Міністерство освіти і науки, молоді та спорту України

Національний університет «Львівська політехніка»

Інститут комп’ютерних наук та інформаційних технологій

Кафедра автоматизованих систем управління



**Звіт**

**до лабораторної роботи №1**

**Виконав:**

Ст. гр. КН-203 Павлишин  Д. А.

**Перевірив:**

Ст. викладач Ковальчук А.М.

**Завдання**

Exercise 1: (2) Create a class containing an int and a char that are not initialized, and print their values to verify that Java performs default initialization.

Exercise 2: (1) Following the HelloDate.java example in this chapter, create a “hello, world” program that simply displays that statement. You need only a single method in your class (the “main” one that gets executed when the program starts). Remember to make it static and to include the argument list, even though you don’t use the argument list. Compile the program with javac and run it using java. If you are using a different development environment than the JDK, learn how to compile and run programs in that environment.

Exercise 3: (1) Find the code fragments involving ATypeName and turn them into a program that compiles and runs.

Exercise 4: (1) Turn the DataOnly code fragments into a program that compiles and runs.

Exercise 5: (1) Modify the previous exercise so that the values of the data in DataOnly are assigned to and printed in main( ).

Exercise 6: (2) Write a program that includes and calls the storage( ) method defined as a code fragment in this chapter.

Exercise 7: (1) Turn the Incrementable code fragments into a working program.

Exercise 8: (3) Write a program that demonstrates that, no matter how many objects you create of a particular class, there is only one instance of a particular static field in that class.

Exercise 9: (2) Write a program that demonstrates that autoboxing works for all the primitive types and their wrappers.

Exercise 10: (2) Write a program that prints three arguments taken from the command line. To do this, you’ll need to index into the command-line array of Strings.

Exercise 11: (1) Turn the AllTheColorsOfTheRainbow example into a program that compiles and runs. Exercise 12: (2) Find the code for the second version of HelloDate.java, which is the simple comment documentation example. Execute Javadoc on the file and view the results with your Web browser. Exercise 13: (1) Run Documentation1.java, Documentation2.java, and Documentation3.java through Javadoc. Verify the resulting documentation with your Web browser.

Exercise 14: (1) Add an HTML list of items to the documentation in the previous exercise.

Exercise 15: (1) Take the program in Exercise 2 and add comment documentation to it. Extract this comment documentation into an HTML file using Javadoc and view it with your Web browser.

Exercise 16: (1) In the Initialization & Cleanup chapter, locate the Overloading.java example and add Javadoc documentation. Extract this comment documentation into an HTML file using Javadoc and view it with your Web browser.

**Код програми:**

**LabOne.java:**

package com.pavlyshyn;

public class LabOne {

public static void main(String[] args) {

//Task1

IntChar objIntChar = new IntChar();

System.out.println("Int:" + objIntChar.sampleInt + "\nChar:" + objIntChar.sampleChar);

//Task2

HelloWorld helloWorld = new HelloWorld();

helloWorld.sayHelloWorld();

//Task3

ATypeName a = new ATypeName();

//Task4

DataOnly data = new DataOnly();

//Task5

data.i = 1;

data.d = 1.5;

data.b = true;

System.out.println("data.i: " + data.i + "\ndata.d: " + data.d + "\ndata.b: " + data.b);

//Task6

LabOne labOne = new LabOne();

System.out.println(labOne.storage("Hello"));

//Task7

Incrementable incrementable = new Incrementable();

incrementable.increment();

Incrementable.increment();

System.out.println(StaticTest.i);

//Task8

Incrementable incrementable1 = new Incrementable();

incrementable1.increment();

Incrementable incrementable2 = new Incrementable();

incrementable2.increment();

Incrementable incrementable3 = new Incrementable();

incrementable3.increment();

System.out.println(StaticTest.i);

//Task9

boolean b = true;

Boolean B;

B = b;

System.out.println(B);

char ch = 'c';

Character Ch;

Ch = ch;

System.out.println(Ch);

byte by = 3;

Byte By;

By = by;

System.out.println(By);

short sh = 6;

Short Sh;

Sh = sh;

System.out.println(Sh);

int i = 1;

Integer I;

I = i;

System.out.println(I);

long l = 9;

Long L;

L = l;

System.out.println(L);

float f = (float) 1.1;

Float F;

F = f;

System.out.println(F);

double d = 3.4;

Double D;

D = d;

System.out.println(D);

//Task 10

for (int index = 0; index < 3; index++) {

System.out.println("agrgs[" + index + "] = " + args[index]);

}

//Task 11

AllTheColorsOfTheRainbow allTheColorsOfTheRainbow = new AllTheColorsOfTheRainbow();

allTheColorsOfTheRainbow.ChangeTheHueOfTheColor(2);

System.out.println(allTheColorsOfTheRainbow.anIntegerRepresentingColors);

//Task12

HelloDate helloDate = new HelloDate();

helloDate.runHelloDate();

//Task 16

Overloading overloading = new Overloading();

overloading.runOverloading();

}

int storage(String s) {

return s.length() \* 2;

}

}

**IntChar.Java:**

package com.pavlyshyn;

public class IntChar {

int sampleInt;

char sampleChar;

