Коренская Ирина Николаевна
старший преподаватель
Институт информационных технологий
УО «Белорусский государственный
университет информатики и радиоэлектроники»
г. Минск, Республика Беларусь

МЕТОДИЧЕСКИЕ УКАЗАНИЯ К ПРАКТИЧЕСКОМУ ЗАНЯТИЮ ПРИ ИЗУЧЕНИИ ОСНОВНЫХ АЛГОРИТМОВ РАБОТЫ С ОДНОМЕРНЫМИ МАССИВАМИ

Аннотация*: в статье подробно рассматриваются этапы решения задач на примере алгоритмов вычисления суммы, произведения и количества элементов одномерного массива с заданными характеристиками. Особое внимание уделяется детализированной записи алгоритма и реализации его на языке программирования. Приведены возможные условия отбора элементов массива. Указаны правила составления программ.*

Ключевые слова*: алгоритмы, одномерные массивы, элементы массива, индекс, размер, сумма, счетчик, произведение, инициализация, условия проверки, особенности реализации, правила составления программ.*

Рассмотрим базовые алгоритмы вычисления суммы, количества и произведения элементов массива и реализуем их на языке высокого уровня Паскаль.

Вначале составим алгоритм вычисления суммы и количества, например для отрицательных и положительных элементов массива [1].

Алгоритм решения задачи включает следующие действия:

1. Ввод количества элементов массива (размер Size).

2. Ввод значений элементов массива Arr[i].

3. Вывод значений элементов массива Arr[i].

4. Инициализация сумм (Sum_p=0; Sum_o=0;) и количества (Count_p=0; Count_o=0;).

5. Просмотр всех элементов массива с первого (индекс 1) до последнего (индекс Size):

5.1. Если элемент массива Arr[i] положительный, то следует увеличить:

5.1.1. Счетчик количества положительных элементов Count_p на 1.

5.1.2. Сумму положительных элементов Sum_p на значение данного элемента массива Arr[i].

5.2. Если элемент массива отрицательный, то увеличить:

5.2.1. Счетчик количества отрицательных элементов Count_o на 1.

5.2.2. Сумму отрицательных элементов Sum_o на значение данного элемента массива Arr[i].

6. Вывод полученных значений счетчиков (Count_p и Count_o) и сумм (Sum_p и Sum_o).

Перед составлением программы введем обозначения используемых в ней переменных:

N_Max – максимальный размер массива, опишем его как константу;

Arr – одномерный массив элементов;

Size – размер массива (количество его элементов);

i – индекс (порядковый номер) элемента массива;

Count_p – количество положительных элементов массива;

Coun_o – количество отрицательных элементов массива;

Sum_p – сумма положительных элементов массива;

Sum_ o – сумма отрицательных элементов массива.

Рассмотрим особенности реализации алгоритма на языке Паскаль.

Так как по условию задачи элементы массива могут быть любыми числами (как дробными, так и целыми), то тип элементов будет вещественный (Real) [1]. Значит, для получения на экране числа с фиксированной, а не с плавающей формой представления надо указать при выводе значений сумм их формат, например, :-6:2, где ‘-’ означает выравнивание числа по левому краю, 6 – общее число позиций вывода под значение, :2 – точность вывода числа. Значения счетчиков и сумм при выводе отделяются друг от друга тремя пробелами («’:3).

С учетом рассмотренного выше приведем код программы на Паскале [1; 2].

```
Program Array_1;
Const      // описание максимального размера массива:
 N_Max = 100; // количество элементов массива не может быть больше 100
Type // описание типа массива действительных чисел
 Mas=Array [1..N_Max] Of Real;
Var
 Arr: Mas; // описание массива
i, Size, Count_p, Count_o: Byte; {индекс i, размер size и счетчики Count_p и Count_o неотрицательны и не больше 100, поэтому для них выбирается тип Byte, значения которого могут изменяться от 0 до 255}
Sum_p, Sum_o: Real; {суммы Sum_p и Sum_o являются результатом сложения элементов массива, значит, их тип будет совпадать с типом массива}
Begin
   Write («Введите размер массива: ’);      // вывод пояснения
   Read (size);     // ввод количества элементов массива
   WriteLn («Введите элементы массива:’); // вывод пояснения
   For i := 1 To Size Do   // ввод значений элементов массива
      Begin
      Write («Arr[’, i, ‘]=’); //вывод пояснения «Arr[индекс элемента]=»
      Read (Arr[i]); // ввод значения элемента массива с клавиатуры
      End;
WriteLn;      // переход на новую строку
WriteLn («Массив Arr:’); // вывод пояснения «Массив Arr:»
For i := 1 To Size Do   // вывод элементов массива в строку
   Write (Arr [i], «’:3); // через 3 пробела («’:3)
Sum_P:=0; // инициализация переменных: сумм и счетчиков элементов
Count_p := 0; // (они начинают вычисляться с нуля)
Sum_o := 0;
```

```
Count_o := 0;
For i :=1 To Size Do   // подсчет сумм и количества элементов
  Begin
    If Arr [i] > 0 Then // если элемент массива положительный, то:
      Begin
        Inc(Count_p); // счетчик положительных элементов увеличивается на 1,
        Sum_p:=Sum_p+Arr[i]; // в Sum_p добавляется элемент Arr[i]
  End;
If Arr [i] < 0 Then // если элемент массива отрицательный, то:
  Begin
    Inc(Count_o); // счетчик отрицательных элементов увеличивается на 1,
    Sum_o:=Sum_o+Arr[i]; // в Sum_o добавляется элемент Arr[i]
  End;
 End; // конец цикла For i
WriteLn;      // переход на новую строку
WriteLn («Count_p = ’,Count_p,‘ ‘:3,»Sum_p = ‘,Sum_p:-6:2); // вывод
WriteLn («Count_o = ’,Count_o,‘ ‘:3,»Sum_o = ‘,Sum_o:-6:2); // результатов
ReadLn; // организация паузы на экране до нажатия любой клавиши
End.
```

