

Лабораторная работа № 1

Выполнил: Мурзяков Егор 6204 – 010302D

Ход выполнения задания № 1

Запуск компилятора javac без параметров выдал список команд и их описание (рис. 1)

```
PS C:\Users\Mi\Labs\Lab_1\Task_2> javac
Usage: javac <options> <source files>
where possible options include:
  @<filename>           Read options and filenames from file
  -Akey[=value]         Options to pass to annotation processors
  --add-modules <module>(<module>)*
                        Root modules to resolve in addition to the initial modules,
                        or all modules on the module path if <module> is ALL-MODULE-PATH.
  --boot-class-path <path>, -bootclasspath <path>
                        Override location of bootstrap class files
  --class-path <path>, -classpath <path>, -cp <path>
                        Specify where to find user class files and annotation processors
  -d <directory>        Specify where to place generated class files
  -deprecation           Output source locations where deprecated APIs are used
```

Рис. 1

Запуск программы java без параметров так же выдал список команд и их описание (рис. 2)

```
PS C:\Users\Mi\Labs\Lab_1\Task_2> java
Usage: java [java options...] <application> [application arguments...]

Where <application> is one of:
  <mainclass>           to execute the main method of a compiled main class
  -jar <jarfile>.jar     to execute the main class of a JAR archive
  -m <module>[/<mainclass>] to execute the main class of a module
  <sourcefile>.java     to compile and execute a source-file program

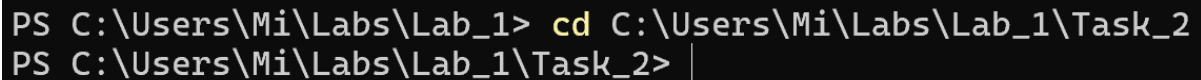
Where key java options include:
  --class-path <class path>
    where <class path> is a list of directories and JAR archives to search for class files, separated by ";"
  --module-path <module path>
    where <module path> is a list of directories and JAR archives to search for modules, separated by ";"
  -version
    to print product version to the error stream and exit

For additional help on usage:      java --help
For an interactive Java environment: jshell
```

Рис. 2

Ход выполнения задания № 2

Согласно заданию, создал файл “MyFirstProgram.java”, содержащий исходный код одного пустого класса с именем “MyFirstClass”. В консоли ввел команду “cd C:\Users\Mi\Labs\Lab_1\Task_2” чтобы сменить текущую папку (рис. 3)



```
PS C:\Users\Mi\Labs\Lab_1> cd C:\Users\Mi\Labs\Lab_1\Task_2
PS C:\Users\Mi\Labs\Lab_1\Task_2> |
```

Рис. 3

Далее откомпилировал ранее написанный код с помощью команды “javac MyFirstProgram.java”, после чего в папке с кодом получил файл “MyFirstClass.class”. Чтобы запустить программу, ввел следующий код “java MyFirstClass”.

Следуя инструкциям из задания, дописал тело класса:

```
class MyFirstClass {

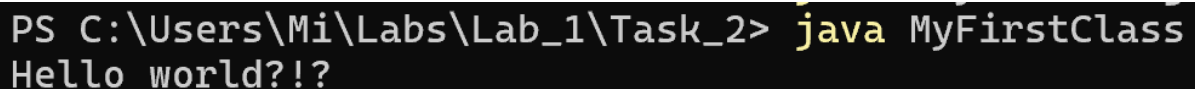
void main(String[] s) {

System.out.println(“Hello world?!?”);

}

}
```

Для того чтобы программа заработала, сделал метод main() статическим. Провел компиляцию и запустил программу (рис. 4)



```
PS C:\Users\Mi\Labs\Lab_1\Task_2> java MyFirstClass
Hello world?!?
```

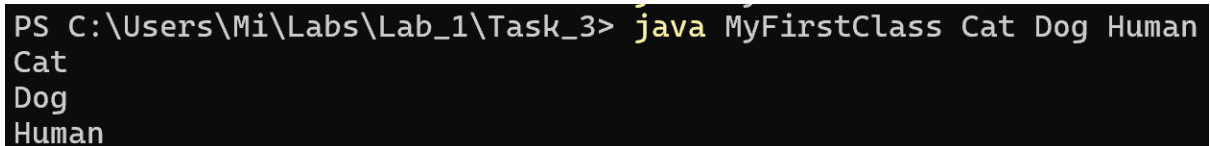
Рис. 4

Ход выполнения задания № 3

Следуя инструкциям из задания, получил следующий код:

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

Откомпилировал и запустил программу, введя в командную строку ряд аргументов. Программа вывела каждый аргумент с новой строки (рис. 5)



```
PS C:\Users\Mi\Labs\Lab_1\Task_3> java MyFirstClass Cat Dog Human  
Cat  
Dog  
Human
```

