5.1.

V. Программа:

```
var a: array[1..10] of integer;
    n, x: integer;
    p: boolean;
begin
    write('Количество n =');
    readln(n);
    writeln('Элементы массива');
    for var i := 1 to n do
        read(a[i]);
    write('Число x =');
    readln(x);
    //линейный поиск элемента
    p := false;
    for var i := 1 to n do
        if a[i] = x then
            p := true;
    if p then
        writeln('Элемент найден')
    else
        writeln('Элемент не найден');
end.
```

VI. Тестирование.

Окно вывода

```
Количество n = 5
Элементы массива
1 2 3 4 5
Число x = 4
Элемент найден
```

Окно вывода

```
Количество n = 5
Элементы массива
1 3 5 7 9
Число x = 6
Элемент не найден
```

Пример 5.2.

V. Программа:

```

var a: array[1..10] of integer;
    n, x, k: integer;
begin
  write('Количество n=');
  readln(n);
  writeln('Элементы массива');
  for var i := 1 to n do
    read(a[i]);
  write('Число x =');
  readln(x);
  //линейный поиск элемента
  k := 0;
  for var i := 1 to n do
    if a[i] = x then
      k := i;
  if k = 0 then
    writeln('Элемент не найден')
  else
    writeln('Элемент найден
            на месте ', k);
end.

```

VI. Тестирование.

Окно вывода

```

Количество n = 5
Элементы массива
1 3 3 3 5
Число x = 3
Элемент найден
на месте 4

```

Окно вывода

```

Количество n = 5
Элементы массива
1 3 5 7 9
Число x = 4
Элемент не найден

```

Пример 5.3.

Фрагмент программы:

```

//линейный поиск элемента
k:=1;
while (k<=n) and (a[k]<>x) do
  k:=k+1;
if k = n+1 then
  writeln('Элемент не найден')
else
  writeln('Элемент найден
          на месте ', k);

```

в массиве найден, и «ложь» — в противном случае. До просмотра элементов массива $p := \text{false}$.

3. В цикле будем просматривать все числа в массиве и сравнивать их с числом x .

4. После окончания поиска возможна одна из двух ситуаций:

4.1. Искомый элемент найден ($p := \text{true}$), т. е. в массиве есть такой элемент $a[i]$, что $a[i] = x$.

4.2. Весь массив просмотрен, и совпадений не обнаружено ($p := \text{false}$).

5. Вывод результата.

IV. Описание переменных: a — `array[1..10] of integer`; n, x — `integer`; p : `boolean`.

Часто требуется не только определить, есть ли в массиве искомый элемент, но и установить, на каком месте он находится.

Будем хранить индекс найденного элемента (пример 5.2) в переменной k . После выполнения данного алгоритма по значению переменной k можно определить, есть ли в массиве искомый элемент, и если есть, то где он стоит. Если в массиве несколько таких элементов, то в переменной k будет храниться номер последнего из них. Если такого элемента нет, то значение переменной k не изменится (k останется равным 0).

На практике операцию поиска приходится выполнять достаточно часто, и скорость работы программы находится в прямой зависимости от используемого алгоритма поиска.

В рассмотренных выше алгоритмах требуется просмотреть весь массив, даже в том случае, если искомый элемент находится в массиве на первом месте.

Для сокращения времени поиска можно останавливаться сразу после того, как элемент найден. В этом случае весь массив придется просмотреть только тогда, когда искомый элемент последний или его нет вообще (пример 5.3). Цикл заканчивает работу, когда будет найден искомый элемент либо когда $k = n + 1$, т. е. элемента, совпадающего с x , не существует.

*При такой реализации на каждой итерации цикла требуется увеличивать индекс k и вычислять логическое выражение. Ускорить поиск можно, упростив логическое выражение. Поместим в конец массива дополнительный элемент со значением x . Тогда совпадение с x обязательно произойдет, и можно не проверять условие $a[k] < > x$. Такой вспомогательный элемент часто называют «барьером» или «часовым», так как он препятствует выходу за пределы массива. В исходном массиве теперь будет $n + 1$ элемент (пример 5.4).

5.3. Нахождение всех элементов, удовлетворяющих условию поиска

Если требуется определить количество элементов, удовлетворяющих какому-либо условию, то для этого определяют отдельную переменную, значение которой увеличивают на 1 каждый раз, когда найден нужный элемент. Такую переменную называют счетчиком. До начала просмотра

Пример 5.4*.

