

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ  
НАЦІОНАЛЬНОМУ УНІВЕРСИТЕТУ "ЛЬВІВСЬКА ПОЛІТЕХНІКА"

Кафедра систем штучного інтелекту

**Лабораторна робота №2**  
з дисципліни  
«Об'єктно-орієнтоване програмування»

**Виконав:**

студент групи КН-108

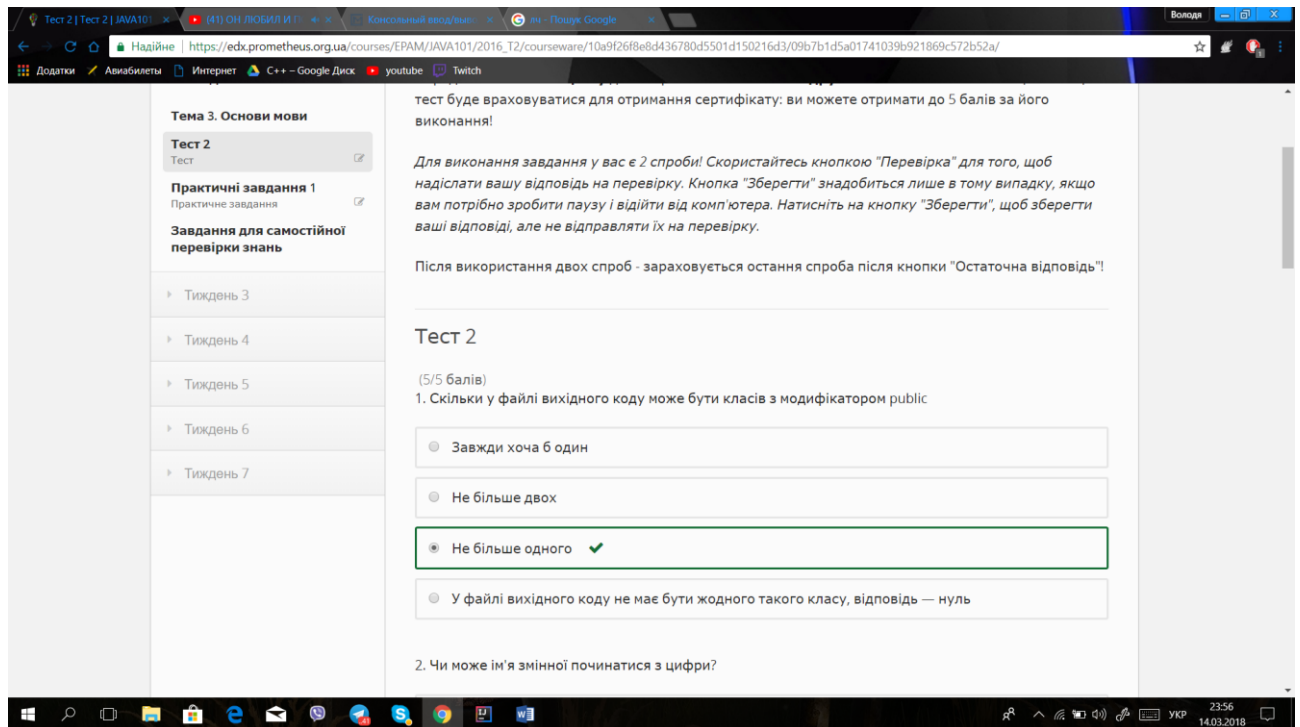
Шалавило Володимир

**Викладач:**

Гасько Р.Т.

Львів – 2018 р.

Уважно передивившись відео другого тижня на платформі Prometheus я пройшов Тест 2.



Після чого я виконав Практичні завдання 1. Написавши всі коди у IntelliJ , я закинув їх на перевірку на платформі Prometheus .

# 1. Корені рівняння

The image displays two screenshots of the IntelliJ IDEA IDE, showing the development of a Java program to find the roots of a quadratic equation.

**Top Screenshot:** The IDE window shows the file `SquareRoot.java` with the following code:

```
import java.util.Scanner;

public class SquareRoot {

    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);

        double a = in.nextDouble();
        double b = in.nextDouble();
        double c = in.nextDouble();

        double Des;
        Des = b * b - 4 * a * c;

        double D = Math.sqrt(Des);
        if ((a == 0) & (b != 0)) {
            if (c == 0) {
                double x1 = 0.0;
                System.out.println("x1=" + x1);
                System.out.println("x2=" + x1);
                return;
            } else if (c != 0) {
                double x1 = -(c / b);
                System.out.println("x1=" + x1);
                System.out.println("x2=" + x1);
                return;
            }
        }
        return;
    }
}
```

The code calculates the discriminant  $D = b^2 - 4ac$  and uses `Math.sqrt(Des)` to find the square root of the discriminant. The program handles the case where  $a = 0$  and  $b \neq 0$ , printing the root  $x_1 = -c/b$  twice.

