

Министерство образования Нижегородской области  
Государственное бюджетное профессиональное образовательное учреждение  
«Нижегородский радиотехнический колледж»

ОУД.07 Информатика

## ОДНОМЕРНЫЕ МАССИВЫ ВЕЛИЧИН

Учебно-методическая разработка учебного занятия  
для преподавателей

Составитель: Пономарева М.В.,  
преподаватель ГБПОУ «НРТК»

Нижний Новгород  
2017г.

## Методическое обоснование

Данная тема относится к общеобразовательной дисциплине «Информатика» и направлена на изучение понятия «массив». Само понятие массива является малоизученным для обучающихся, соответствующего программная реализация при решении математических задач представляется для обучающихся затруднительной. При изучении новой темы "Одномерные массивы" важно создать проблемную ситуацию на уроке с целью активации деятельности обучающихся, повышенной мотивации при изучении данной темы. Познавательный процесс обучающихся должен начинаться с самого начала занятия и продолжаться на протяжении всего занятия программированием.

Цели и задачи данной учебно-методической разработки:

Образовательная: закрепить умения выполнять алгоритм с циклами, создать условия для формирования представления массиве, одномерном массиве, индексе массива и элементе массива.

Развивающая: развивать логическое, абстрактное и алгоритмическое мышление через установление причинно-следственных связей; развивать способность воспринимать, обрабатывать и обобщать информацию; развивать навыки самоконтроля и взаимоконтроля.

Воспитательная: развивать любознательность и познавательный интерес; воспитывать сознательное отношение и творческий подход к изучаемому предмету, четкость и организованность в труде, аккуратность, внимательность, бережное отношение к технике и к информации.

Мотивационная: побудить интерес к изучению предмета.

В работе представлен теоретический материал и примеры индивидуальных заданий для закрепления навыков работы с одномерными массивами.

Данная разработка была апробирована в группах 1КСК-16-1, 1РА-16-1уп, 1РА-16-2 и 1РЭТ-16-1. Апробация показала, что обучающиеся получили представление об одномерном массиве и научились работать с ним при решении математических задач.

## СОДЕРЖАНИЕ: КОНСПЕКТ УЧЕБНОГО ЗАНЯТИЯ

Тема: Одномерные массивы величин

Вид занятия: комбинированное занятие

Цель: Изучение понятия «массив», правил оформления алгоритмов и программ с массивами; формирование навыков решения задач с применением одномерных массивов; закреплениенавыков по составлениюциклических программ и алгоритмов, самостоятельной работы и работы в группе.

Задачи:

обучающие: формированиепонятия массива, знаний характеристик массивов; формированиепрактических навыков разработки алгоритмов и программ с одномерными массивами; закрепить умения выполнять алгоритмы и программы с циклами.

воспитывающие: воспитать сознательное отношение и творческий подход к изучаемому предмету, четкость и организованность в труде, аккуратность, внимательность, бережное отношение к технике и к информации.

развивающие:развить любознательности познавательныйинтерес; развить логическое, абстрактное и алгоритмическое мышление через установление причинно – следственных связей; развить способность воспринимать, обрабатыватьи обобщать информацию; развить навыки самоконтроля и взаимоконтроля; формирование умения применять на практике полученные знания.

Методы работы: словесные, проблемно-поисковые, наглядно-иллюстрационные, репродуктивные, методы самостоятельной работы.

Оборудование и наглядные пособия: проектор, компьютер преподавателя, компьютеры обучающихся.

Междисциплинарные связи: математика, логика.

Внутридисциплинарные связи: основы алгоритмизации, линейный алгоритм, разветвляющийся алгоритм, циклический алгоритм.