Фрагмент программы:

```
//линейный поиск с барьером
a[n + 1] := x;
k := 1;
while a[k]<>x do
  k := k + 1;
if k = n + 1 then
  writeln('Элемент не найден')
else
  writeln('Элемент найден
        на месте ', k);
```

V. Тестирование.

Окно вывода

```
Количество n = 5
Элементы массива
1 5 6 7 1
Число x = 7
Элемент найден
на месте 4
```

Окно вывода

```
Количество n = 5
Элементы массива
1 2 3 4 5
Число x = -2
Элемент не найден
```

В современных языках программирования используются библиотеки, содержащие функции для поиска элементов в массивах (и других структурах данных). В PascalABC.Net такие функции реализованы только для динамических массивов (размер массива может изменяться во время выполнения программы, элементы нумеруются с нуля). Описание функций можно найти в справочнике в разделе «Методы расширения одномерных динамических массивов». Пример использования функции поиска всех элементов массива, больших 5:

```
var c : array of integer;
begin
  setlength(c, 10);
  for var i := 0 to 9 do
    c[i] := random(1, 10);
  println(c);
  c.FindAll(p -> p > 5).Println;
end.
```

Пример 5.5.

V. Программа:

```

var a: array[1..10] of integer;
n, x, k: integer;
begin
write('Количество n = ');
readln(n);
writeln('Элементы массива');
for var i := 1 to n do
  read(a[i]);
write('Число x = ');
readln(x); k := 0;
for var i := 1 to n do
  if a[i] mod x = 0 then
    k := k + 1;
writeln('В массиве ', k, '
элемент(-а,-ов), кратный(-х)', x);
end.

```

VI. Тестирование.

Окно вывода
Количество n = 5
Элементы массива
1 2 3 4 5
Число x = 2
В массиве 2 элемент(-а,-ов),
кратный(-х) 2

Пример 5.6.

V. Программа:

```

var a, b: array[1..10] of integer;
n, x, k: integer;
begin
write('Количество n = ');
readln(n);
writeln('Элементы массива');
for var i := 1 to n do
  read(a[i]);
write('Число x = ');
readln(x); k := 0;
for var i := 1 to n do
  if a[i] mod x = 0 then
    begin
      k := k + 1; b[k] := i;
    end;
writeln('В массиве ', k, '
элемент(-а,-ов), кратный(-х)', x);
writeln('Местоположение ');
for var i := 1 to k do
  write(b[i], ' ');
end.

```

элементов массива счетчику нужно задать начальное значение, или, другими словами, **инициализировать** значение переменной. В случае подсчета количества элементов, удовлетворяющих условию, счетчик инициализируется нулем. Для решения задачи нужно просматривать весь массив.

Пример 5.5. Задан одномерный массив из n чисел. Определить количество элементов, кратных x в линейном массиве.

I. Исходные данные: массив a , количество чисел n , искомое число x .

II. Результат: количество элементов, удовлетворяющих условию, — k .

III. Алгоритм решения задачи.

1. Ввод исходных данных.
2. Инициализация счетчика.

3. В цикле будем просматривать все числа в массиве и сравнивать с нулем их остатки от деления на число x . Если остаток равен нулю, то счетчик увеличиваем на 1.

4. Вывод результата.

IV. Описание переменных: a — `array[1..10] of integer`; n , x , k — `integer`.

Если необходимо не только посчитать, сколько элементов удовлетворяют условию, но и сохранить индексы таких элементов, то для этого можно воспользоваться дополнительным массивом. Создадим новый массив b . Как только будет найден необходимый элемент, его индекс будет заноситься в массив b . Переменная k будет хранить номер последнего занятого места в массиве b . Вначале $k = 0$ (пример 5.6).

После завершения работы первые k элементов массива b будут содержать индексы искоемых элементов.

Если для решения задачи потребуются значения всех найденных элементов, то в программе возможно такое обращение к элементам массива: $a[b[i]]$. Адрес элемента в массиве a будет определяться значением элемента массива b по адресу i . Для вывода значений элементов в примере 5.6 последний цикл нужно заменить на

```
for var i := 1 to k do
  write(a[b[i]], ' ');
```

5.4. Решение задач с использованием алгоритма линейного поиска

Пример 5.7. Известны результаты ЦТ по математике для n человек. Определить, есть ли среди них хотя бы один человек с баллом выше x . Значение x вводится с клавиатуры. Результаты экзамена получить случайным образом.

