

Министерство науки и высшего образования Российской Федерации

федеральное государственное автономное  
образовательное учреждение высшего образования  
«Самарский национальный исследовательский университет  
имени академика С.П. Королева»

Институт информатики и кибернетики

Кафедра технической кибернетики

Отчет по лабораторной работе №1

Дисциплина: «ООП»

Тема «Пакеты»

Выполнил: Пантелеев Ю.В.

Группа: 6201-120303

Самара, 2025



```

    To be used in conjunction with either -source or --release.
-encoding <encoding>          Specify character encoding used by source files
-endorseddirs <dirs>          Override location of endorsed standards path
-extdirs <dirs>               Override location of installed extensions
-g                             Generate all debugging info
-g:{lines,vars,source}        Generate only some debugging info
-g:none                        Generate no debugging info
-h <directory>                Specify where to place generated native header files
--help, -help, -?             Print this help message
--help-extra, -X              Print help on extra options
-implicit:{none,class}        Specify whether to generate class files for implicitly referenced files
-J<flag>                      Pass <flag> directly to the runtime system
--limit-modules <module>(,<module>)*
    Limit the universe of observable modules
--module <module>(,<module>)*, -m <module>(,<module>)*
    Compile only the specified module(s), check timestamps
--module-path <path>, -p <path>
    Specify where to find application modules
--module-source-path <module-source-path>
    Specify where to find input source files for multiple modules
--module-version <version>
    Specify version of modules that are being compiled
-nowarn                        Generate no warnings
-parameters                   Generate metadata for reflection on method parameters
-proc:{none,only,full}        Control whether annotation processing and/or compilation is done.
-processor <class1>[,<class2>,<class3>...]
    Names of the annotation processors to run;
    bypasses default discovery process
--processor-module-path <path>
    Specify a module path where to find annotation processors
--processor-path <path>, -processorpath <path>
    Specify where to find annotation processors
-profile <profile>
    Check that API used is available in the specified profile.
    This option is deprecated and may be removed in a future release.
--release <release>
    Compile for the specified Java SE release.
    Supported releases:
        8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24
-s <directory>                Specify where to place generated source files
--source <release>, -source <release>
    Provide source compatibility with the specified Java SE release.
    Supported releases:
        8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24
--source-path <path>, -sourcepath <path>
    Specify where to find input source files
--system <jdk>|none            Override location of system modules
--target <release>, -target <release>
    Generate class files suitable for the specified Java SE release.
    Supported releases:
        8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24
--upgrade-module-path <path>
    Override location of upgradeable modules
-verbose                       Output messages about what the compiler is doing
--version, -version            Version information
-Werror                        Terminate compilation if warnings occur

```

C:\Users\yuriy>\_

Я ввел в консоль “java”.

## Результат работы 1.2

```
C:\Users\yuriy>java
Usage: java [java options...] <application> [application arguments...]

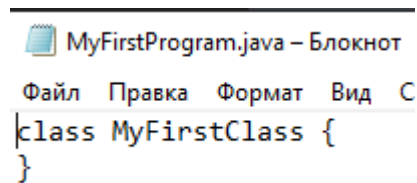
Where <application> is one of:
  <mainclass>          to execute the main method of a compiled main class
  -jar <jarfile>.jar    to execute the main class of a JAR archive
  -m <module>[/<mainclass>] to execute the main class of a module
  <sourcefile>.java    to compile and execute a source-file program

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
```

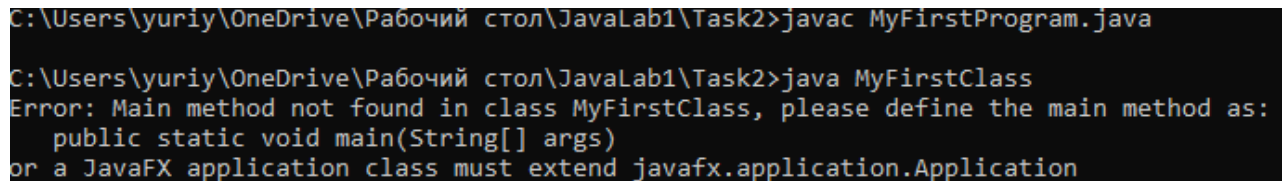
## Задание 2

Я создал файл “MyFirstProgram.java”.



