

Sprawozdanie nr 1

Temat ćwiczenia:

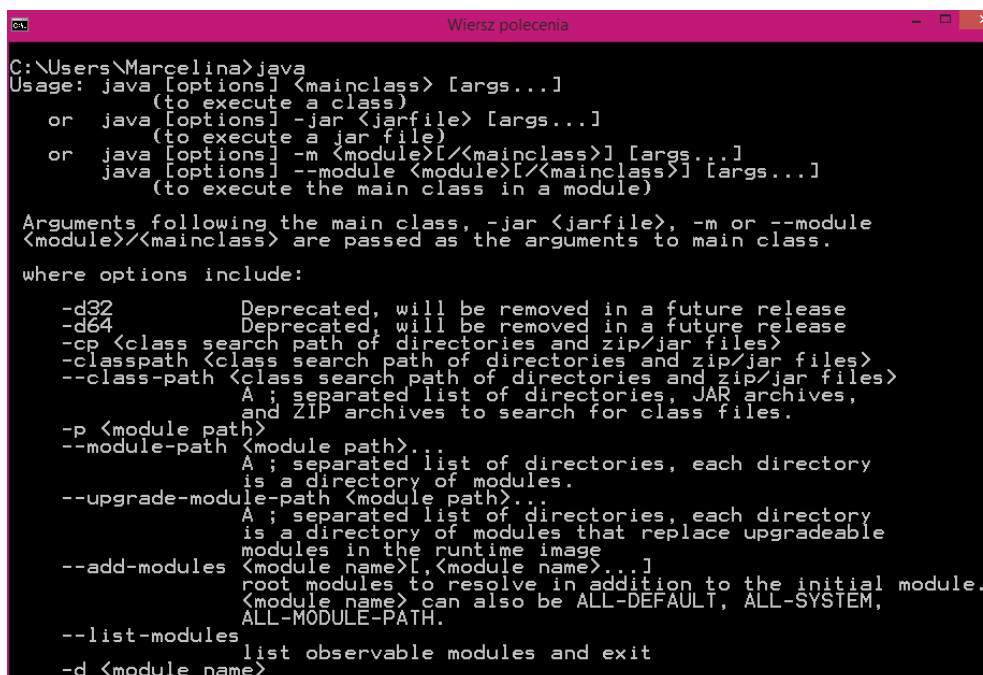
Zapoznanie się ze środowiskiem pracy.

Cel ćwiczenia

Celem ćwiczenia jest uruchomienie programu wypisującego tekst na ekran za pomocą wirtualnej maszyny w języku Java.

Wykonanie

Pierwszym krokiem do wykonania ćwiczenia było sprawdzenie czy na moim komputerze jest zainstalowana maszyna wirtualna Javy. W tym celu uruchomiłam polecenie java, po czym na konsoli zostały wyświetlone informacje, które oznaczały istnienie wcześniej wspomnianej maszyny wirtualnej.



```
C:\Users\Marcelina>java
Usage: java [options] <mainclass> [args...]
        (to execute a class)
or java [options] -jar <jarfile> [args...]
        (to execute a jar file)
or java [options] -m <module>[/<mainclass>] [args...]
        java [options] --module <module>[/<mainclass>] [args...]
        (to execute the main class in a module)

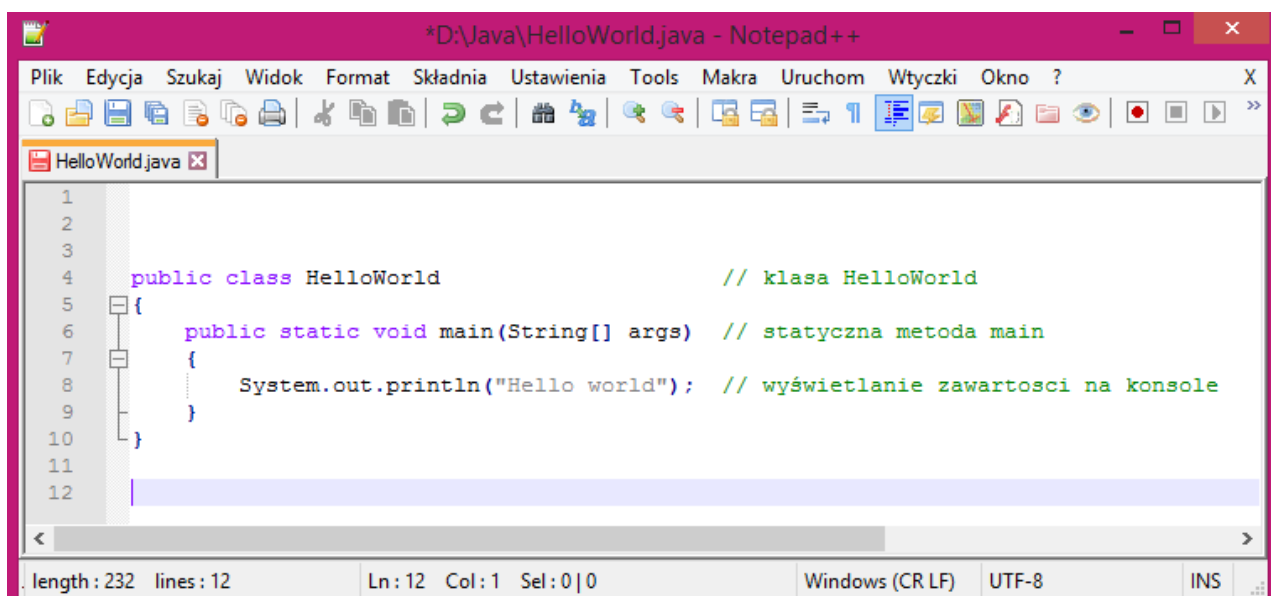
Arguments following the main class, -jar <jarfile>, -m or --module
<module>/<mainclass> are passed as the arguments to main class.

where options include:
  -d32           Deprecated, will be removed in a future release
  -d64           Deprecated, will be removed in a future release
  -cp <class search path of directories and zip/jar files>
  -classpath <class search path of directories and zip/jar files>
  --class-path <class search path of directories and zip/jar files>
                 A ; separated list of directories, JAR archives,
                 and ZIP archives to search for class files.
  -p <module path>
  --module-path <module path>...
                 A ; separated list of directories, each directory
                 is a directory of modules.
  --upgrade-module-path <module path>...
                 A ; separated list of directories, each directory
                 is a directory of modules that replace upgradeable
                 modules in the runtime image
  --add-modules <module name>[,<module name>...]
                 root modules to resolve in addition to the initial module.
                 <module name> can also be ALL-DEFAULT, ALL-SYSTEM,
                 ALL-MODULE-PATH.
  --list-modules
                 list observable modules and exit
  -d <module name>
```

