

Plan

2023-09-15

1. Homework Review
2. While Loop
3. Do ... While Loop
4. Practice:

-
4. Разбор домашнего задания
 5. Цикл while (while loop)
 6. Цикл do ... while (do while loop)
 7. Практикум:

Theory

Increment and Decrement in Java

Definition

Increment

Increment is an operation that increases the value of a variable by one. In Java, the `++` operator is used for this purpose.

Decrement

Decrement is an operation that decreases the value of a variable by one. In Java, the `--` operator is used for this purpose.

Prefix and Postfix Forms

Prefix Increment (`++i`)

In this form, the value of the variable is first increased by 1, and then it is used in the expression.

Example:

```
public class IncidentCounterexample {  
    public static void main(String[] args) {  
        int counter = 123;           // 123  
        counter++;                   // 123 (+1 after this line, i.e., before the next line)  
        System.out.println(++counter); // 125 (because ++counter, i.e., 124 + 1)
```

```
}
}
```

Again:

- In the case of ++i (prefix increment), the value of i is increased by 1 before the operation, and this new value is used in the expression.
- In the case of i++ (postfix increment), the current value of i is used in the expression, and only after that is i increased by 1.
- The same applies to decrement (--i and i--).

Loops: while / do ... while

Loops allow you to execute some code several times in a row. Each such repetition is called an **iteration**.

while Loop - Loop with *Precondition*

The **while** loop runs as long as the given condition is true (has the value **true**):

```
// loop with precondition
while (expression) { // expression - predicate
...
// Loop body
...
}

public class WhileLoop {
    public static void main(String[] args) {
        int j = 6;
        while (j > 0) {
            System.out.println(j);
            j--;
        }
    }
}
```

Predicate — the condition specified in parentheses after the while keyword and evaluated on each iteration. **Loop body** — a block of code in curly braces, similar to a code block in a method. All constants or variables defined inside this block will only be visible inside this block.

Practice:

- Print 10 lines: "Task1". "Task2". ... "Task10". Use the *while* loop
- Print all numbers from 1 to 100 that are divisible by 5 without a remainder. Use the *do-while* loop

- Print only 7 numbers from 1 to 100 that are divisible by 5 without a remainder. *Use the `while` loop*
- Even numbers: Write a program that prints all even numbers from **A** to **B**. *Use the `do-while` loop*
- Countdown: Write a program that starts a countdown from **A** to 0 with a pause of 1 second between numbers. (Hint: use `Thread.sleep(1000);` for the pause). *Use the `while` loop*
- Multiplication Table: Write a program that prints the multiplication table for the number 5 (from 1 to 10). *Use the `do-while` loop*

// Print 10 lines: "Task1". "Task2". ... "Task10". Use the `while` loop

```
public class Main {
    public static void main(String[] args) {
        int i = 1;
        while (i <= 10) {
            System.out.println("Task" + i);
            i++;
        }
    }
}
```

// Print all numbers from 1 to 100 that are divisible by 5 without a remainder

```
public class Main {
    public static void main(String[] args) {
        int i = 1;
        do {
            if (i % 5 == 0) {
                System.out.println(i);
            }
            i++;
        } while (i <= 100);
    }
}
```

// Print only 7 numbers from 1 to 100 that are divisible by 5 without a remainder

```
public class Main {
    public static void main(String[] args) {
        int i = 1;
        int count = 0;
        while (count < 7) {
            if (i % 5 == 0) {
```

```

        System.out.println(i);
        count++;
    }
    i++;
}
}
}

```

do-while Loop - Loop with *Postcondition*

- The **do** loop first executes the loop code, and then checks the condition in the **while** statement. And as long as this condition is true, the loop repeats.

```

// loop with postcondition
do {
    ...
    // Loop body
    ...
} while (expression) //expression - predicate

```

```

public class DoLoop {
    public static void main(String[] args) {
        int j = 6;
        do {
            System.out.println(j);
            j--;
        } while (j > 0);
    }
}

```

It is important to note that the **do** loop guarantees at least one execution of the actions, even if the condition in the **while** statement is not true.

