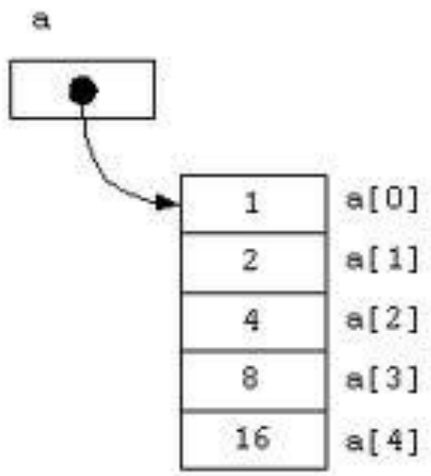
II. Типы, переменные, управляющие инструкции

5. Массивы



	Массив это объект который содержит фиксированное число значений одного типа. Длина массива задаётся в момент создания массива и после создания не меняется. Значения хранящиеся в массиве называются элементами массива. Доступ к элементам массива осуществляется с помощью целочисленного индекса. Первый элемент массива имеет индекс 0. Тип значений элементов массива может быть любым примитивным или ссылочным типом. Сам тип массива независимо от того является ли он массивом значений примитивного типа или ссылочного типа является ссылочным типом.

Объявление переменной типа массив

```
type[] variable;
```

При объявлении переменной типа массив необходимо указать тип значений элементов массива. Переменная типа массив может хранить ссылку на массив значений заданного типа. Объявление перменной типа массив не создаёт массив.

Объявление переменной типа массив с инициализацией

```
type[] variable = new type[size];
```

```
type[] variable = new type[] { val1, val2, ..., valn };
```

```
type[] variable = { val1, val2, ..., valn };
```

Для создания массива можно использовать оператор new. При создании массива значения элементов массива можно явно проинициализировать заданными значениями. Если элементы массива явно не инициализируются их значения задаются равными значениям по умолчанию.

Объявление и инициализация массива

```
package arrays;

public class DeclDemo {

   public static void main(String[] args) {

      int[] anArray;
      anArray = new int[10];

      for (int i = 0; i < 10; i++) {

         anArray[i] = 100 * i;
      }

      for (int i = 0; i < 10; i++) {

         System.out.println(anArray[i]);
      }
   }
}
```

```
0
100
200
300
400
500
600
700
800
900
```

Анонимные массивы

```
someObject.someMethod( new type[] { val1, val2, ..., valn })
```

Анонимные массивы можно создавать без присваивания ссылки переменной. Ссылку на созданный анонимный массив можно передавать как параметр в методы.

Анонимный массив

DIASOFT
всё по-настоящему

```
public class AnonymousDemo {

    public static void main(String[] args) {

        System.out.println("first sum: "
                + sum(new int[] { 1, 2, 3, 4, 5 }));
        System.out.println("second sum: "
                + sum(new int[] { 1, 2, 3, 4, 5, 6 }));
        System.out.println("third sum: "
                + sum(new int[] { 1, 2, 3, 4, 5, 6, 7}));

    }

    public static int sum(int[] numbers) {
        int total = 0;
        for (int i : numbers) {
            total = total + i;
        }
        return total;
    }
}
```

```
first sum: 15
second sum: 21
third sum: 28
```

Массив это ссылочный тип

Передача ссылки на массив функции

```
public class RefDemo {

   public static void main(String[] args) {

      String[] b = { "Apple", "Mango", "Orange" };
      System.out.println("Before Function Call: " + b[0]);
      changeElement(b);
      System.out.println("After Function Call: " + b[0]);

   }

   public static void changeElement(String[] a) {
      a[0] = "Changed";
   }
}
```

```
Before Function Call: Apple
After Function Call: Changed
```

Длина массива и обход массива

Попытка обратиться к элементу массива с индексом за пределами диапазона индексов элементов массива приведёт к исключению. Минимальный индекс элемента в массиве 0. Максимальный индекс элемента в массиве на единицу меньше длины массива. Длину массива можно получить с помощью свойства length.