I. Исходные данные: массив a , количество чисел n , число x .

II. Результат: сообщение соответствует условию задачи.

III. Алгоритм решения задачи.

1. Ввод исходных данных.

2. Просмотр элементов с начала. Как только элемент найден, остановимся.

3. Если весь массив просмотрен, значит, в исходном массиве нет элемента, удовлетворяющего условию задачи, иначе выводим номер найденного элемента.

IV. Описание переменных: a — `array[1..20] of integer`; n , x , k — `integer`.

Пример 5.6. Продолжение.

VI. Тестирование.

Окно вывода

```
Количество n = 5
Элементы массива
6 3 2 4 5
Число x = 2
В массиве 3 элемент (-a, -ов),
кратный (-x) 2
Местоположение
1 3 4
```

Пример 5.7.

V. Программа:

```
var a: array[1..20] of integer;
    n, x, k: integer;
begin
  write('Количество n =');
  readln(n);
  writeln('Элементы массива');
  for var i := 1 to n do
  begin
    a[i] := random(0,100);
    write(a[i], ' ');
  end;
  writeln;
  write('Число x =');
  readln(x);
  //линейный поиск с барьером
  a[n+1] := x + 1;
  k := 1;
  while (a[k] <= x) do
    k := k + 1;
  if k = n+1 then
    writeln('Нет таких')
  else
    writeln('Это человек с № ',
           k, ', его балл - ', a[k]);
end.
```

VI. Тестирование.

Окно вывода

```
Количество n = 10
Элементы массива
48 12 96 48 9 95 5 71 77 24
Число x = 80
Это человек с №3, его балл - 96
```

Пример 5.8.

V. Программа:

```

uses graphABC;
var X,Y: array [1..1000] of
    integer;
    n, k1, k2, R: integer;
begin
write('Количество точек n =');
read(n);
writeln(n);
for var i := 1 to n do
begin
X[i]:= random(-200,200);
Y[i]:= random(-200,200);
end;
writeln('Радиус окружности');
read(R);
writeln(R);
{Построение окружности и
осей координат}
circle(200,200,R);
line(0,200,400,200);
line(200,0,200,400);
k1 := 0; k2 := 0;
for var i := 1 to n do
if X[i]*X[i]+Y[i]*Y[i]<=R*R then
begin
k1 := k1 + 1;
SetPixel(X[i]+200,200-Y[i],
clred);
end
else
begin
k2 := k2 + 1;
SetPixel(X[i]+200,200-Y[i],
clblue);
end;
if k1 > k2 then
writeln('Внутри больше')
else
if k1<k2 then
writeln('Снаружи больше')
else
writeln('Поровну')
end.

```

Пример 5.8. В двух линейных массивах x и y , заданных случайным образом, хранятся координаты точек плоскости ($-200 \leq X[i], Y[i] \leq 200$). Определить, каких точек больше — лежащих внутри или снаружи области, ограниченной окружностью радиуса R с центром в начале координат (будем считать, что точки, лежащие на окружности, лежат внутри области). Построить окружность и точки. Точки, принадлежащие внутренней области, нарисовать красным цветом, а внешней области — синим цветом.

I. Исходные данные: X, Y — массивы чисел, R — радиус окружности, n — количество точек.

II. Результат: рисунок, соответствующий условию задачи, и сообщение: «Внутри точек больше», «Снаружи точек больше» или «Точек поровну».

III. Алгоритм решения задачи.

1. Ввод исходных данных.

2. Инициализация счетчиков:

$k1 := 0; k2 := 0$.

3. Будем просматривать все точки и для каждой проверять принадлежность области. Если $X^2 + Y^2 \leq R^2$, то точка лежит внутри области, тогда увеличим значение счетчика $k1$ на 1, если нет, то увеличим на 1 значение счетчика $k2$.

4. Сравним значения $k1$ и $k2$ и выведем результат.

5. Поскольку координаты точек принадлежат отрезку $[-200, 200]$, то оси координат можно нарисовать пересекающимися в точке с координатами $(200, 200)$. Для преобразования координат точек в экранные нужно к значению аб-

сциссы прибавить 200, а значение ординаты нужно отнять от 200 (ось Y на экране направлена вниз, поэтому нужно поменять знак ординаты). Строить точки можно в том же цикле, в котором происходит проверка.