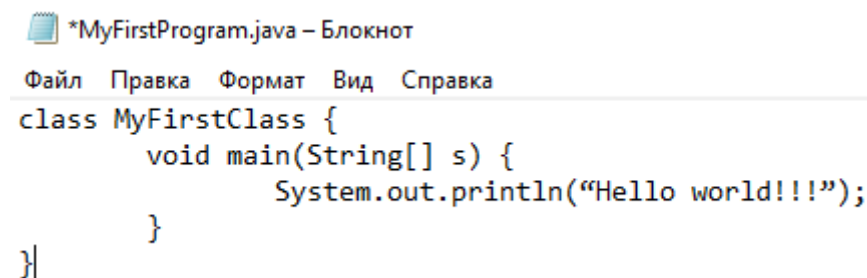
```
class MyFirstClass {  
}
```

Откомпилировал его и попытался запустить файл “MyFirstClass.class”.



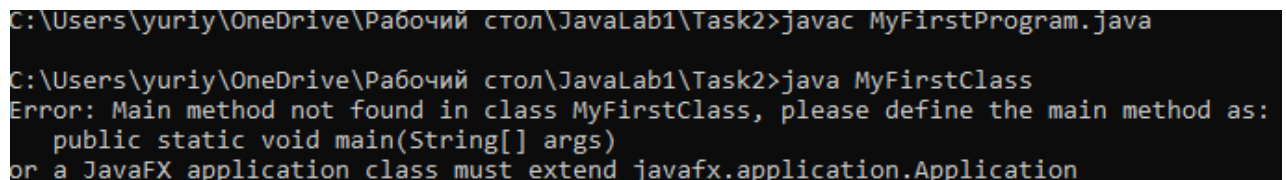
```
C:\Users\yuriy\OneDrive\Рабочий стол\JavaLab1\Task2>javac MyFirstProgram.java  
C:\Users\yuriy\OneDrive\Рабочий стол\JavaLab1\Task2>java MyFirstClass  
Error: Main method not found in class MyFirstClass, please define the main method as:  
    public static void main(String[] args)  
or a JavaFX application class must extend javafx.application.Application
```

Я добавил функцию main по заданию.



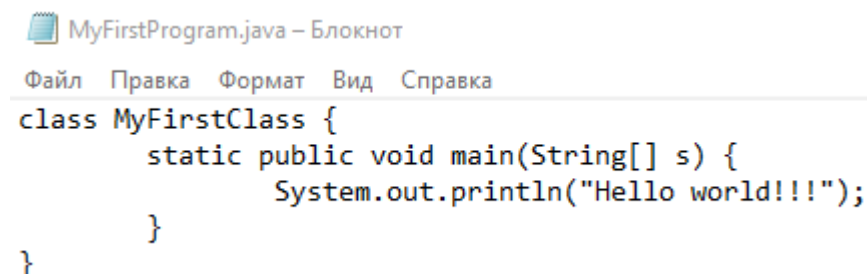
```
class MyFirstClass {  
    void main(String[] s) {  
        System.out.println("Hello world!!!");  
    }  
}
```

Снова попытался запустить файл “MyFirstClass.class”.



```
C:\Users\yuriy\OneDrive\Рабочий стол\JavaLab1\Task2>javac MyFirstProgram.java  
C:\Users\yuriy\OneDrive\Рабочий стол\JavaLab1\Task2>java MyFirstClass  
Error: Main method not found in class MyFirstClass, please define the main method as:  
    public static void main(String[] args)  
or a JavaFX application class must extend javafx.application.Application
```

Сделал функцию main публичной и статической.



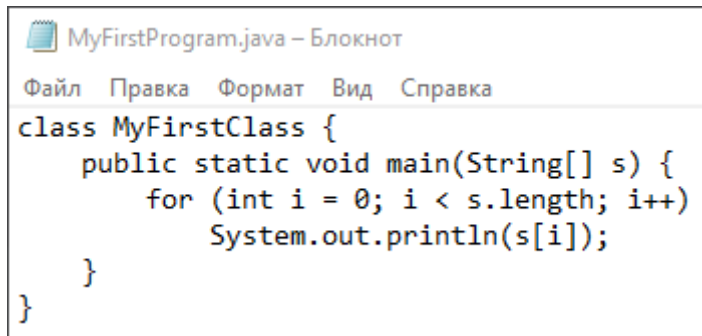
```
class MyFirstClass {  
    static public void main(String[] s) {  
        System.out.println("Hello world!!!");  
    }  
}
```