Длина массива

DIASOFT
всё по-настоящему

```
public class LengthDemo {

   public static void main(String[] args) {

      int[] anArray = { 0, 100, 200, 300, 400, 500, 600, 700, 800, 900 };

      for (int i = 0; i < anArray.length; i++) {

         System.out.println(anArray[i]);
      }

      System.out.println();
      System.out.println("Array length is: " + anArray.length);
   }
}
```

```
0
100
200
300
400
500
600
700
800
900

Array length is: 10
```

Выход за пределы массива

```
public class OutOfBoundsDemo {

   public static void main(String[] args) {

      int[] anArray = { 0, 100, 200, 300, 400, 500, 600, 700, 800, 900 };


      for (int i = 0; i <= 10; i++) {

         System.out.println(anArray[i]);
      }

      System.out.println();
      System.out.println("Array length is: " + anArray.length);
   }
}
```

```
0
100
200
300
400
500
600
700
800
900
Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: 10
at arrays.OutOfBoundsDemo.main(OutOfBoundsDemo.java:12)
```

Цикл “for each” или “enhanced for”

DIASOFT
всё по-настоящему

```
public class EnhancedDemo {

   public static void main(String[] args) {

      int[] anArray = {0, 100, 200, 300, 400, 500, 600, 700, 800, 900 };

      for (int a : anArray) {

         System.out.println(a);
      }
   }
}
```

```
0
100
200
300
400
500
600
700
800
900
```

Копирование массивов

Копирование массивов

```
package java.util;

public class Arrays {
    ...

     public static int[] copyOf(int[] original, int newLength) {
        int[] copy = new int[newLength];
        System.arraycopy(original, 0, copy, 0,
                         Math.min(original.length, newLength));
        return copy;
    }

    public static int[] copyOfRange(int[] original, int from, int to) {
        int newLength = to - from;
        if (newLength < 0)
            throw new IllegalArgumentException(from + " > " + to);
        int[] copy = new int[newLength];
        System.arraycopy(original, from, copy, 0,
                         Math.min(original.length - from, newLength));
        return copy;
    }
    ...
}
```

```
package java.lang;

public final class System {

   public static native void arraycopy(Object src, int srcPos, Object dest, int destPos, int length)
   ...
}
```

C	В классе Arrays из пакета java.util есть методы copyOf и copyOfRange которые создают новый массив содержащий часть элементов другого массива и возвращают ссылку на этот новый массив. Внутри эти методы для копирования массивов используют метод arraycopy из класса System.

Копирование массивов

DIASOFT
всё по-настоящему

```
public class CopyDemo {

   public static void main(String[] args) {

      char[] copyFrom = { 'd', 'e', 'c', 'a', 'f', 'f', 'e', 'i', 'n', 'a', 't', 'e', 'd' };
      char[] copyTo = new char[7];

      System.out.println("Initial array: ");
      System.out.println(copyFrom);

      System.out.println("Copying Array using arraycopy...");
      System.arraycopy(copyFrom, 2, copyTo, 0, 7);
      System.out.println(copyTo);

      System.out.println("Copying Array using copyOfRange...");
      copyTo = Arrays.copyOfRange(copyFrom, 2, 9);
      System.out.println(copyTo);
   }
}
```

```
Initial array:
decaffeinated
Copying Array using arraycopy...
caffein
Copying Array using copyOfRange...
caffein
```

Сортировка массивов

Сортировка массивов

```
package java.util;

public class Arrays {
    ...

    public static void sort(int[] a)
    public static void sort(int[] a, int fromIndex, int toIndex)

     public static void sort(Object[] a) {
        Object[] aux = (Object[])a.clone();
        mergeSort(aux, a, 0, a.length, 0);
    }

    public static void sort(Object[] a, int fromIndex, int toIndex) {
        rangeCheck(a.length, fromIndex, toIndex);
        Object[] aux = copyOfRange(a, fromIndex, toIndex);
        mergeSort(aux, a, fromIndex, toIndex, -fromIndex);
    }

    ...
}
```

В классе Arrays из пакета java.util есть методы sort для сортировки массива или диапазона элементов массива. Для сортировки массива ссылочного типа необходимо чтобы тип реализовывал интерфейс Comparable.