Вместо заданных условий проверки элементов на положительность (Arr[i]>0) и отрицательность (Arr[i]<0) в зависимости от поставленной задачи могут рассматриваться другие условия, например:

Arr [i] >= 0 – элемент массива неотрицательный;

Arr [i] <> 0 – элемент массива, не равный 0;

Arr [i] = 0 – элемент массива, равный 0;

Arr [i] = t – элемент массива, равный t (значение t вводится с клавиатуры);

(Arr [i] >= a) And (Arr [i] <=b) – элемент массива принадлежит [a, b];

(Arr [i]< a) Or (Arr [i] >b) – элемент массива не принадлежит [a, b].

Если элементы массива описаны как целые числа, например, имеют тип Integer, то можно добавить к вышеперечисленным условиям проверку на четность (нечетность) и кратность элементов массива:

Odd(Arr [i]) – элемент массива является нечетным;

Not Odd(Arr [i]) – элемент массива является четным;

Arr [i] Mod 3 = 0 – элемент массива кратен 3;

Arr [i] Mod 5 <> 0 – элемент массива не кратен 5;

Arr [i] Mod 10 = 7 – последняя цифра элемента массива равна 7.

Далее рассмотрим алгоритм вычисления произведения, например, неотрицательных и четных элементов массива.

Для вычисления произведения элементов массива в соответствии с условием задачи следует использовать счетчик Count, так как не всегда в массиве будут находиться неотрицательные и четные значения. Если таких элементов в массиве не будет (Count=0), тогда произведение будет = 0.

Как и в предыдущей задаче поиска суммы и количества, пункты алгоритма 1–3 (ввод размера и элементов массива и его вывод) будут совпадать. Поэтому при рассмотрении алгоритма эти действия опустим. Начнем с инициализации переменных:

1. Начальное значение произведения Pr :=1.

2. Начальное значение счетчика неотрицательных и четных элементов массива Count:=0.

3. Просмотрим элементы массива, начиная с первого (индекс 1) до последнего (индекс Size):

3.1. Если элемент массива неотрицательный и четный, то:

3.1.1. Увеличим счетчик Count на 1.

3.1.2. Увеличим произведение Pr в Arr[i] раз.

4. Если после просмотра элементов массива (пункт 3) значение счетчика Count=0, то неотрицательных и четных элементов в массиве нет, следовательно, изменим значение произведения Pr на 0.

5. Выведем значение произведения Pr.

Перед написанием программы введем обозначения используемых в ней переменных:

N_Max – максимальный размер массива, опишем его как константу;

Arr – одномерный массив элементов;

Size – размер массива (количество его элементов);

i – индекс элемента массива;

Count – количество неотрицательных и четных элементов массива;

Pr – произведение неотрицательных и четных элементов массива.

Рассмотрим особенности реализации алгоритма на языке программирования.

Так как по условию задачи элементы массива проверяются на четность, то они должны быть целыми числами (тип Integer), как и их произведение.

Приведем фрагмент кода программы в соответствии с описанным алгоритмом и особенностями языка программирования.

```
Count := 0;
Pr :=1;
   For i :=1 To Size Do // вычисление произведения заданных элементов
      If (Arr[i] >= 0) And (Not Odd(Arr[i])) Then // если элемент массива
         Begin // неотрицательный и четный, то увеличивается:
            Inc(Count); // счетчик неотрицательных и четных элементов на 1
               Pr :=Pr*Arr[i]; // произведение Pr в Arr[i] раз
            End;
If (Count = 0) Then Pr :=0; {если неотрицательных и четных элементов в массиве нет, то значение произведения корректируется (становится = 0)}
  WriteLn («Произведение неотрицательных и четных элементов = ’, Pr);
  // вывод результата
```

На примерах рассмотренных задач можно обобщить правила составления программ:

1. Правильно рассмотреть условие и в соответствии с ним решить задачу.

2. Записать алгоритм решения задачи.

3. Ввести обозначения переменных.

4. Рассмотреть особенности реализации алгоритма на выбранном языке программирования.

5. Составить программу.

6. Протестировать ее на разных наборах данных.

Для более наглядного представления материал данной темы лучше представить в виде презентации с применением эффектов анимации.

Список литературы

1. Коренская И.Н. Основы алгоритмизации и программирования / И.Н. Коренская, Е.В. Николаенко, А.А. Осмоловский, В.А. Полубок. – Минск: БГУИР, 2012. – 105 с.

2. Коренская И.Н. Основы программирования в среде Delphi. Лабораторный практикум / И.Н. Коренская, И.В. Лущицкая. – Минск: БГУИР, 2013. – 129 с.