Снова попытался запустить файл “MyFirstClass.class”.

```
C:\Users\yuriy\OneDrive\Рабочий стол\JavaLab1\Task2>javac MyFirstProgram.java  
C:\Users\yuriy\OneDrive\Рабочий стол\JavaLab1\Task2>java MyFirstClass  
Hello world!!!
```

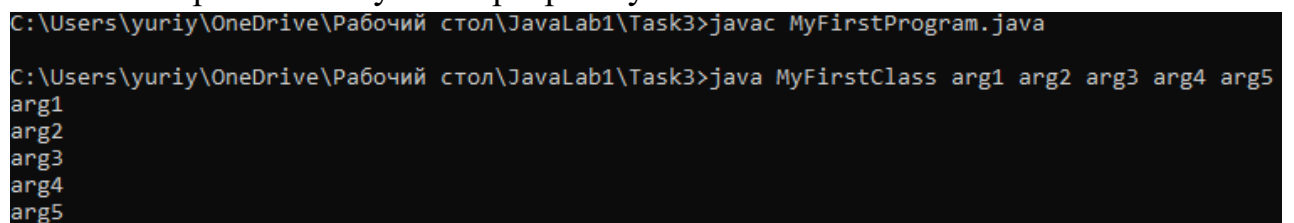
### Задание 3

Я переписал файл “MyFirstProgram.java” по заданию.



```
MyFirstProgram.java – Блокнот
Файл  Правка  Формат  Вид  Справка
class MyFirstClass {
    public static void main(String[] s) {
        for (int i = 0; i < s.length; i++)
            System.out.println(s[i]);
    }
}
```

Откомпилировал и запустил программу.



```
C:\Users\yuriy\OneDrive\Рабочий стол\JavaLab1\Task3>javac MyFirstProgram.java
C:\Users\yuriy\OneDrive\Рабочий стол\JavaLab1\Task3>java MyFirstClass arg1 arg2 arg3 arg4 arg5
arg1
arg2
arg3
arg4
arg5
```

## Задание 4

В файле “MyFirstProgram.java” после описания класса “MyFirstClass” я добавил описание второго класса “MySecondClass” и переписал “MyFirstClass” по заданию

```
MyFirstProgram.java – Блокнот
Файл  Правка  Формат  Вид  Справка
class MyFirstClass {
    public static void main(String[] args) {
        MySecondClass o = new MySecondClass(0,0);

        int i, j;
        for (i = 1; i <= 8; i++) {
            for(j = 1; j <= 8; j++) {
                o.setFirstNumber(i);
                o.setSecondNumber(j);
                System.out.print(o.mult());
                System.out.print(" ");
            }
            System.out.println();
        }
    }
}

class MySecondClass {
    private int x;
    private int y;

    public int getFirstNumber() {
        return x;
    }
    public int getSecondNumber() {
        return y;
    }

    public void setFirstNumber(int value) {
        x = value;
    }
    public void setSecondNumber(int value) {
        y = value;
    }

    public MySecondClass(int x1, int y1) {
        x = x1;
        y = y1;
    }

    public int mult() {
        return x * y;
    }
}
```



Откомпилировал и запустил программу.

```
C:\Users\yuriy\OneDrive\Рабочий стол\JavaLab1\Task4>javac MyFirstProgram.java
C:\Users\yuriy\OneDrive\Рабочий стол\JavaLab1\Task4>java MyFirstClass
1 2 3 4 5 6 7 8
2 4 6 8 10 12 14 16
3 6 9 12 15 18 21 24
4 8 12 16 20 24 28 32
5 10 15 20 25 30 35 40
6 12 18 24 30 36 42 48
7 14 21 28 35 42 49 56
8 16 24 32 40 48 56 64
```

## Задание 5

Я удалил все откомпилированные байт-коды классов, создал поддиректорию myfirstpackage.

```
C:\Users\yuriy\OneDrive\Рабочий стол\JavaLab1\Task5>del /s *.class
Удален файл - C:\Users\yuriy\OneDrive\Рабочий стол\JavaLab1\Task5\MyFirstClass.class
Удален файл - C:\Users\yuriy\OneDrive\Рабочий стол\JavaLab1\Task5\MySecondClass.class
C:\Users\yuriy\OneDrive\Рабочий стол\JavaLab1\Task5>mkdir myfirstpackage
```

Попытался откомпилировать файл “MyFirstPackage.java”

```
C:\Users\yuriy\OneDrive\Рабочий стол\JavaLab1\Task5>javac -d . myfirstpackage/MyFirstPackage.java
myfirstpackage\MyFirstPackage.java:3: error: class MySecondClass is public, should be declared in a file named MySecondClass.java
public class MySecondClass {
      ^
1 error
```