### Хронокарта учебного занятия

№	Этапы и содержание занятия	Время
1.	Перекличка. Формулирование темы и цели занятия.	10 мин
2.	Актуализация знаний	10 мин
3.	Теоретический материал занятия	55 мин
4.	Рефлексия, задание на дом	15 мин

## Ход урока

### I. Актуализация знаний

- ☒ Повторить понятие цикла, циклического алгоритма (Цикл – это последовательность операторов, которая может выполняться более одного раза. Циклический алгоритм – это алгоритм, содержащий один или несколько циклов).
- ☒ Какие виды циклов мы изучили (с постусловием, с предусловием, с параметром).
- ☒ Какой цикл обязательно выполниться хотя бы один раз (с постусловием).
- ☒ В каком цикле начальное значение для параметра цикла не задается перед циклом (цикл с параметром).
- ☒ Какой цикл может ни разу не выполниться (с предусловием).

### II. Теоретический материал занятия

Предположим, что программа работает с большим количеством однотипных данных. Скажем около ста разных целых чисел нужно обработать, выполнив над ними те или иные вычисления. Как вы себе представляете 100 переменных в программе? И для каждой переменной нужно написать одно и то же выражение вычисления значения?

ПРИМЕР 1: Найти сумму пяти целых чисел.

РЕШЕНИЕ: Для решения этой задачи можно описать пять целых переменных для данных чисел и еще одну - для их суммы. Обозначим исходные числа  $a_1, a_2, a_3, a_4, a_5$ , а их сумму -  $s$ . Тогда можно составить такую программу:

```
program summa1;  
  var a1, a2, a3, a4, a5: integer;  
  begin  
    writeln('введите пять целых чисел');  
    readln(a1, a2, a3, a4, a5);  
    s:=a1+a2+a3+a4+a5;  
    writeln('s=',s);  
    readln;  
  end.
```

ПРИМЕР 2: Найти сумму тридцати целых чисел.

РЕШЕНИЕ: если эту задачу решать по аналогии с предыдущей, то необходимо будет описать 30 переменных для всех исходных чисел.

Это очень неэффективно. Есть более простое решение. Это использование такой структуры (типа) данных как массив. В математике, экономике, информатике часто используются упорядоченные наборы данных, например, последовательности чисел, таблицы, списки фамилий. Для обработки наборов данных одного типа вводится понятие массива.

Массив представляет собой последовательность ячеек памяти, в которых хранятся однотипные данные. При этом существует всего одно имя переменной связанной с массивом, а обращение к конкретной ячейке происходит по ее

индексу (номеру) в массиве.

Массив – это упорядоченная последовательность однотипных данных, состоящая из фиксированного числа элементов, имеющих один и тот же тип, и обозначаемая одним именем.

Массивы бывают одномерные, двумерные, трехмерные. В математике одномерные массивы называют вектором, двумерные – матрицей (таблицей).

Каждая ячейка содержит элемент массива. Элементы нумеруются по порядку, но необязательно начиная с единицы (хотя в языке программирования Pascal чаще всего именно с нее). Порядковый номер элемента массива называется индексом этого элемента.

Нужно четко понимать, что индекс ячейки массива не является ее содержимым. Содержимым являются хранимые в ячейках данные, а индексы только указывают на них. Одномерный массив можно представить в виде пронумерованных и выставленных в ряд одинаковых коробок с апельсинами. Индексом будет являться номер коробки, а элементом – содержимое коробки, т.е. количество апельсинов в коробке. Если такой ряд коробок назвать, например, «Апельсины», то это название и будет являться именем массива.

Действия над массивом осуществляются путем использования имени переменной, связанной с областью данных, отведенной под массив. Формат вызова элемента массива:

имя\_массива[индекс].

То есть, если рассматривать пример ряда коробок «Апельсины», то для получения информации о том, сколько апельсинов в коробке с номером 6, мне нужно указать имя ряда и номер коробки:

Апельсины[6].

Все элементы определенного массива имеют один и тот же тип. У разных массивов типы данных могут различаться. Например, один массив может состоять из чисел типа integer, а другой – из чисел типа real.

Индексы элементов массива обычно целые числа, однако могут быть и символами, а также описываться другими порядковыми типами. Т.е. для индекса можно использовать тип, в котором определена дискретная последовательность значений, и все эти значения можно пересчитать по порядку. Индексировать можно как константами и переменными, так и выражениями, результат вычисления которых дает значение перечислимого типа.