Сортировка массивов

DIASOFT
всё по-настоящему

```
public class SortDemo {

   public static void main(String[] args) {

      int[] anArray = { 900, 200, 1000, 700, 500, 600, 400, 800, 300, 100 };

      for (int i = 0; i < 10; i++) {

         System.out.print(anArray[i] + " ");
      }
      System.out.println();

      System.out.println("Sorting Array...");
      Arrays.sort(anArray);

      for (int a : anArray) {

         System.out.print(a + " ");
      }
      System.out.println();
   }
}
```

```
900 200 1000 700 500 600 400 800 300 100
Sorting Array...
100 200 300 400 500 600 700 800 900 1000
```

Строковое представление

Как распечатывать массивы

Массивы наследуют метод toString из класса Object, при этом тип массива печатается как ‘[‘, плюс сокращённое название типа элемента, плюс хеш код. Для получение строкового представления содержащего информацию об элементах массива необходимо использовать статический метод toString из класса Arrays.

Строковое представления массивов

```
package java.util;

public class Arrays {
    ...

    public static String toString(int[] a)

    public static String toString(Object[] a) {
        if (a == null)
            return "null";
        int iMax = a.length - 1;
        if (iMax == -1)
            return "[]";

        StringBuilder b = new StringBuilder();
        b.append('[');
        for (int i = 0; ; i++) {
            b.append(String.valueOf(a[i]));
            if (i == iMax)
                return b.append(']').toString();
            b.append(", ");
        }
    }
 ...
}
```

В классе Arrays из пакета java.util есть методы toString для получения строкового представления массивов примитивных типов и ссылочных типов.

Строковое представление

```
public class PrintDemo {

   public static void main(String[] args) {

      int[] anArray = { 100, 200, 300, 400, 500, 600, 700, 800, 900, 1000 };

      System.out.println(anArray.getClass().getName() + "@" + Integer.toHexString(anArray.hashCode()));
      System.out.println(anArray);
      System.out.println("ToString from Arrays: \n" + Arrays.toString(anArray));
   }
}
```

```
[I@190d11
[I@190d11
ToString from Arrays:
[100, 200, 300, 400, 500, 600, 700, 800, 900, 1000]
```

Проверка на равенство

Как проверять массивы на равенство

Массивы наследуют метод equals из класса Object, этот метод просто проверяет ссылки на равенство. Для проверки массивов на равенство необходимо использовать статический метод equals из класса Arrays.