IV. Описание переменных: X, Y — `array[1..1000] of integer`; $n, k1, k2, R$ — `integer`.

Пример 5.9. На складе хранятся пустые ящики для упаковки товара. Известно, что масса одного пакета с конфетами x кг. Какова суммарная масса пакетов с конфетами, которые можно упаковать в такие ящики, заполнив ящик целиком?

I. Исходные данные: массив a , количество чисел n , число x .

II. Результат: количество ящиков — k , суммарная масса — S .

III. Алгоритм решения задачи.

1. Ввод исходных данных.

2. Инициализация счетчика и значения суммы: $k := 0$; $S := 0$.

3. Просматривая массив, проверим, является ли текущий элемент числом, кратным x (в этом случае ящик будет заполнен целиком). Как только элемент найден, увеличим счетчик k на 1, а переменную S — на значение найденного элемента массива.

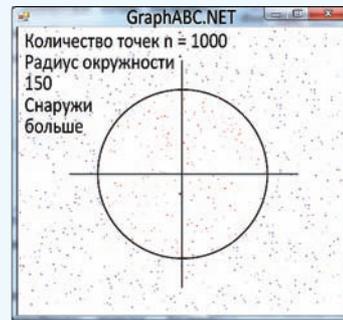
4. Вывод результата.

IV. Описание переменных: a — `array[1..20] of integer`; n, x, k, S — `integer`.

Пример 5.10. Имеется список мальчиков 10 В класса и результаты их бега на 100 м. Для сдачи норматива

Пример 5.8. Продолжение.

VI. Тестирование.



Пример 5.9.

V. Программа:

```
var a: array [1..20] of integer;
    n, x, k, S: integer;
begin
  write('Количество ящиков:');
  readln(n);
  writeln('Вместимость ящиков');
  for var i := 1 to n do
    read(a[i]);
  write('Масса конфет:'); read(x);
  k := 0; S := 0;
  for var i := 1 to n do
    if a[i] mod x = 0 then
      begin
        k := k + 1;
        S := S + a[i];
      end;
  writeln('На складе', k, 'ящ. ');
  writeln('Суммарная масса', S);
end.
```

VI. Тестирование.

```
Окно вывода
Количество ящиков: 8
Вместимость ящиков
15 23 64 27 35 10 48 13
Масса конфет: 5
На складе 3 ящ.
Суммарная масса 60
```

VII. Анализ результатов. Ящики, удовлетворяющие условию задачи, имеют вес — 15, 25 и 10.

Пример 5.10.

V. Программа:

```

var r: array [1..20] of real;
    fam: array [1..20] of string;
    n, k: integer;
begin
  writeln('Количество учащихся: ');
  readln(n);
  writeln('Фамилия и результат: ');
  for var i := 1 to n do
  begin
    readln(fam[i]);
    readln(r[i]);
  end;
  writeln('Фамилии не сдавших
          норматив:');
  k := 0;
  for var i := 1 to n do
  if r[i] > 16 then
  begin
    k := k + 1;
    writeln(fam[i]);
  end;
  writeln('Не сдали норматив:', k);
end.

```

VI. Тестирование.

Окно вывода

```

Количество учащихся:
5
Фамилия и результат:
Иванов
13.5
Петров
14.0
Сидоров
21
Королев
16.1
Веремей
15.9
Фамилии не сдавших норматив:
Сидоров
Королев
Не сдали норматив: 2

```

VII. Анализ результатов. Результат Сидорова 21, а Королева 16.1, что превышает норматив.

необходимо пробежать дистанцию не более чем за 16 с. Вывести фамилии учащихся, которые не выполнили норматив по бегу. Сколько таких учащихся в классе?

I. Исходные данные: массивы `fam` (фамилии учащихся) и `r` (результаты бега в секундах), количество учащихся `n`.

II. Результат: фамилии тех учащихся, которые не выполнили норматив по бегу.

III. Алгоритм решения задачи.

1. Ввод исходных данных.

2. Инициализация счетчика:

`k := 0`.

3. Будем просматривать массив с результатами и проверять, является ли текущий элемент числом больше 16 (норматив не сдан). Если такое значение найдено, то выведем элемент массива `fam` с соответствующим номером и увеличим значение счетчика на 1.

4. Вывод значения счетчика.

IV. Описание переменных: `fam` — `array[1..20] of string`; `r` — `array[1..20] of real`, `n, k` — `integer`.

Пример 5.11*. Задан одномерный массив из N целых чисел. Определить количество элементов, которые являются числами Смита. (Число Смита — это такое составное число, сумма цифр которого равна сумме цифр всех его простых множителей.) Например, числом Смита является $202 = 2 \times 101$, поскольку $2 + 0 + 2 = 4$, и $2 + 1 + 0 + 1 = 4$.