Инкремент и декремент в Java

Определение

Инкремент

Инкремент — это операция, которая увеличивает значение переменной на единицу. В Java для этого используется оператор **++**.

Декремент

Декремент — это операция, которая уменьшает значение переменной на единицу. В Java для этого используется оператор **--**.

Префиксная и постфиксная формы

Префиксный инкремент (++i)

В этой форме сначала увеличивается значение переменной на 1, а затем оно используется в выражении.

Пример:

```
public class IncidentCounterexample {

    public static void main(String[] args) {
        int counter = 123;           // 123
        counter++;                    // 123 (+1 после этой строки, т.е. с
        System.out.println(++counter); // 125 (т.к. ++counter, т.е. к прош.
    }

    public static void prefixIncrement() {
        int i = 5;
        int j = ++i; // i = 6, j = 6
        System.out.println(j); // 6
        System.out.println(i); // 6
    }

    public static void postfixIncrement() {
        int i = 5;
        int j = i++; // i = 5 (+1), j = 5
        System.out.println(j); // 5
        System.out.println(i); // 6
    }

    public static void prefixDecrement() {
        int i = 5;
        int j = --i; // i = 4, j = 4
        System.out.println(j); // 4
        System.out.println(i); // 4
    }

    public static void postfixDecrement() {
        int i = 5;
        int j = i--; // i = 5 (-1), j = 5
        System.out.println(j); // 5
        System.out.println(i); // 4
    }
}
```

```
}  
}
```

еще раз:

- В случае с `++i` (префиксный инкремент), значение `i` увеличивается на 1 до выполнения операции, и это новое значение используется в выражении.
- В случае с `i++` (постфиксный инкремент), текущее значение `i` используется в выражении, и только после этого `i` увеличивается на 1.
- То же самое относится и к декременту (`--i` и `i--`).

Циклы: "while" / "do ... while"

Циклы позволяют выполнить некий код несколько раз подряд. Каждый такой повтор называется **итерацией**.

Цикл **while** - цикл с *предусловием*

Цикл **while** выполняется до тех пор, пока заданное условие является верным (имеет значение **true**):

```
// цикл с предусловием  
while (expression) { // expression - предикат  
    ...  
// Тело цикла  
    ...  
}
```

```
public class WhileLoop {  
    public static void main(String[] args) {  
        int j = 6;  
        while (j > 0) {  
            System.out.println(j);  
            j--;  
        }  
    }  
}
```

Предикат — условие, которое указывается в скобках после ключевого слова `while` и вычисляется на каждой итерации.

Тело цикла — блок кода в фигурных скобках, аналогичный блоку кода в методе. Все константы или переменные, определенные внутри этого блока, будут видны только внутри этого блока.

цикл **do-while** - цикл с *постусловием*

- Цикл `do` сначала выполняет код цикла, а потом проверяет условие в инструкции `while`. И пока это условие истинно, цикл повторяется.

```
// цикл с постусловием
do {
    ...
    // Тело цикла
    ...
} while (expression) //expression - предикат
```

```
public class DoLoop {
    public static void main(String[] args) {
        int j = 6;
        do {
            System.out.println(j);
            j--;
        } while (j > 0);
    }
}
```

Важно отметить, что цикл `do` гарантирует хотя бы однократное выполнение действий, даже если условие в инструкции `while` не будет истинно.