Проверка на равенство

DIASOFT
всё по-настоящему

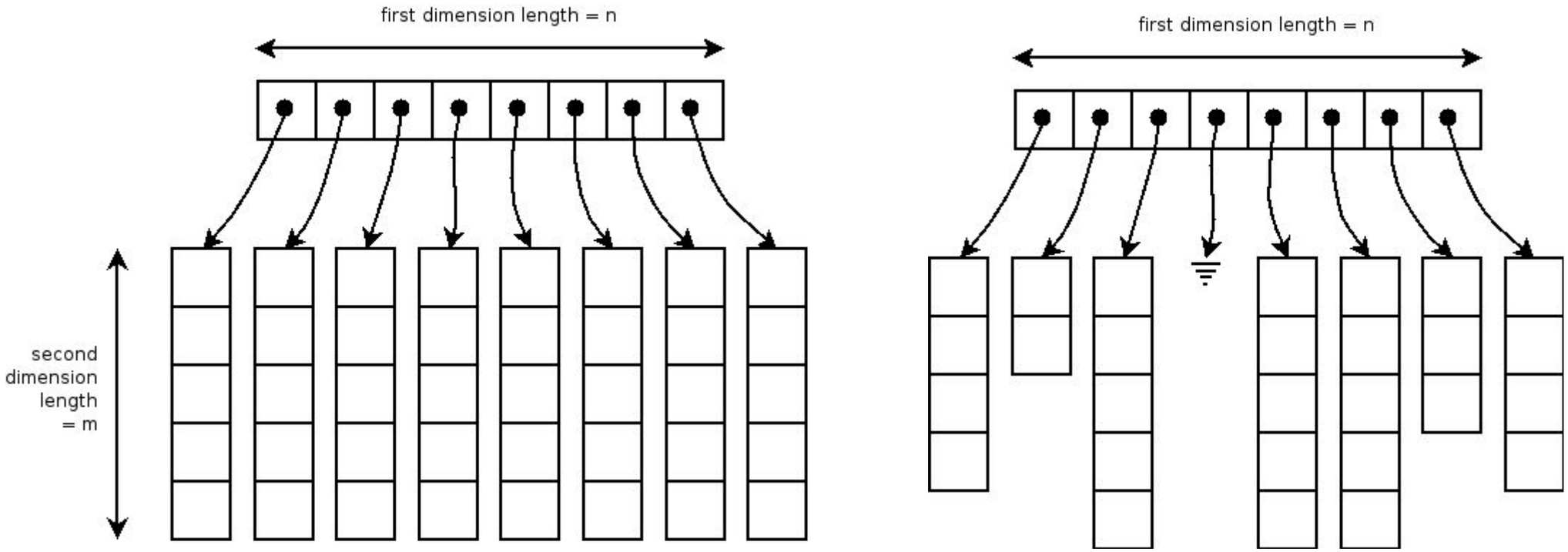
```
package java.util;

public class Arrays {
    ...

    public static boolean equals(int[] a, int[] a2)

    public static boolean equals(Object[] a, Object[] a2) {
        if (a==a2)
            return true;
        if (a==null || a2==null)
            return false;

        int length = a.length;
        if (a2.length != length)
            return false;

        for (int i=0; i<length; i++) {
            Object o1 = a[i];
            Object o2 = a2[i];
            if (!(o1==null ? o2==null : o1.equals(o2)))
                return false;
        }
        return true;
    }
    ...
}
```

Проверка на равенство

```
public class EqualsDemo {

    public static void main(String[] args) {

        int[] one = { 1, 2, 3, 4, 5 };
        int[] two = { 1, 2, 3, 4, 5 };

        System.out.println("Array one: " + Arrays.toString(one));
        System.out.println("Array two: " + Arrays.toString(two));

        System.out.println("Comparing arrays one and two.....");

        System.out.println("Using equals(): " + one.equals(two));
        System.out.println("Using Arrays.equals(): " + Arrays.equals(one,two));
    }
}
```

```
Array one: [1, 2, 3, 4, 5]
Array two: [1, 2, 3, 4, 5]
Comparing arrays one and two.....
Using equals(): false
Using Arrays.equals(): true
```

Многомерные массивы



Многомерные массивы строятся на основе одномерных массивов. Таким образом многомерный массив это массив массивов. У двумерных массивов например строки могут быть разной длины и каждая строка является отдельным объектом и может быть независимо использована.

Объявление переменной типа массив массивов

```
type[][]  variable;
```

При объявлении переменной типа массив массивов необходимо указать тип значений элементов массива. Объявление переменной типа массив массивов не создаёт ни один массив.

Объявление переменной с инициализацией

```
type[][]  variable = new type[size][];
```

```
type[][]  variable = new type[size1][size2];
```

```
type[][]  variable = new type[][]{{val11, val12, ..., val1n},{val21, val22, ..., val2m}};
```

```
type[][]  variable = {{val11, val12, ..., val1n},{val21, val22, ..., val2m}};
```

Для создания массива можно использовать оператор new. При создании массива значения элементов массива можно явно проинициализировать заданными значениями. Если элементы массива явно не инициализируются их значения задаются равными значениям по умолчанию.