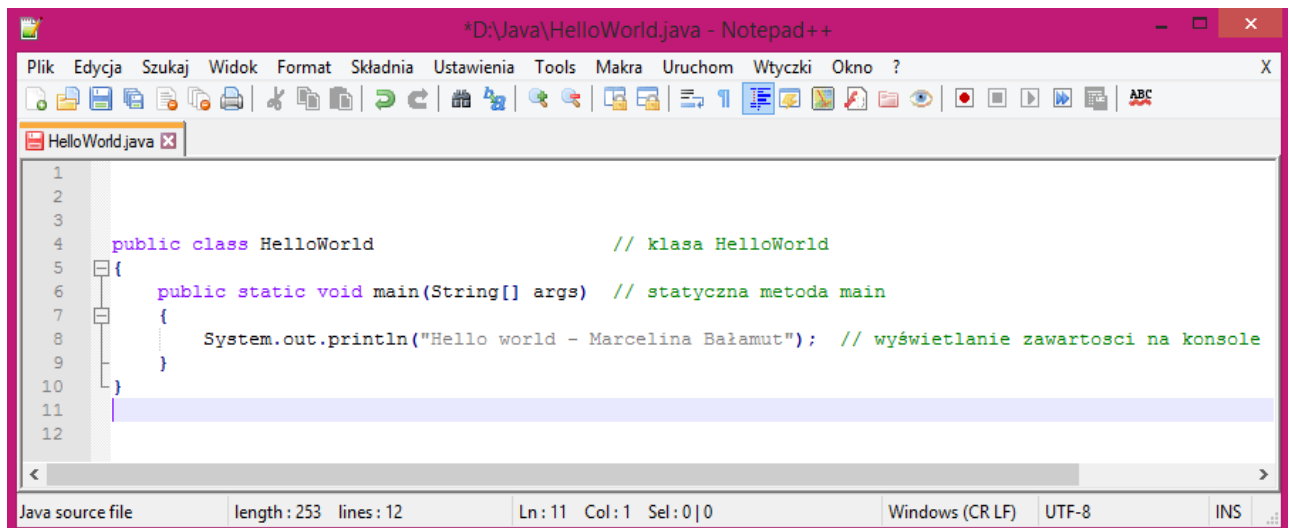
Kolejno wpisując komendę javac sprawdziłam czy na moim komputerze zainstalowany jest kompilator Javy ze skutkiem pozytywnym.

```
C:\Users\Marcelina>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 m
es
                        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
  -encoding <encoding> Specify character encoding used by source fil
  -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 or not to generate class files for implicitly refere
files
  -J<flag>             Pass <flag> directly to the runtime system
  --limit-modules <module>(<module>)*
                        Limit the universe of observable modules
  --module <module-name>, -m <module-name>
                        Compile only the specified module, check timestamps
  --module-path <path>, -p <path>
```

W kolejnym kroku pobrałam z Internetu program Notepad++ ułatwiający edycję kodu i napisałam w nim program wypisujący na ekran tekst „Hello Word” oraz wykorzystałam Notepad++ do wprowadzenia zmian w napisanym kodzie. Plik zapisałam z rozszerzeniem .java.