Примеры одномерных массивов:

- ☒  $X_1, X_2, \dots, X_n$  – одномерный массив, состоящий из  $n$  элементов;
- ☒  $A_0, A_1, A_2, \dots, A_{10}$  – одномерный массив, состоящий из 11 элементов;
- ☒ (1.6, 14.9, -5.0, 8.5, 0.46) – одномерный массив, состоящий из 5 действительных чисел;
- ☒ (Панин, Горбунов, Радьков, Бочкарев, Ларьков) – одномерный массив, состоящий из 5 текстовых величин (фамилий);
- ☒ (1, 2, 3, 4, 5) – одномерный массив, состоящий из 5 натуральных чисел.

Общий вид описания одномерного массива в Pascal:

Var <имя массива>: array [<тип индекса>] of <тип компонентов>;

Например:

Var A: array [1..5] of integer; - описывает одномерный массив состоящей из 5 элементов целого типа.

Пример описания массивов:

const n = 200;

type

months = (jan, feb, mar, apr, may, jun, jul, aug, sep, oct, nov, dec);

years = 1900..2100;

people = array[years] of longint;

var

growth: array[months] of real;

hum: people;

notes: array[1..n] of string;

Массивы описываются в блоке описания переменных. Особенностью описания массивов на языке Pascal является то, что количество элементов должно быть известно заранее. Если вы не знаете, сколько элементов массива вам понадобится, опишите в блоке констант переменную с примерным количеством элементов и укажите ее при описании массива.

Алгоритм решения задач с использованием массивов:

1. Описание массива.

2. Заполнение массива.

3. Вывод (распечатка) массива.

4. Выполнение условий задачи.

5. Вывод результата.

Таблица 1. Способы заполнения массивов

1 способ (задание элементов вручную)	2 способ (ввод элементов с экрана)
<pre>program pr1; var a: array [1..5] of integer;     i: integer; begin     writeln ('Первый способ');     a[1]:=6;     a[2]:=9;     a[3]:=5;     a[4]:=8;     a[5]:=6;     for i:=1 to 5 do writeln (a [i]);     readln; end.</pre>	<pre>Program pr2; Var A: array [1..5] of real;     i: integer; begin     writeln ('Второй способ');     For i:=1 to 5 do Readln (a [i]);     Readln; end.</pre>

## Примеры использования массивов

Найти сумму тридцати целых чисел.

Program pr1;

var a: array [1..30] of integer;

sum, i: integer;

begin

writeln (' задать 5 целых значений элементов массива a');

for i: =1 to 30 do read (a [i]); readln;

sum: =0;

for i: =1 to 30 do sum: =sum+a[i];

writeln('сумма элементов массива=', sum);

end.

Поиск минимального (максимального) элемента массива.

Алгоритм поиска минимального (максимального) элемента в неупорядоченном массиве довольно очевиден: делается предположение, что первый элемент массива последовательно сравнивается с этим элементом. Если во время очередной проверки обнаруживается, что проверяемый элемент меньше (больше) принятого за минимальный (максимальный), то этот элемент принимается за минимальный (максимальный) и продолжается проверка оставшихся элементов.

Ниже представлена программа поиска минимального элемента в массиве целых чисел.

program min1;

var a: array [1..10] of integer; {массив целых}

min: integer; {номер минимального элемента массива}

i: integer; {номер элемента, сравниваемого с минимальным}

begin

writeln(' задать 10 целых значений элементов массива a');

for i: =1 to 10 do read (a [i]); readln;

min: =1; {пусть первый элемент минимальный }

for i: =2 to 10 do

if a [i]<a[min] then min:=i ;

writeln('минимальный элемент массива:', a [min]);

writeln ('номер элемента: ', min);

end.

Задача 1. Дан целочисленный одномерный массив, состоящий из  $n$  элементов. Найти сумму и произведение нечетных элементов, кратных 3.

Введем обозначения:  $n$  – количество элементов в массиве;  $A$  – имя массива;  $i$  – индекс элемента массива;  $A_i$  –  $i$ -й элемент массива  $A$ ;  $s$  – сумма нечетных элементов массива, кратных 3;  $p$  – произведение нечетных элементов массива, кратных 3.