**Bottom Screenshot:** The IDE window shows the same file `SquareRoot.java` with the following code:

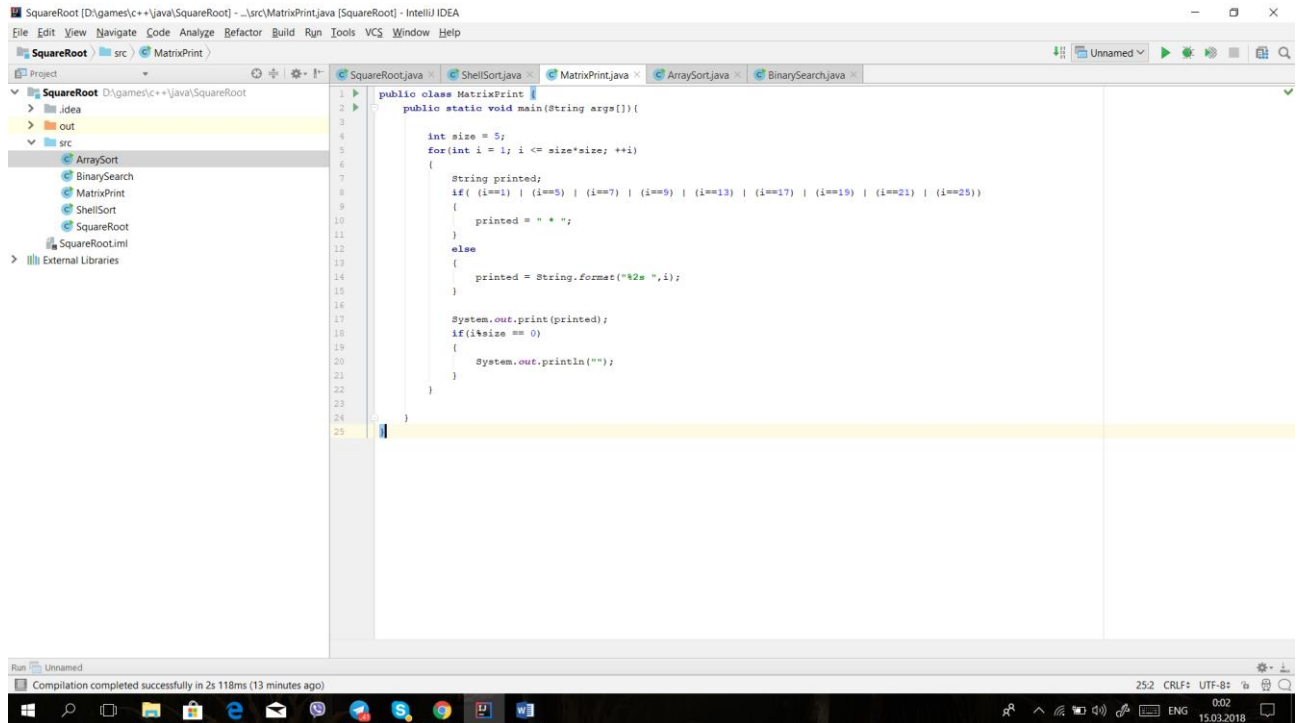
```
double D = Math.sqrt(Des);
if ((a == 0) & (b != 0)) {
    if (c == 0) {
        double x1 = 0.0;
        System.out.println("x1=" + x1);
        System.out.println("x2=" + x1);
        return;
    } else if (c != 0) {
        double x1 = -(c / b);
        System.out.println("x1=" + x1);
        System.out.println("x2=" + x1);
        return;
    }
    return;
}

if (Des >= 0) {
    double x1 = (-b + D) / (2 * a);
    double x2 = (-b - D) / (2 * a);
    System.out.println("x1 = " + x1);
    System.out.println("x2 = " + x2);
} else {
    System.out.println("x1 = ");
    System.out.println("x2 = ");
    return;
}

if ((a == 0) & (b == 0) & (c == 0)) {
    System.out.println("x1 = ");
    System.out.println("x2 = ");
    return;
}
}
```