}

**AllTheColorsOfTheRainbow.java:**

package com.pavlyshyn;

public class AllTheColorsOfTheRainbow {

int anIntegerRepresentingColors;

void ChangeTheHueOfTheColor(int newHue) {

anIntegerRepresentingColors = newHue;

}

}

**TypeName.java:**

package com.pavlyshyn;

public class ATypeName {

}

**DataOnly.java:**

package com.pavlyshyn;

public class DataOnly {

int i;

double d;

boolean b;

}

**Documentation1.java:**

package com.pavlyshyn;

/\*\* A class comment \*/

public class Documentation1 {

/\*\* A field comment \*/

public int i;

/\*\* A method comment \*/

public void f() {}

} ///:~

**Documentation2.java:**

package com.pavlyshyn;

/\*\*

\* <pre>

\* System.out.println(new Date());

\* </pre>

\* <ol>

\* <li>List of items 1</li>

\* <li>List of items 2</li>

\* <li>List of items 3</li>

\* </ol>

\*/

public class Documentation2 {

}///:~

**Documentation3.java:**

package com.pavlyshyn;

/\*\*

\* You can <em>even</em> insert a list:

\* <ol>

\* <li> Item one</li>

\* <li> Item two</li>

\* <li> Item three</li>

\* </ol>

\*/

public class Documentation3 {

}

///:~

package com.pavlyshyn;

**HelloDate.java:**

//: object/HelloDate.java

import java.util.\*;

/\*\*

\* The first Thinking in Java example program.

\* Displays a string and today’s date.

\*

\* @author Bruce Eckel

\* @author www.MindView.net

\* @version 4.0

\*/

public class HelloDate {

/\*\*

\* Entry point to class & application.

\*

\* @param args No arguments

\* @throws exceptions No exceptions thrown

\*/

public static void runHelloDate() {

System.out.println("Hello, it\'s: ");

System.out.println(new Date());

}

} /\* Output: (55% match)

Hello, it’s:

Wed Oct 05 14:39:36 MDT 2005

\*///:~

**HelloWorld.java:**

package com.pavlyshyn;

//:HelloWorld.java

/\*\*

\* Display's message: 'Hello World'

\* @author Dmytro Pavlyshyn

\* @version 1.0

\*/

public class HelloWorld {

/\*\*

\* only method of class.

\* @param args array of string arguments (optional)

\* @throws exceptions No exceptions thrown

\*/

void sayHelloWorld() {

System.out.println("Hello World");

}

}/\* Output: (100% match)

HelloWorld

\*///:~

**Incrementable.java:**

package com.pavlyshyn;

class StaticTest {

public static int i = 47;

}

public class Incrementable {

static void increment() {

StaticTest.i++;

}

}

**IntChar.java:**

package com.pavlyshyn;

public class IntChar {

int sampleInt;

char sampleChar;

}

**Overloading.java:**

//: initialization/Overloading.java

// Demonstration of both constructor

// and ordinary method over

package com.pavlyshyn;

//:Overloading.java

/\*\*

\* Create tree that have one field height and two methods

\* @author Dmytro Pavlyshyn

\* @version 1.0

\*

\* \*/

class Tree {

int height;

/\*\*

\* Constructor by default

\* @param args No arguments

\* @throws exceptions No exceptions thrown

\*/

Tree() {

System.out.println("Planting a seedling");

height = 0;

}

/\*\*

\* Constructor

\* @param args int

\* @throws exceptions No exceptions thrown

\*/

Tree(int initialHeight) {

height = initialHeight;

System.out.println("Creating new Tree that is " + height + " feet tall");

}

/\*\*

\* Displays information about height of tree

\* @param args No arguments

\* @throws exceptions No exceptions thrown

\*/

void info() {

System.out.println("Tree is " + height + " feet tall");

}

/\*\*

\* Displays information about height of tree

\* @param args No arguments

\* @throws exceptions String

\*/

void info(String s)

{

System.out.println(s + ": Tree is " + height + " feet tall");

}

}

/\*\*

\* Create 5 objects of class Tree and display their height

\* @author Dmytro Pavlyshyn

\* @version 1.0

\*

\* \*/

public class Overloading {

/\*\*

\* Entry point to class & application.

\*

\* @param args No arguments

\* @throws exceptions No exceptions thrown

\*/

public static void runOverloading() {

for (int i = 0; i < 5; i++) {

Tree t = new Tree(i);

t.info();

t.info("overloaded method");

}

// Overloaded constructor:

new Tree();

}

} /\* Output:

Creating new Tree that is 0 feet tall

Tree is 0 feet tall

overloaded method: Tree is 0 feet tall

Creating new Tree that is 1 feet tall

Tree is 1 feet tall

overloaded method: Tree is 1 feet tall

Creating new Tree that is 2 feet tall

Tree is 2 feet tall

overloaded method: Tree is 2 feet tall

Creating new Tree that is 3 feet tall

Tree is 3 feet tall

overloaded method: Tree is 3 feet tall

Creating new Tree that is 4 feet tall

Tree is 4 feet tall

overloaded method: Tree is 4 feet tall

Planting a seedling

\*///:~