Объявление и инициализация

DIASOFT
всё по-настоящему

```
public class Init2dDemo {

   public static void main(String[] args) {

      int[][] anArray;
      anArray = new int[2][3];

      anArray[0][0] = 0;
      anArray[0][1] = 1;
      anArray[0][2] = 2;
      anArray[1][0] = 3;
      anArray[1][1] = 4;
      anArray[1][2] = 5;

      for (int i = 0; i < 2; i++) {
         for (int j = 0; j < 3; j++) {

            System.out.print(anArray[i][j] + " ");
         }
         System.out.println();
      }
   }
}
```

```
0 1 2
3 4 5
```

Объявление с инициализацией

DIASOFT
всё по-настоящему

```
public class Decl2dDemo {

   public static void main(String[] args) {

      int[][] anArray = new int[3][];

      anArray[0] = new int[] { 0, 1, 2 };
      anArray[1] = new int[] { 3, 4, 5, 6, 7 };
      anArray[2] = new int[] { 8, 9, 10 };

      for (int i = 0; i < anArray.length; i++) {
         for (int j = 0; j < anArray[i].length; j++) {

            System.out.print(anArray[i][j] + " ");
         }
         System.out.println();
      }
   }
}
```

```
0 1 2
3 4 5 6 7
8 9 10
```

Строковое представление многомерного массива

Как распечатывать многомерные массивы

Массивы наследуют метод toString из класса Object, при этом тип массива печатается как ‘[‘ плюс сокращённое название типа элемента плюс хеш код. Для получения строкового представления многомерного массива содержащего информацию об элементах массива необходимо использовать статический метод deepToString из класса Arrays.

Строковое представление

```
public class Print2dDemo {

   public static void main(String[] args) {

      int[][] anArray = { { 1 }, { 1, 2 }, { 1, 2, 3 }, { 1, 2, 3, 4 }, { 1, 2, 3, 4 ,5} };

      System.out.println(anArray.getClass().getName() + "@" + Integer.toHexString(anArray.hashCode()));
      System.out.println(anArray);
      System.out.println(Arrays.toString(anArray));
      System.out.println(Arrays.deepToString(anArray));
   }
}
```

```
[[I@190d11
[[I@190d11
[[I@a90653, [I@de6ced, [I@c17164, [I@1fb8ee3, [I@61de33]
[[1], [1, 2], [1, 2, 3], [1, 2, 3, 4], [1, 2, 3, 4, 5]]
```

Проверка на равенство многомерных массивов

Как проверять многомерные массивы на равенство

Массивы наследуют метод equals из класса Object, этот метод просто проверяет ссылки на равенство. Для проверки многомерных массивов на равенство необходимо использовать статический метод deepEquals из класса Arrays.

Проверка на равенство

DIASOFT
всё по-настоящему

```
public class DeepEqualsDemo {

    public static void main(String[] args) {

        int[][] one = { { 1, 2, 3 }, { 4, 5, 6 }, { 7, 8, 9 } };
        int[][] two = { { 1, 2, 3 }, { 4, 5, 6 }, { 7, 8, 9 } };

        System.out.println("Array one: " + Arrays.deepToString(one));
        System.out.println("Array two: " + Arrays.deepToString(two));

        System.out.println("Comparing arrays one and two.....");

        System.out.println("Using equals(): " + one.equals(two));
        System.out.println("Using Arrays.equals(): " + Arrays.equals(one, two));
        System.out.println("Using Arrays.deepEquals(): "
                + Arrays.deepEquals(one, two));
    }
}
```

```
Array one: [[1, 2, 3], [4, 5, 6], [7, 8, 9]]
Array two: [[1, 2, 3], [4, 5, 6], [7, 8, 9]]
Comparing arrays one and two.....
Using equals(): false
Using Arrays.equals(): false
Using Arrays.deepEquals(): true
```

Спасибо

Россия, 127018,
Москва, ул. Полковая 3, стр. 14
Тел.: +7(495) 780 7575, 789 9339
Факс: +7(495) 780 7576, 789 9338
info@diasoft.ru, www.diasoft.ru