Лекция 4. Ссылочные типы. Операторы цикла.

NetCracker®

Массив – это последовательность объектов и примитивов одного типа, обозначаемая одним идентификатором.

Массивы определяются с помощью оператора индексирования [].

Одномерный массив,

```
int a[];
int[] a;
```

Многомерный массив:

```
int a[][];
int[] a[];
int[][] a;
```

```
int[] a, b[]; == int a[], b[][];
```

```
public class ArraysOfPrimitives {
    public static void main(String[] args) {
        int[] a1 = { 1, 2, 3, 4, 5 };
        int[] a2;
        a2 = a1;
        for(int i = 0; i < a2.length; i++)
            a2[i] = a2[i] + 1;
        for(int i = 0; i < a1.length; i++)
            print("a1[" + i + "] = " + a1[i]);
    }
} /* Output:
        a1[0] = 2
        a1[1] = 3
        a1[2] = 4
        a1[3] = 5
        a1[4] = 6
        *///
```



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

int[] predefined = new int[] { 1, 2, 3 };

Object[] objects = { null, "Hello" };

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

char[] string;

string = new char[5];

// Каждый элемент многомерного массива - массив

int[][] nonsquare = new int[3][];

nonsquare[0] = new int[3];

nonsquare[1] = new int[] { 1, 2 };

nonsquare[2] = new int[1];
```


Использование поля *length* при работе с массивом:

```
Point p[] = new Point[5];
for (int i = 0; i < p.length; i++) {
        p[i] = new Point(i, i);
}
```

Значение индекса массива всегда имеет тип **int**.

Допустимые типы при обращении к элементу: **byte, short, char**.

Попытка задействовать **long** приведет к ошибке компиляции.

Максимальное количество элементов в массиве: *2,147,483,647*

Объявление нескольких локальных переменных с одинаковыми именами в пределах видимости блока недопустимо:

```
public class Test {
    public Test() {}
    public static void main(String[] args) {
        Test t = new Test();
        int x;
        {
            int x = 0;
            System.out.println("x = " + x);
        }
        for (int i = 0; i < 2; i++);
        for (int i = 0; i < 2; i++);
    }
}
```

Локальные переменные и параметры методов перекрывают видимость полей класса:

```
public class Test {
    static int x = 5;
    static int args = 0;
    public static void main(String[] args) {
        Test t = new Test();
        int x = 1;
        System.out.println("x = " + x + ", " + args[0]);
    }
}
```

Общая конструкция оператора:

```
if (логическое выражение)
    выражение или блок 1
else
    выражение или блок 2
```

Пример:

```
int x = 5;

if (x < 4)
    System.out.println("Меньше 4");
else if (x > 4) {
    System.out.println("Больше 4");
}
else if (x == 5)
    System.out.println("Равно 5");
else
    System.out.println("Другое значение");
```


Общая конструкция оператора:

```
switch(int value) {
   case const1:
      выражение или блок
   case const2:
      выражение или блок
   case constn:
      выражение или блок
   default:
      выражение или блок
}
```



```
while (логическое условие продолжения)
    повторяющееся выражение или блок

do
   повторяющееся выражение или блок
While (логическое условие продолжения);

for (выполнить до цикла; условие продолжения;
        выполнить после каждого повторения)
    повторяющееся выражение или блок

for (Тип_элемента имя_переменной : массив или коллекция)
    повторяющееся выражение или блок
```



```
public class Test2 {
  public static void main(String[] args) {
    int i = 0;
    outer: for (;;) { // бесконечный цикл
      for (; i < 10; i++) {
        System.out.println("i = " + i);
        if (i == 0) {
          System.out.println("continue");
          continue;
        }
        if (i == 1) {
          System.out.println("break");
          i++; // В противном случае i не будет увеличено.
          break;
        }
        if (i == 2) {
          System.out.println("continue outer");
          i++; // В противном случае i не будет увеличено.
          continue outer;
        }
        if (i == 3) {
          System.out.println("break outer");
          break outer;
```

Результат:

```
i = 0
continue
i = 1
break
i = 2
continue outer
i = 3
break outer
```

```
    } } } } }
```

- Эккель Б. Философия Java. –Эккель Б. Философия Java. – СПб.: Питер, 2009. – 640 с.
- http://www.intuit.ru/
- Шилдт Г. Java. Полное руководство. – СПб.: Вильямс, 2012. – 1104 с.
- Шилдт Г. Полный справочник по Java. Java SE 6 Edition. – СПб.: Вильямс, 2007. – 1040 с.
- Шилдт Г., Холмс Д. Искусство программирования на Java. – СПб.: Вильямс, 2005. – 333 с.
- Шилдт Г. Java. для начинающих. – СПб.: Вильямс, 2009. – 720 с.

Q&A

Thank you!