Рис. 5

Ход выполнения задания № 4

Изменил код согласно заданию (рис. 6)

```
class MyFirstClass {
    void main() {
        MySecondClass mySecClass = new MySecondClass();
        int i, j;
        for (i = 1; i <= 8; i++){
            for(j = 1; j <= 8; j++){
                mySecClass.SetFieldA(i);
                mySecClass.SetFieldB(j);
                System.out.print(mySecClass.Action());
                System.out.print(" ");
            }
            System.out.println();
        }
    }
}

class MySecondClass {
    private int fieldA;
    private int fieldB;

    public MySecondClass() { // Конструктор по умолчанию
        this.fieldA = 0;
        this.fieldB = 0;
    }
    public MySecondClass(int _a, int _b){ // Конструктор, инициализирующий значение полей входящими значениями
        this.fieldA = _a;
        this.fieldB = _b;
    }

    public void SetFieldA(int _a) { // Изменение значений
        fieldA = _a;
    }
    public void SetFieldB(int _b) {
        fieldB = _b;
    }

    public int GetFieldA() { // Получение значений
        return fieldA;
    }
    public int GetFieldB() {
        return fieldB;
    }

    public int Action() { // A * B - A + B
        return fieldA * fieldB - fieldA + fieldB;
    }
}
```

Рис. 6

Результат выполнения программы (рис. 7)

```
PS C:\Users\Mi\Labs\Lab_1\Task_3> cd C:\Users\Mi\Labs\Lab_1\Task_4
PS C:\Users\Mi\Labs\Lab_1\Task_4> java MyFirstClass
1 3 5 7 9 11 13 15
1 4 7 10 13 16 19 22
1 5 9 13 17 21 25 29
1 6 11 16 21 26 31 36
1 7 13 19 25 31 37 43
1 8 15 22 29 36 43 50
1 9 17 25 33 41 49 57
1 10 19 28 37 46 55 64
PS C:\Users\Mi\Labs\Lab_1\Task_4> |
```

Рис. 7

Ход выполнения задания № 5

Согласно заданию, удалил все байт-коды классов, вынес код класса “MySecondClass” в отдельный файл, изменил код в файле “MyFirstProgram.java”, добавив в начало “import myfirstpackage.*;”. После компиляции возникли некоторые ошибки, чтобы их исправить внес следующие изменения:

1. Добавил модификатор доступа public в класс “MySecondClass”
2. Добавил модификатор доступа public в класс “MyFirstClass”
3. Изменил названия файлов, переименовав их в названия классов, содержащихся в них

Эти изменения позволили программе корректно отработать (рис. 8)

```
PS C:\Users\Mi\Labs\Lab_1\Task_4> cd C:\Users\Mi\Labs\Lab_1\Task_5
PS C:\Users\Mi\Labs\Lab_1\Task_5> java MyFirstClass
1 3 5 7 9 11 13 15
1 4 7 10 13 16 19 22
1 5 9 13 17 21 25 29
1 6 11 16 21 26 31 36
1 7 13 19 25 31 37 43
1 8 15 22 29 36 43 50
1 9 17 25 33 41 49 57
1 10 19 28 37 46 55 64
```

Рис. 8

Ход выполнения задания № 6

Запусти программу “jar”, а затем “jar --help” и ознакомился с форматом задания ключей для создания архивов (рис. 9)

```
PS C:\Users\Mi\Labs\Lab_1\Task_6> jar
Usage: jar [OPTION...] [ [--release VERSION] [-C dir] files] ...
Try 'jar --help' for more information.
PS C:\Users\Mi\Labs\Lab_1\Task_6> jar --help
Usage: jar [OPTION...] [ [--release VERSION] [-C dir] files] ...
jar creates an archive for classes and resources, and can manipulate or
restore individual classes or resources from an archive.

Examples:
# Create an archive called classes.jar with two class files:
jar --create --file classes.jar Foo.class Bar.class
# Create an archive using an existing manifest, with all the files in foo/:
```

Рис. 9

Выполнив инструкции из задания, создал файл “manifest.mf”, содержащий следующий код:

Manifest-Version: 1.0

Created-By: Murzyakov

Main-Class: MyFirstClass

Далее ввел в командную строку “jar cvfm myfirst.jar manifest.mf MyFirstClass.class myfirstpackage”, которая создала архив “myfirst.jar”. Для его запуска воспользовался программой “java -jar myfirst.jar” (рис. 10)

```
PS C:\Users\Mi\Labs\Lab_1\Task_6> java -jar myfirst.jar
1 3 5 7 9 11 13 15
1 4 7 10 13 16 19 22
1 5 9 13 17 21 25 29
1 6 11 16 21 26 31 36
1 7 13 19 25 31 37 43
1 8 15 22 29 36 43 50
1 9 17 25 33 41 49 57
1 10 19 28 37 46 55 64
```

Рис. 10