Лабораторная работа № 1

Титульный лист

Дисциплина: Объектно-ориентированное программирование
Название работы: Знакомство с компилятором, запуском виртуальной машины Java и работой с пакетами.
Выполнил: Мостовщиков Владимир Витальевич
Группа: 6204-010302D
Преподаватель: Борисов Дмитрий Сергеевич
Год: 2025

Содержание:

1. Задание 1. Ознакомление с параметрами компилятора и JVM
2. Задание 2. Создание класса и точка входа
3. Задание 3. Аргументы командной строки
4. Задание 4. Работа с двумя классами
5. Задание 5. Пакеты и импорт
6. Задание 6. Создание JAR архива
7. Выводы

Задание 1. Ознакомление с параметрами компилятора и JVM

Для выполнения первого задания нужно было запустить компилятор javac и программу java без указания параметров.

Ход работы:

```
/*
$ javac
Usage: javac <options> <source files>
 -d <dir>       записывает классы в указанный каталог
 -classpath <path>  указывает, где искать классы для компиляции
 -sourcepath <path> указывает, где искать исходники
…

$ java
Usage: java [options] class [args]
   or  java [options] -jar jarfile [args]
…
 -cp <classpath>    задаёт путь к классам и JAR-файлам
 -jar <file>        запускает класс из указанного JAR
*/
```

Задание 2. Создание класса и точка входа

Во втором задании я создал файл MyFirstProgram.java, и пустой класс MyFirstClass, добавил в него метод main.

Ход работы:

1. Создаём исходный файл с пустым классом:

```
class MyFirstClass {
}
```

Команда компиляции:

javac MyFirstProgram.java

Компилятор успешно создаёт файл MyFirstClass.class. Попытка запуска приводит к ошибке отсутствия метода main.

2. Добавляем метод main без модификатора static:

```
class MyFirstClass {
  void main(String[] s) {
    System.out.println("Hello world!!!");
  }
}
```

Компилятор принимает код, но запуск снова заканчивается ошибкой: JVM ожидает сигнатуру public static void main(String[] args).

3. Добавляем модификаторы public и static:

```
class MyFirstClass {
  public static void main(String[] s) {
    System.out.println("Hello world!!!");
  }
}
```

Теперь программа успешно компилируется и запускается:

```
javac MyFirstProgram.java
java MyFirstClass
Hello world!!!
```

Задание 3. Аргументы командной строки

Третье задание требует заменить тело метода main так, чтобы программа выводила переданные аргументы.

Ход работы:

Изменяем метод main следующим образом:

```
class MyFirstClass {
  public static void main(String[] s) {
    for (int i = 0; i < s.length; i++) {
      System.out.println(s[i]);
    }
  }
}
```

Перекомпилируем и запускаем программу, передавая пять аргументов:

```
javac MyFirstProgram.java
java MyFirstClass arg1 arg2 arg3 arg4 arg5
arg1
arg2
arg3
arg4
arg5
```

Программа успешно обрабатывает параметры командной строки.

Задание 4. Работа с двумя классами

В этом задании необходимо добавить второй класс и реализовать в нём простую логику, а также использовать его в main.

Код MyFirstProgram.java:

```
class MySecondClass {
  private int a;
  private int b;

  MySecondClass(int a, int b) {
    this.a = a;
    this.b = b;
  }

  public int getA() { return a; }
  public int getB() { return b; }
  public void setA(int a) { this.a = a; }
  public void setB(int b) { this.b = b; }

  public int subtraction() {
    return a - b;
  }
}

class MyFirstClass {
  public static void main(String[] s) {
    MySecondClass o = new MySecondClass(0, 0);
    for (int i = 1; i <= 8; i++) {
      for (int j = 1; j <= 8; j++) {
        o.setA(i);
        o.setB(j);
        System.out.print(o.subtraction());
```

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

Ход работы:

1. Во втором классе объявлены два закрытых поля a и b и конструктор, инициализирующий их. Реализованы геттеры и сеттеры.
2. Метод subtraction() возвращает разность полей.
3. В main создаётся объект MySecondClass, затем в двойном цикле от 1 до 8 устанавливаются значения полей и выводится результат вычитания.
4. Команды:

```
javac MyFirstProgram.java
java MyFirstClass
```

Вывод:

```
0 -1 -2 -3 -4 -5 -6 -7
1 0 -1 -2 -3 -4 -5 -6
2 1 0 -1 -2 -3 -4 -5
3 2 1 0 -1 -2 -3 -4
4 3 2 1 0 -1 -2 -3
5 4 3 2 1 0 -1 -2
6 5 4 3 2 1 0 -1
7 6 5 4 3 2 1 0
```

Задание 5. Пакеты и импорт

Далее было необходимо вынести класс MySecondClass в отдельный пакет.

Ход работы:

1. Удалил все скомпилированные файлы (*.class).
2. Создал подкаталог myfirstpackage и файл MySecondClass.java со следующим содержимым:

```
package myfirstpackage;

public class MySecondClass {
  private int a;
  private int b;
  public MySecondClass(int a, int b) {
```

```
    this.a = a;
    this.b = b;
  }
  public int getA() { return a; }
  public int getB() { return b; }
  public void setA(int a) { this.a = a; }
  public void setB(int b) { this.b = b; }
  public int subtraction() { return a - b; }
}
```

3. В файле MyFirstProgram.java добавили строку import myfirstpackage.MySecondClass; и оставили MyFirstClass без объявленного пакета:

```
import myfirstpackage.MySecondClass;

class MyFirstClass {
  public static void main(String[] s) {
    MySecondClass o = new MySecondClass(0, 0);
    for (int i = 1; i <= 8; i++) {
      for (int j = 1; j <= 8; j++) {
        o.setA(i);
        o.setB(j);
        System.out.print(o.subtraction());
        System.out.print(" ");
      }
      System.out.println();
    }
  }
}
```

4. Компилируем, указав каталог вывода out:

```
mkdir -p out
javac -d out myfirstpackage/MySecondClass.java MyFirstProgram.java
```

5. Запускаем программу, указав classpath:

```
java -cp out MyFirstClass
```

Программа корректно работает при разнесении классов по пакетам. Для доступа к классу из другого пакета он объявлен public, а его файл должен иметь имя, совпадающее с именем класса

Задание 6. Создание JAR-архива

Финальное задание требует собрать исполняемый JAR-архив

Ход работы:

1. Создал каталог classes и скопировали в него скомпилированные файлы из каталога out задания 5, сохраняя структуру каталогов.
2. Создал файл manifest.mf со следующим содержимым

```
Manifest-Version: 1.0
Created-By: Mostovshikov Vladimir
Main-Class: MyFirstClass
```

3. Сформировал архив командой:

```
jar cfm myfirst.jar manifest.mf -C classes .
```

4. Переместил архив в папку MyJar и проверили его запуск:

```
mkdir MyJar
mv myfirst.jar MyJar/
java -jar MyJar/myfirst.jar
```

Программа запустилась без ошибок и вывела ту же таблицу, что в пятом задании.

Файл manifest.mf должен содержать атрибут Main-Class и заканчиваться пустой строкой, иначе JVM не сможет найти точку входа.

Выводы

В ходе лабораторной работы я научился:

- Запускать компилятор Java (javac) и виртуальную машину (java) из консоли и использовать их ключи для задания путей и каталогов.
- Создавать простейшие классы, определять точку входа программы и различать имена файла и класса при компиляции и запуске
- Обрабатывать аргументы командной строки в методе main.
- Разрабатывать классы с полями, конструкторами, методами доступа и выполняющими действия; использовать их из других классов
- Организовывать код в пакеты, выносить классы в отдельные каталоги, импортировать их, правильно объявлять модификаторы доступа и имена файлов
- Формировать исполняемые JAR-архивы с собственным манифестом и запускать их