I. Исходные данные: `a` — массив чисел, `n` — количество чисел в массиве.

II. Результат: числа Смита и их количество в массиве.

III. Алгоритм решения задачи.

1. Ввод исходных данных.

2. Инициализация счетчика:
k := 0.

3. Будем просматривать каждый элемент массива и определять, является ли он числом Смита. Для проверки создадим функцию check, которая будет получать в качестве параметра элемент массива, а также возвращать значение true, если число является числом Смита, и false в противном случае.

3.1. Найдем сумму цифр числа.

3.2. Будем раскладывать число на простые множители и для каждого множителя находить сумму цифр.

3.3. Для разложения числа на простые множители будем делить его сначала на 2 (пока делится), затем на 3. На 4 число уже делиться не будет, будем делить его на 5 и т. д. Закончится разложение тогда, когда после всех делений число станет равным 1.

4. Также нам понадобится функция sum, которая для числа будет возвращать его сумму цифр.

IV. Описание переменных: a — `array[1..100] of integer`; n, k — `integer`.

Пример 5.12*. Задан одномерный массив из n строк. Каждая строка является предложением из слов, разделенных пробелами. Найти и вывести те предложения, в которых нечетное количество слов.

Пример 5.11*.

V. Программа:

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

function sum(x: integer):
    integer;
var s: integer;
begin
    s := 0;
    while x > 0 do
        begin
            s := s + x mod 10; x := x div 10;
        end;
    sum := s;
end;

function check(x: integer):
    boolean;
var s1, s2, d: integer;
begin
    s1 := sum(x); s2 := 0; d := 2;
    //разложение на простые множители
    while x <> 1 do
        begin
            while x mod d = 0 do
                begin
                    s2 := s2 + sum(d);
                    x := x div d;
                end;
            d := d + 1;
        end;
    check := s1 = s2;
end;

begin
    writeln('Количество');
    readln(n);
    writeln('Элементы');
    for var i := 1 to n do
        read(a[i]);
    k := 0;
    writeln('Числа Смита');
    for var i := 1 to n do
        if check(a[i]) then
            begin
                inc(k);
                write(a[i], ' ');
            end;
    writeln;
    writeln('Всего - ', k);
end.
```

Пример 5.11*. *Продолжение.*
VI. Тестирование.

Окно вывода	Окно вывода
Количество 5	Количество 5
Элементы 202 3 323 85 117	Элементы 8 8 8 8 8
Числа Смита 202 3 85	Числа Смита
Всего - 3	Всего - 0

Пример 5.12.
V. Программа:

```
var a: array [1..100] of
    string;
    n, k: integer;

function check(x: string):
    integer;