Практика:

- Распечатать 10 строк: "Task1". "Task2". ... "Task10". *Используем цикл `while`*
- Распечатать все числа от 1 до 100, которые делятся на 5 без остатка. *Используем цикл `do-while`*
- Распечатать только 7 чисел от 1 до 100, которые делятся на 5 без остатка. *Используем цикл `while`*
- Четные числа: Напишите программу, которая выводит все четные числа от **A** до **B**. *Используем цикл `do-while`*
- Обратный отсчет: Напишите программу, которая начинает обратный отсчет от **A** до 0 с паузой в 1 секунду между числами. (Подсказка: используйте `Thread.sleep(1000)`; для паузы). *Используем цикл `while`*
- Таблица умножения: Напишите программу, которая выводит таблицу умножения для числа 5 (от 1 до 10). *Используем цикл `do-while`*


```
// Распечатать 10 строк: "Task1". "Task2". ... "Task10". Используем цикл while
public class Main {
    public static void main(String[] args) {
        int i = 1;
        while (i <= 10) {
            System.out.println("Task" + i);
        }
    }
}
```

```
        i++;
    }
}

// Распечатать все числа от 1 до 100, которые делятся на 5 без остатка. Исп
public class Main {
    public static void main(String[] args) {
        int i = 1;
        do {
            if (i % 5 == 0) {
                System.out.println(i);
            }
            i++;
        } while (i <= 100);
    }
}
```



```
// Распечатать только 7 чисел от 1 до 100, которые делятся на 5 без остатка
public class Main {
    public static void main(String[] args) {
        int i = 1;
        int count = 0;
        while (count < 7) {
            if (i % 5 == 0) {
                System.out.println(i);
                count++;
            }
            i++;
        }
    }
}
```



Homework

Task

Print 10 lines: "Task1". "Task2". ... "Task10". Use the `while` loop

Task

Print all numbers from 1 to 100 that are divisible by 5 without a remainder. Use the **do-while** loop

Task

Using the **while** loop, write a program that displays the cube of numbers from 1 to a given number n **Example:** for the number n=3

- 1 cubed is 1
- 2 cubed is 8
- 3 cubed is 27

Task

Using the **while** loop, write a program that displays the result of multiplying a given number n by all integers from 0 to n **Example:** for the number 3 the result will be

- 0
- 3
- 6
- 9

Задача

Распечатать 10 строк: "Task1". "Task2". ... "Task10". Используем цикл while

Задача

Распечатать все числа от 1 до 100, которые делятся на 5 без остатка. Используем цикл do-while

Задача

С помощью цикла while написать программу, выводящую на экран куб числа от 1 до заданного числа n

Пример: для числа n=3

- 1 в кубе 1
- 2 в кубе 8
- 3 в кубе 27

Задача

С помощью цикла while написать программу, выводящую на экран результат умножения данного числа n на все целые числа от 0 до n

Пример: для числа 3 результат будет

- 0
- 3
- 6
- 9

Code

code/HwSolution_07/src/SalaryCalculator.java

```
/**
 * @author Andrej Reutow
 * created on 15.09.2023
 */

// У Пети есть два ведра: одно пустое, а в другом 100 яблок.
// Ваша задача помочь Пете переложить все яблоки из полного ведра в пустое.
// Как только все яблоки будут переложены, Петя может идти гулять.
// Требования:
// Используйте цикл while для перекладывания яблок.
// После каждого перекладывания яблока выводите количество оставшихся ;
// Когда все яблоки будут переложены, выведите сообщение, что Петя мож

public class SalaryCalculator {

    static final double BASE_SALARY = 500;
    static final double BONUS_3 = .1;
    static final double BONUS_5 = .5;
    static final double BONUS_10 = 1;
    static final double BONUS_15 = 1.5;

    public static void main(String[] args) {
        int employerExperience = 4;
        double bonus = calculateBonus(employerExperience);
        System.out.println("Бонус для " + employerExperience + " лет: " + bonus);
        double result = BASE_SALARY + bonus;
        System.out.println("Зарплата с учетом бонуса за стаж: " + result);

        employerExperience = 20;
        bonus = calculateBonus(employerExperience);
        System.out.println("Бонус для " + employerExperience + " лет: " + bonus);
        result = BASE_SALARY + bonus;
        System.out.println("Зарплата с учетом бонуса за стаж: " + result);
    }
}
```

```

        employerExperience = 1;
        bonus = calculateBonus(employerExperience);
        System.out.println("Бонус для " + employerExperience + " лет: " + bonus);
        result = BASE_SALARY + bonus;
        System.out.println("Зарплата с учетом бонуса за стаж: " + result);
    }

    public static double calculateBonus(int experience) {
        double bonus = 0;

        if (experience >= 3 && experience <= 4) {
            bonus = BASE_SALARY * BONUS_3;
        } else if (experience >= 5 && experience < 10) {
            bonus = BASE_SALARY * BONUS_5;
        } else if (experience >= 10 && experience < 15) {
            bonus = BASE_SALARY * BONUS_10;
        } else if (experience >= 15) {
            bonus = BASE_SALARY * BONUS_15;
        }

        return bonus;
    }
}

