

Java Basic

lecture #10. Arrays in Java

Mentor: Il'yas Miftakhov

lecture #10. Arrays in Java

- Array
 - 1-dimentional arrays, max index of an array
 - operations over array (new array, fill array, print array, get elements, iterate over)
 - Random class. Get random value. Create random array
 - Arrays class methods: copyOf, toString, sort, copyRange
 - Practice

Определение

Массив в Java — это группа переменных одинакового типа, на которые ссылается общее имя.

- В Java все массивы распределяются динамически.
- Массивы хранятся как последовательные ячейки памяти.
- в Java массивы являются объектами, мы можем найти их длину, используя свойство объекта `length`.
- Переменная массива Java может быть объявлена, как и другие переменные, `<тип>[] <имя>`.
- Переменные в массиве упорядочены, и каждая имеет индекс, **начинающийся с 0**.
- Размер массива должен быть указан как целое значение.
- Прямой суперкласс типа массива — `Object`.
- Размер массива не может быть изменен (после инициализации). Однако ссылка на массив может указывать на другой массив.

Создание, инициализация и доступ к массиву

Объявление массива состоит из двух компонентов: типа и имени.

Синтаксис:

```
<type>[] <name> = new <type>[<size>]
```

```
int[] intArray;           //declaring array
intArray = new int[20];    // allocating memory to array
OR
int[] intArray = new int[20];
```

1. Элементы в массиве по умолчанию, будут автоматически инициализированы **0** (для числовых типов), **false** (для логического значения) или **null** (для ссылочных типов).
2. Получение массива представляет собой двухэтапный процесс.
 1. Во-первых, вы должны объявить переменную нужного типа массива.
 2. Во-вторых, вы должны выделить память для хранения массива, используя **new**, и присвоить ее переменной массива. Таким образом, в Java все массивы распределяются **динамически**.






Литерал массива и доступ к элементам массива

```
int[] intArray = { 1,2,3,4,5,6,7,8,9,10 };
```

Длина этого массива определяет длину создаваемого массива.

1. Доступ к каждому элементу массива осуществляется через его индекс.
2. **Индекс начинается с 0 и заканчивается на (общий размер массива)-1.**

```
for (int i = 0; i < arr.length; i++)  
{  
    System.out.println("Element at index " + i + " : "+ arr[i]);  
}
```




Arrays of Objects

Массив объектов создается подобно массиву элементов данных примитивного типа.


```
Elf[] array = new Elf[5];
```

```
Orc[] array = new Orc[5];
```

```
Student[] students = new Student[] {new Student("Aleks"), new Student("Thea")};
```




Что произойдет, если мы попытаемся получить доступ к элементам за пределами размера массива?

- JVM генерирует исключение `ArrayIndexOutOfBoundsException`, чтобы указать, что к массиву был осуществлен доступ с использованием недопустимого индекса.
 - Индекс либо отрицателен, либо больше или равен размеру массива.
- 



clone, copyOf, toString, sort, copyOfRange

- clone() Когда вы клонируете одномерный массив, такой как Object[], выполняется «глубокая копия» с новым массивом, содержащим копии элементов исходного массива, а не ссылки.
 - copyOf() начинает копирование с 0-го индекса исходного массива и копирует указанное количество элементов
 - copyOfRange() может копировать диапазон элементов из исходного массива
 - toString() возвращает строковое представление содержимого указанного массива
 - sort() сортирует указанный массив целых чисел в порядке возрастания.
- 

practice

1. Создайте массив из всех нечётных чисел от 1 до 99, выведите его на экран в строку в обратном порядке
2. Создайте массив из 15 случайных целых чисел из отрезка $[0;9]$. Подсчитайте сколько в массиве чётных элементов и выведите это количество на экран на отдельной строке.
3. Создать массив типа String с размером 7. Записать в него значения дней недели. Вывести на консоль значение последнего элемента.
4. Найти максимальный по значению элемент.