Входные данные:  $n$ ,  $A$ .

Выходные данные:  $s$ ,  $p$ .

Первоначально сумма искомых элементов равна нулю: не

просуммировано ни одно слагаемое, то есть  $s:=0$ . Далее, используя любой оператор цикла, просматриваем весь массив от первого и до последнего элемента. И если при этом элемент массива нечетный и кратен 3, то к уже накопленной сумме добавляется очередное слагаемое, т.е.  $s:=s + A[i]$ . Здесь слева и справа от знака присваивания записано имя одной и той же переменной  $s$ , именно это обеспечивает постепенное накопление суммы:  $s$  справа – уже вычисленное известное значение суммы,  $s$  – ее новое, вычисляемое значение.

При просмотре массива можно сразу вычислить и произведение элементов массива, удовлетворяющих заданному условию. Произведение вычисляется с помощью оператора  $p:=p*A[i]$ . При этом  $p$  справа и  $p$  слева имеют разные значения:  $p$  справа – уже известное, вычисленное ранее значение произведения,  $p$  слева – новое, вычисляемое его значение. Первоначально искомое произведение равно единице, т.е.  $p:=1$ .

При решении этой задачи можно использовать любой из видов циклов. Рассмотрим несколько вариантов решения задачи.

Для решения используется цикл с параметром:

```
Program Primer1_1;  
Var A: Array[1..20] Of Integer;  
    i, n, s, p: Integer;  
Begin  
    Write ('n='); Readln (n);  
    For i:=1 To n Do Readln (A[i]); {ввод массива}  
    s:= 0; p:=1;  
    For i:=1 To n Do {обработка массива}  
        If (A[i] mod 2 <>0) and (A[i] mod 3 = 0) Then  
            Begin  
                s:=s+A[i];  
                p:= p*A[i]  
            End;  
        Writeln ('s=', s, 'p=', p);  
    Readln  
End.
```

Задача 2. Дан массив целых чисел. Найти количество тех элементов, значения которых положительны и не превосходят заданного натурального числа  $A$ .

Введем обозначения:  $n$  – количество элементов в массиве;  $X$  – имя массива;  $i$  – индекс элемента массива;  $X_i$  –  $i$ -й элемент массива  $X$ ;  $A$  – заданное число;  $k$  – количество элементов, значения которых положительны и не превосходят заданного числа  $A$ .

Входные данные:  $n$ ,  $X$ ,  $A$ .

Выходные данные:  $k$ .

Вводим с клавиатуры значение числа  $A$ . Количество элементов, значения которых положительны и не превосходят заданного числа  $A$ , вначале полагаем равным нулю, то есть  $k:=0$ . Если очередной элемент массива положителен и не



превосходят заданного числа  $A$ , то количество таких элементов нужно увеличить на единицу, то есть  $k := k + 1$ . Таким образом, обрабатываются все элементы массива.

При решении этой задачи можно использовать любой из видов циклов. Рассмотрим несколько вариантов решения задачи.

Для решения используется цикл с параметром:

```
Program Primer2_1;
```

```
Var X: Array[1..20] Of Integer;
```

```
    i, n, k, A: Integer;
```

```
Begin
```

```
    Write ('n='); Readln (n);
```

```
    For i:=1 To n Do Readln (X[i]); {ввод массива}
```

```
    Write ('A='); Readln (A); k:= 0;
```

```
    For i:=1 To n Do {обработка массива}
```

```
        If (X[i] > 0) and (X[i] <= A) Then k:=k + 1;
```

```
    Writeln ('k=', k);
```

```
    Readln
```

```
End.
```

### 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  $X_1 \leq X_2 \leq \dots \leq X_n$ , и в порядке убывания, если  $X_1 \geq X_2 \geq \dots \geq X_n$ .

Существует большое количество алгоритмов сортировки, но все они базируются на трех основных:

- ☒ сортировка обменом;
- ☒ сортировка выбором;
- ☒ сортировка вставкой.