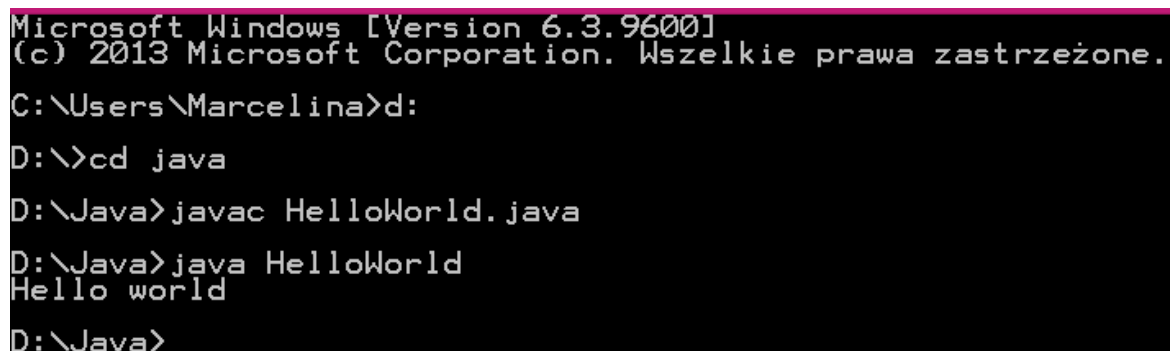


```
*D:\Java\HelloWorld.java - Notepad++
Plik Edycja Szukaj Widok Format Składnia Ustawienia Tools Makra Uruchom Wtyczki Okno ?
HelloWorld.java
1
2
3
4 public class HelloWorld // klasa HelloWorld
5 {
6     public static void main(String[] args) // statyczna metoda main
7     {
8         System.out.println("Hello world"); // wyświetlanie zawartosci na konsole
9     }
10 }
11
12
length: 232 lines: 12 Ln: 12 Col: 1 Sel: 0|0 Windows (CR LF) UTF-8 INS
```

A screenshot of the Notepad++ text editor window. The title bar reads '*D:\Java\HelloWorld.java - Notepad++'. The menu bar includes 'Plik', 'Edycja', 'Szukaj', 'Widok', 'Format', 'Składnia', 'Ustawienia', 'Tools', 'Makra', 'Uruchom', 'Wtyczki', 'Okno', and '?'. The toolbar contains various icons for file operations and editing. The editor area shows a Java source file named 'HelloWorld.java' with 12 lines of code. The code defines a public class 'HelloWorld' with a static 'main' method that prints 'Hello world - Marcelina Balamut' to the console. The status bar at the bottom indicates 'Java source file', 'length: 253 lines: 12', and the current cursor position 'Ln: 11 Col: 1 Sel: 0|0'. It also shows 'Windows (CR LF)' line endings, 'UTF-8' encoding, and 'INS' input mode.

```
1
2
3
4 public class HelloWorld // klasa HelloWorld
5 {
6     public static void main(String[] args) // statyczna metoda main
7     {
8         System.out.println("Hello world - Marcelina Balamut"); // wyświetlanie zawartosci na konsole
9     }
10 }
11
12
```

Następnie po otwarciu wiersza poleceń uruchomiłam wcześniej napisany program z użyciem maszyny wirtualnej. W tym celu pierwszym krokiem było przejście do folderu w którym został zapisany plik HelloWorld, następnie wpisanie komendy `javac HelloWorld.java` kompilującej kod i uruchomienie programu poleceniem `java HelloWorld`.

A screenshot of a Windows command prompt window. The title bar says 'Microsoft Windows [Version 6.3.9600]'. The text shows the user navigating to the 'D:\Java' directory and running the 'javac' and 'java' commands. The output of the 'java' command is 'Hello world'.

```
Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. Wszelkie prawa zastrzeżone.

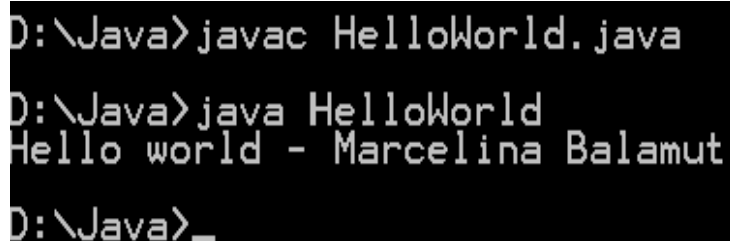
C:\Users\Marcelina>d:

D:\>cd java

D:\Java>javac HelloWorld.java

D:\Java>java HelloWorld
Hello world

D:\Java>
```

A second screenshot of a Windows command prompt window, showing the same sequence of commands as the previous one, but with a different output for the 'java' command: 'Hello world - Marcelina Balamut'.

```
D:\Java>javac HelloWorld.java

D:\Java>java HelloWorld
Hello world - Marcelina Balamut

D:\Java>_
```