Выдало ошибку, так как название файла и класса отличаются. Изменил название класса на MyFirstPackage и снова попытался откомпилировать файл “MyFirstPackage.java”

```
C:\Users\yuriy\OneDrive\Рабочий стол\JavaLab1\Task5>javac -d . myfirstpackage/MyFirstPackage.java
myfirstpackage\MyFirstPackage.java:21: error: invalid method declaration; return type required
    public MyFirstClass(int x1, int y1) {
           ^
1 error
```

Снова выдало ошибку так как название метода класса отличается от названия файла. Изменил название метода класса на MyFirstPackage и успешно откомпилировал файл “MyFirstPackage.java”. После попытался откомпилировать файл “MyFirstProgram.java”.

```
C:\Users\yuriy\OneDrive\Рабочий стол\JavaLab1\Task5>javac MyFirstProgram.java
MyFirstProgram.java:5: error: cannot find symbol
        MySecondClass o = new MySecondClass(0,0);
        ^
symbol:   class MySecondClass
location: class MyFirstClass
MyFirstProgram.java:5: error: cannot find symbol
        MySecondClass o = new MySecondClass(0,0);
                                ^
symbol:   class MySecondClass
location: class MyFirstClass
2 errors
```

Выдало ошибку, так как класс MyFirstClass пытается вызвать метод MySecondClass, который я переименовал. Везде меняю старое название метода на новое и снова компилирую и запускаю программу.

Содержимое файлов:

1)



MyFirstProgram.java – Блокнот

Файл Правка Формат Вид Справка

---

```
import myfirstpackage.*;

class MyFirstClass {
    public static void main(String[] args) {
        MyFirstPackage o = new MyFirstPackage(0,0);

        int i, j;
        for (i = 1; i <= 8; i++) {
            for(j = 1; j <= 8; j++) {
                o.setFirstNumber(i);
                o.setSecondNumber(j);
                System.out.print(o.mult());
                System.out.print(" ");
            }
            System.out.println();
        }
    }
}
```

2)



MyFirstPackage.java – Блокнот

Файл Правка Формат Вид Справка

```
package myfirstpackage;

public class MyFirstPackage {
    private int x;
    private int y;

    public int getFirstNumber() {
        return x;
    }
    public int getSecondNumber() {
        return y;
    }

    public void setFirstNumber(int value) {
        x = value;
    }
    public void setSecondNumber(int value) {
        y = value;
    }

    public MyFirstPackage(int x1, int y1) {
        x = x1;
        y = y1;
    }

    public int mult() {
        return x * y;
    }
}
```

Результат:

```
C:\Users\yuriy\OneDrive\Рабочий стол\JavaLab1\Task5>javac MyFirstProgram.java
C:\Users\yuriy\OneDrive\Рабочий стол\JavaLab1\Task5>java MyFirstClass
1 2 3 4 5 6 7 8
2 4 6 8 10 12 14 16
3 6 9 12 15 18 21 24
4 8 12 16 20 24 28 32
5 10 15 20 25 30 35 40
6 12 18 24 30 36 42 48
7 14 21 28 35 42 49 56
8 16 24 32 40 48 56 64
```