Представим, что нам необходимо разложить по порядку карты в колоде. Для сортировки карт обменом можно разложить карты на столе лицевой стороной вверх и менять местами те карты, которые расположены в неправильном порядке, делая это до тех пор, пока колода карт не станет упорядоченной.

Для сортировки выбором из разложенных на столе карт выбирают самую младшую (старшую) карту и держат ее в руках. Затем из оставшихся карт вновь выбрать наименьшую (наибольшую) по значению карту и помещают ее позади той карты, которая была выбрана первой. Этот процесс повторяется до тех пор, пока вся колода не окажется в руках. Поскольку каждый раз выбирается наименьшая (наибольшая) по значению карта из оставшихся на столе карт, по завершению такого процесса карты будут отсортированы по возрастанию (убыванию).

Для сортировки вставкой из колоды берут две карты и располагают их в необходимом порядке по отношению друг к другу. Каждая следующая карта, взятая из колоды, должна быть установлена на соответствующее место по отношению к уже упорядоченным картам.

Итак, решим следующую задачу. Задан массив  $Y$  из  $n$  целых чисел. Расположить элементы массива в порядке возрастания их значений.

### Сортировка методом «пузырька»

Сортировка пузырьковым методом является наиболее известной. Ее популярность объясняется запоминающимся названием, которое происходит из-за подобия процессу движения пузырьков в резервуаре с водой, когда каждый пузырек находит свой собственный уровень, и простотой алгоритма. Сортировка методом «пузырька» использует метод обменной сортировки и основана на выполнении в цикле операций сравнения и при необходимости обмена соседних элементов. Рассмотрим алгоритм пузырьковой сортировки более подробно. Сравним первый элемент массива со вторым, если первый окажется больше второго, то поменяем их местами. Те же действия выполним для второго и третьего, третьего и четвертого,  $i$ -го и  $(i+1)$ -го,  $(n-1)$ -го и  $n$ -го элементов. В результате этих действий самый большой элемент станет на последнее ( $n$ -е) место. Теперь повторим данный алгоритм сначала, но последний ( $n$ -й) элемент, рассматривать не будем, так как он уже занял свое место. После проведения данной операции самый большой элемент оставшегося массива станет на  $(n-1)$ -е место. Так повторяем до тех пор, пока не упорядочим весь массив.

В табл.1 подробно расписан процесс упорядочивания элементов в массиве. Нетрудно заметить, что для преобразования массива, состоящего из  $n$  элементов, необходимо просмотреть его  $n-1$  раз, каждый раз уменьшая диапазон просмотра на один элемент. Блок-схема описанного алгоритма приведена на рис. 1. Обратите внимание на то, что для перестановки элементов (блок 4) используется буферная переменная  $b$ , в которой временно хранится значение элемента, подлежащего замене.

Таблица 2. Процесс упорядочивания элементов в массиве по возрастанию

Номер элемента	1	2	3	4	5
Исходный массив	7	3	5	4	2
Первый просмотр	3	5	4	2	7
Второй просмотр	3	4	2	5	7
Третий просмотр	3	2	4	5	7
Четвертый просмотр	2	3	4	5	7

Вопрос: Что необходимо сделать для перестановки элементов в массиве по убыванию их значений? - Необходимо при сравнении элементов массива заменить знак  $>$  на  $<$ .