var
    s, len: integer;
begin
    s := 0;
    x := ' ' + x;
    len := length(x);
    for var i := 1 to len - 1 do
        if (x[i] = ' ') and (x[i + 1] <> ' ')
        then
            inc(s);
        check := s;
    end;
begin
    writeln('Количество');
    readln(n);
    writeln('Элементы');
    for var i := 1 to n do
        readln(a[i]);
    k := 0;
    writeln;
    writeln('Искомые строки:');
    for var i := 1 to n do
        if check(a[i]) mod 2 <> 0 then
            begin
                inc(k);
                writeln(a[i]);
            end;
    writeln('Всего - ', k);
end.
```

I. Исходные данные: a — массив строк, n — количество строк в массиве.

II. Результат: искомые строки и их количество.

III. Алгоритм решения задачи.

1. Ввод исходных данных.

2. Инициализация счетчика:

$k := 0$.

3. Будем просматривать каждую строку и определять, сколько в ней слов. Для проверки создадим функцию `check`, которая будет получать в качестве параметра элемент массива и возвращать количество слов в строке. Если количество слов является нечетным числом, то выведем строку и увеличим значение счетчика.

3.1. Перед каждым словом предложения, кроме первого, стоит пробел, слово начинается с символа, который пробелом не является.

3.2. Добавим пробел перед первым словом, тогда количество слов будет определяться количеством сочетаний пар символов: пробел и не пробел.

IV. Описание переменных: a — `array[1..100] of string`; n, k — `integer`.

При вводе данных для тестирования программы нужно помнить, что после каждого предложения необходимо нажимать клавишу `Enter`.

В данном случае строка как тип данных может не соответствовать строке в окне вывода. В примере 5.12 вводятся 3 строки:

1. «Многие компании разрабатывают свои правила по оформлению кода».

2. «В них прописаны также правила именования переменных».

3. «Компания Microsoft использует так называемую «венгерскую нотацию».

В окне вывода количество строк может быть больше (на рисунке их 6).

То, как будут выглядеть вводимые строки в окне вывода, зависит от ширины окна приложения PascalABC.NET. На большом мониторе, если окно приложения развернуто на весь экран, строк может быть три.

Компилятор определяет, что ввод строки закончен, если была нажата клавиша Enter. Внешний вид строк в окне вывода для компилятора не имеет значения.

Пример 5.12. Продолжение.

VI. Тестирование.

Окно вывода

Количество

3

Элементы

Многие компании разрабатывают свои правила по оформлению кода.

В них прописаны также правила именования переменных.

Компания Microsoft использует так называемую «венгерскую нотацию».

Искомые строки

В них прописаны также правила именования переменных.

Компания Microsoft использует так называемую «венгерскую нотацию».

Всего – 2



1. Что называют последовательным поиском?

2. Как определить, что в массиве был найден элемент с определенными свойствами?

3. Для чего используют переменные «счетчики»?

4. Что такое инициализация переменной?



Упражнения

1 Для примеров 5.1—5.3 выполните перечисленные задания.

1. Заполните таблицу.

№	n	Массив	x	Результат
1	5	2 14 7 20 16	20	
2	5	2 3 4 5 6	8	
3	7	2 4 6 8 10 11 12	8	
4	5	6 3 9 12 15	3	

2. Добавьте в таблицу свои данные, такие, чтобы искомый элемент был первым в массиве; последним в массиве.

3. Какой ответ выдаст каждая из программ, если в массиве несколько элементов, удовлетворяющих условию задачи? Почему?

4. Что нужно изменить в программе 5.2, чтобы выдавался не последний из найденных элементов, а первый?

5. Что нужно изменить в программе 5.1, чтобы выдавался не последний из найденных элементов, а первый?
6. Измените условие цикла `while` примера 5.3 так, чтобы использовалась логическая операция `not`.
- 2 Рост учащихся класса представлен в виде массива. Определите количество учащихся, рост которых больше среднего роста по классу.
- 3 Заданы фамилии и рост учащихся 10-го класса. Вывести фамилии тех учащихся, рост которых меньше среднего роста по классу.
- 4 Известны данные о площади n стран (в млн кв. км) и численности населения (в млн жителей). Выведите номера тех стран, плотность населения которых больше x .
- 5 Для упражнения 4 добавьте возможность вводить и выводить названия стран.
- 6 Определите, есть ли в линейном массиве хотя бы один элемент, который является нечетным числом, кратным 7. Если да, то следует вывести его номер.
- 7* В линейном массиве найдите и выведите все простые числа с нечетной суммой цифр. Укажите, сколько чисел вывели.
- 8* В линейном массиве найдите и выведите все числа Армстронга. (Числом Армстронга называется такое число, которое равно сумме своих цифр, возведенных в степень, равную количеству его цифр. Например, числом Армстронга является число $371 : 371 = 3^3 + 7^3 + 1^3 = 27 + 343 + 1$.) Укажите, сколько чисел вывели.
- 9 Задан одномерный массив из n строк. Каждая строка является предложением из слов, разделенных пробелами. Найдите и выведите те предложения, в которых есть слова, начинающиеся на гласную (строчную или прописную).

§ 6. Максимальный и минимальный элементы массива

Пример 6.1.
V. Программа:

```
var a: array[1..20] of integer;
n, max: integer;
begin
  write('Количество n = ');
  readln(n);
  writeln('Элементы массива');
  for var i := 1 to n do
    read(a[i]);
  max := a[1];
  for var i := 2 to n do
    if a[i] > max then
      max := a[i];
  writeln('Максимум = ', max);
end.
```

6.1. Поиск максимального (минимального) элемента в массиве

Очень часто для решения задачи требуется находить не заданный элемент массива, а максимальный (наибольший) или минимальный (наименьший).

Рассмотрим задачу нахождения максимального элемента. Если в массиве один-единственный элемент, то он и есть максимальный. Если элементов больше одного, то максимальным в массиве из i элементов является максимум из $a[i]$ и максимального