## Задание 6

Запустил программу jar.

```
C:\Users\yuriy\OneDrive\Рабочий стол\JavaLab1\Task6>"C:\Program Files\Java\jdk-25\bin\jar.exe" --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/:
jar --create --file classes.jar --manifest mymanifest -C foo/ .
# Create a modular jar archive, where the module descriptor is located in
# classes/module-info.class:
jar --create --file foo.jar --main-class com.foo.Main --module-version 1.0
  -C foo/ classes resources
# Update an existing non-modular jar to a modular jar:
jar --update --file foo.jar --main-class com.foo.Main --module-version 1.0
  -C foo/ module-info.class
# Create a multi-release jar, placing some files in the META-INF/versions/9 directory:
jar --create --file mr.jar -C foo classes --release 9 -C foo9 classes

To shorten or simplify the jar command, you can specify arguments in a separate
text file and pass it to the jar command with the at sign (@) as a prefix.

Examples:
# Read additional options and list of class files from the file classes.list
jar --create --file my.jar @classes.list

Main operation mode:

  -c, --create          Create the archive. When the archive file name specified
                        by -f or --file contains a path, missing parent directories
                        will also be created
  -i, --generate-index=FILE Generate index information for the specified jar
                        archives. This option is deprecated and may be
                        removed in a future release.
  -t, --list            List the table of contents for the archive
  -u, --update          Update an existing jar archive
  -x, --extract         Extract named (or all) files from the archive.
                        If a file with the same name appears more than once in
                        the archive, each copy will be extracted, with later copies
                        overwriting (replacing) earlier copies unless -k is specified.
  -d, --describe-module Print the module descriptor, or automatic module name
  --validate           Validate the contents of the jar archive. This option:
                        - Validates that the API exported by a multi-release
                        jar archive is consistent across all different release
                        versions.
                        - Issues a warning if there are invalid or duplicate file names

Operation modifiers valid in any mode:

  -C DIR              Change to the specified directory and include the
                        following file. When used in extract mode, extracts
                        the jar to the specified directory
  -f, --file=FILE     The archive file name. When omitted, either stdin or
                        stdout is used based on the operation
  --release VERSION   Places all following files in a versioned directory
                        of the jar (i.e. META-INF/versions/VERSION/)
  -v, --verbose       Generate verbose output on standard output
```

Operation modifiers valid only in create and update mode:

-e, --main-class=CLASSNAME	The application entry point for stand-alone applications bundled into a modular, or executable, jar archive
-m, --manifest=FILE	Include the manifest information from the given manifest file
-M, --no-manifest	Do not create a manifest file for the entries
--module-version=VERSION	The module version, when creating a modular jar, or updating a non-modular jar
--hash-modules=PATTERN	Compute and record the hashes of modules matched by the given pattern and that depend upon directly or indirectly on a modular jar being created or a non-modular jar being updated
-p, --module-path	Location of module dependence for generating the hash

Operation modifiers valid only in create, update, and generate-index mode:

-0, --no-compress	Store only; use no ZIP compression
--date=TIMESTAMP	The timestamp in ISO-8601 extended offset date-time with optional time-zone format, to use for the timestamps of entries, e.g. "2022-02-12T12:30:00-05:00"

Operation modifiers valid only in extract mode:

-k, --keep-old-files	Do not overwrite existing files. If a Jar file entry with the same name exists in the target directory, the existing file will not be overwritten. As a result, if a file appears more than once in an archive, later copies will not overwrite earlier copies. Also note that some file system can be case insensitive.
--dir	Directory into which the jar will be extracted


Other options:

-?, -h, --help[:compat]	Give this, or optionally the compatibility, help
--help-extra	Give help on extra options
--version	Print program version

An archive is a modular jar if a module descriptor, 'module-info.class', is located in the root of the given directories, or the root of the jar archive itself. The following operations are only valid when creating a modular jar, or updating an existing non-modular jar: '--module-version', '--hash-modules', and '--module-path'.

Mandatory or optional arguments to long options are also mandatory or optional for any corresponding short options.

Создал файл “manifest.mf” и ввел в него всё по заданию.

 manifest.mf – Блокнот

Файл Правка Формат Вид Справка

Manifest-Version: 1.0

Created-By: Пантелеев

Main-Class: MyFirstClass

Создал архив myfirst.jar, переместив туда все классы.

```
C:\Users\yuriy\OneDrive\Рабочий стол\JavaLab1\Task6>"C:\Program Files\Java\jdk-25\bin\jar.exe" cfm myfirst.jar manifest.mf *.class myfirstpackage\*.class
```

Создал директорию “MyJar” и переместил туда созданный архив

```
C:\Users\yuriy\OneDrive\Рабочий стол\JavaLab1\Task6>mkdir MyJar  
C:\Users\yuriy\OneDrive\Рабочий стол\JavaLab1\Task6>move myfirst.jar MyJar\  
Перемещено файлов: 1.
```

Запустил JAR-архив из новой директории.

```
C:\Users\yuriy\OneDrive\Рабочий стол\JavaLab1\Task6>cd MyJar  
C:\Users\yuriy\OneDrive\Рабочий стол\JavaLab1\Task6\MyJar>java -jar myfirst.jar  
1 2 3 4 5 6 7 8  
2 4 6 8 10 12 14 16  
3 6 9 12 15 18 21 24  
4 8 12 16 20 24 28 32  
5 10 15 20 25 30 35 40  
6 12 18 24 30 36 42 48  
7 14 21 28 35 42 49 56  
8 16 24 32 40 48 56 64
```