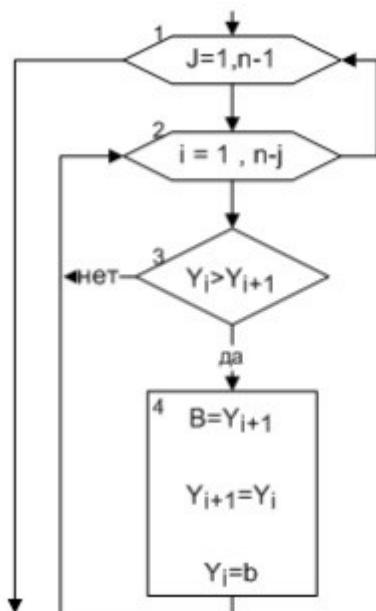


Рис. 1. Сортировка массива пузырьковым методом

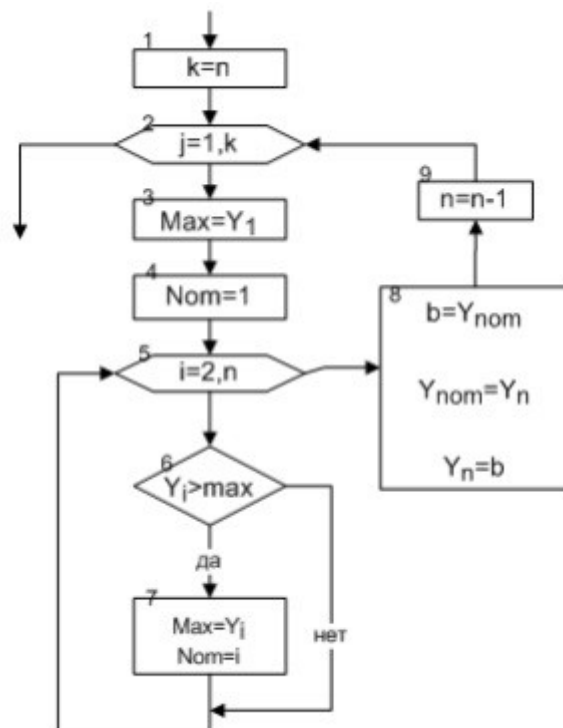


Рис. 2. Сортировка массива выбором наибольшего элемента

### Сортировка выбором

Алгоритм сортировки выбором приведен в виде блок-схемы на рис. 2. Найдем в массиве самый большой элемент (блоки 3–7) и поменяем его местами с последним элементом (блок 8). Повторим алгоритм поиска максимального элемента, уменьшив количество просматриваемых элементов на единицу (блок 9), и поменяем его местами с предпоследним элементом (блоки 3–7). Описанную выше операцию поиска проводим до полного упорядочивания элементов в массиве. Так как в блоке 9 происходит изменение переменной  $n$ , то в начале алгоритма ее значение необходимо сохранить (блок 1).

Вопрос: Для упорядочивания массива по убыванию необходимо перемещать ... элемент? - Минимальный элемент.

Задания для самостоятельной работы.

Дан массив целых чисел, вывести максимальный и минимальный элементы.

Дан массив вещественных чисел, упорядочить массив по возрастанию элементов.

Дан массив вещественных чисел, упорядочить массив по убыванию

элементов.

### III. Рефлексия

1. Дайте ответ на вопрос: что такое массив, приведите пример.
2. Какой массив называется одномерным?
3. Какой способ ввода массива вам больше нравится? Почему?
4. Какой способ упорядочивания данных в массиве вам больше нравится? Почему?

### IV. Задание на дом

Составить программу для решения задачи: найти сумму положительных элементов массива.

## Литература

1. Голицына О. Л., Попов И. И. Основы алгоритмизации и программирования: Учебн. пособие. - М.: ФОРУМ: ИНФРА-М, 2005. - 432с.
2. Колмыкова Е. А., Кумская И. А. Информатика: Учеб. пособие для студ. сред. проф. Образования — М.: Издательский центр «Академия», 2005. - 416с.
3. Михеева Е. В., Титова О. И. Информатика: учебник для студ. учреждений сред. проф. Образования— 8-е изд., стер. - М.: Издательский центр «Академия», 2012. - 352с.
4. Немчанинова Ю.П. Алгоритмизация и основы программирования на базе Kturtle. (ПО для обучения программированию Kturtle): Учебное пособие. – Москва: 2008. - 50 с.
5. Семакин И. Г., Шестаков А.П. Основы алгоритмизации и программирования: учебник для студ. учреждений сред. проф. образования — М.: Издательский центр «Академия», 2013. - 304с.
6. Власова О.А. Урок по теме «Одномерные массивы»  
<http://festival.1september.ru/articles/608710/>
7. Программирование(9 кл.). Обработка табличных данных. Массив  
<https://sites.google.com/site/415ict/textbooks/prog-9/05-array>