```

code/Lesson_09/src/LoopWhileDoWhile.java

```

import java.util.Scanner;

/**
 * @author Andrej Reutow
 * created on 15.09.2023
 */
public class LoopWhileDoWhile {
    public static void main(String[] args) {

        System.out.println("START");
        int counter = 6;

        // while (counter <= 5) {
        //     counter += 1;
        //     System.out.println("Привет, я while");
        //     System.out.println(counter);
        // }
    }
}

```

```
//      do {
//          System.out.println("Привет, я while");
//          System.out.println(counter);
//      } while (counter <= 5);

int menuPoint = 0;
Scanner scanner = new Scanner(System.in);
while (menuPoint != 4) {
    System.out.println("1");
    System.out.println("2");
    System.out.println("3");
    System.out.println("4 - ВЫЙТИ");
    menuPoint = scanner.nextInt();
    if (menuPoint == 1) {
        System.out.println("Menu point 1");
    }
    if (menuPoint == 2) {
        System.out.println("Menu point 2");
    }
    if (menuPoint == 3) {
        System.out.println("Menu point 3");
    }
    if (menuPoint == 4) {
        System.out.println("вы вышли из меню");
    }
    if (menuPoint > 4) {
        System.out.println("Error!");
    }
}

    System.out.println("END");
}
}
```

code/Lesson_09/src/BucketTask.java

```
/**
 * @author Andrej Reutow
 * created on 15.09.2023
 */

//      У Пети есть два ведра: одно пустое, а в другом 100 яблок.
//      Ваша задача помочь Пете переложить все яблоки из полного ведра в пустое.
//      Как только все яблоки будут переложены, Петя может идти гулять.
```

```
// Требования:
//     Используйте цикл while для перекалывания яблок.
//     После каждого перекалывания яблока выводите количество оставшихся ;
//     Когда все яблоки будут переложены, выведите сообщение, что Петя мож

public class BucketTask {

    public static void main(String[] args) {
        int bucketA = 100; // полное ведро
        int bucketB = 0; // пустое ведро

        // 1 - bucketA <= 100 - true
        // 1 - {bucketA = 100 - 1, bucketA = 0 + 1 ...}
        // 2 - bucketA <= 100 = (bucketA - 99, bucketB - 1)

        //     while (bucketA <= 100 & bucketB <= 99) {
        //         bucketA -= 1;
        //         bucketB += 1;
        //         System.out.println("В ведре A " + bucketA + " яблок");
        //         System.out.println("В ведре B " + bucketB + " яблок");
        //     }

        //     while (bucketA != 0) {
        //         bucketA -= 1;
        //         bucketB += 1;
        //         System.out.println("В ведре A " + bucketA + " яблок");
        //         System.out.println("В ведре B " + bucketB + " яблок");
        //     }

        while (bucketB != 100) {
            bucketA -= 1;
            bucketB += 1;
            System.out.println("В ведре A " + bucketA + " яблок");
            System.out.println("В ведре B " + bucketB + " яблок");
        }
    }
}
```