



# Пример 1: Программа "Arrays Demo"

---

## Цель примера

Продемонстрировать процесс создания простейшей программы, которая заполняет и выводит содержимое массива на экран.

## Введение

Массив - это специальная структура данных, которая позволяет нам хранить однотипные данные в логической группе и обращаться к элементам этой группы по номеру, или индексу элемента. Индексирование элементов массива начинается с 0.

На практике часто встречаются ситуации, когда нужно обработать список элементов таким образом, чтобы при изменении размера списка программу не нужно было дорабатывать.

Самый близкий пример из мира Java - это строка. Класс `String` является оберткой над массивом символов `char[]`, содержание которого определяется при создании экземпляра (объекта) и его нельзя изменить после создания. Используя доступные методы в этом классе мы можем выполнять целый ряд операций: выбрать подстроку в диапазоне, узнать какой символ находится под индексом, заменить часть символов на другие и т.д. Все методы работают одинаково хорошо с любой строкой произвольной длины, независимо от содержания.

## Практическое руководство

Рассмотрим программу, которая заполняет массив случайными числами и выводит его содержимое на экран.

---

### Шаг 1.

Логика работы будет реализована в методе `main()` класса `ArrayDemo`:

```
import java.util.Random;

public class ArrayDemo {

    public static void main(String[] args) {

    }

}
```

---

## Шаг 2.

Для работы нам необходимо создать массив целых чисел `int[]` произвольного размера, например 10:

```
import java.util.Random;

public class Statistics {

    public static void main(String[] args) {

        int[] array = new int[10];

    }

}
```

**⚠ Важно:** Обратите внимание на синтаксис создания массива. Массив также можно создать используя сокращенную нотацию: `int[] array = {1, 2, 3, 4, 5};`, но в таком случае мы должны заранее знать все значения элементов массива, поэтому такой способ в данном случае не подходит.

---

## Шаг 3.

Узнать размер массива можно при помощи свойства `length`, например: `array.length`. За счет этого свойства мы можем удобно итерировать по массиву используя циклы. Для заполнения массива случайными числами нам понадобится `Random`.

```
import java.util.Random;

public class Statistics {

    public static void main(String[] args) {

        int[] array = new int[10];

        Random random = new Random();

    }

}
```

---

## Шаг 4.

Заполним массив случайными числами при помощи цикла `for`:

```
import java.util.Random;

public class Statistics {

    public static void main(String[] args) {


        int[] array = new int[10];

        Random random = new Random();

        for (int i = 0; i < array.length; i++) {
            array[i] = random.nextInt();
        }

    }

}
```

 **Важно:** Помимо цикла `for` можно использовать и другие, например `while` или `do while` (с учетом его специфики). Однако, для изменения содержимого массива нельзя использовать цикл `for each`, поскольку он работает только на чтение.

---

## Шаг 5.

Добавим вывод массива на экран:

```
import java.util.Random;
```

```

public class Statistics {

    public static void main(String[] args) {

        int[] array = new int[10];

        Random random = new Random();

        for (int i = 0; i < array.length; i++) {
            array[i] = random.nextInt();
        }

        for (int i = 0; i < array.length; i++) {
            System.out.println("array[" + i + "] = " + array[i]);
        }

    }

}

```

Результат работы программы (каждый запуск программы будет возвращать случайные значения):

```

array[0] = 41 array[1] = 63 array[2] = 16 array[3] = 17 array[4] = 8 array[5] = 61 array[6] = 80
array[7] = 89 array[8] = 90 array[9] = 6

```

## Рекомендации:

- Запустить программу и сравнить результаты;
- Попробовать заменить вывод содержимого массива при помощи цикла `for each`;
- Создать массив случайной длины и сравнить результаты;