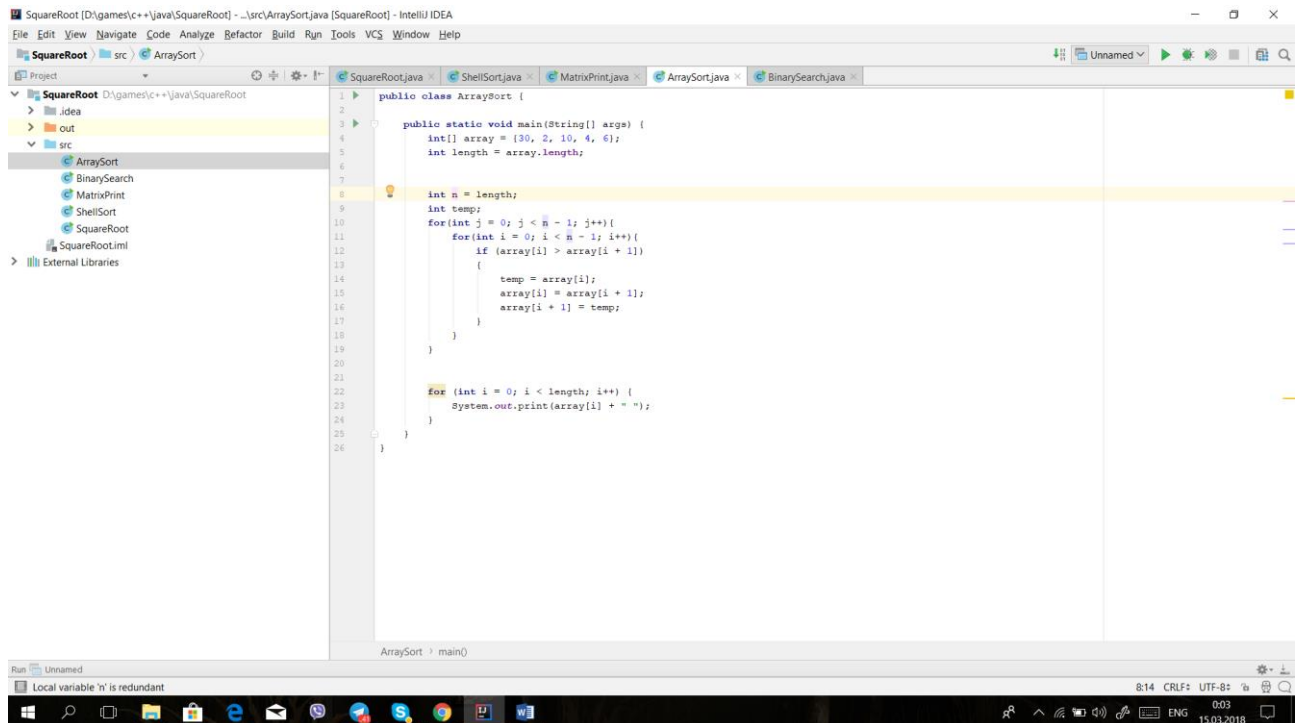
The code calculates the discriminant  $D = b^2 - 4ac$  and uses `Math.sqrt(Des)` to find the square root of the discriminant. The program handles the case where  $a = 0$  and  $b \neq 0$ , printing the root  $x_1 = -c/b$  twice. It also handles the case where  $a \neq 0$  and  $D \geq 0$ , printing the two roots  $x_1 = \frac{-b + D}{2a}$  and  $x_2 = \frac{-b - D}{2a}$ . If  $D < 0$ , it prints empty strings for  $x_1$  and  $x_2$ . Finally, it handles the case where  $a = b = c = 0$ , printing empty strings for  $x_1$  and  $x_2$ .

## 2.MatrixPrint



```
1 public class MatrixPrint {
2     public static void main(String args[]) {
3
4         int size = 5;
5         for(int i = 1; i <= size*size; ++i)
6         {
7             String printed;
8             if( (i==1) | (i==5) | (i==7) | (i==9) | (i==13) | (i==17) | (i==19) | (i==21) | (i==25))
9             {
10                 printed = " 1 ";
11             }
12             else
13             {
14                 printed = String.format("%2d ", i);
15             }
16
17             System.out.print(printed);
18             if(i%size == 0)
19             {
20                 System.out.println("");
21             }
22         }
23     }
24 }
25
```

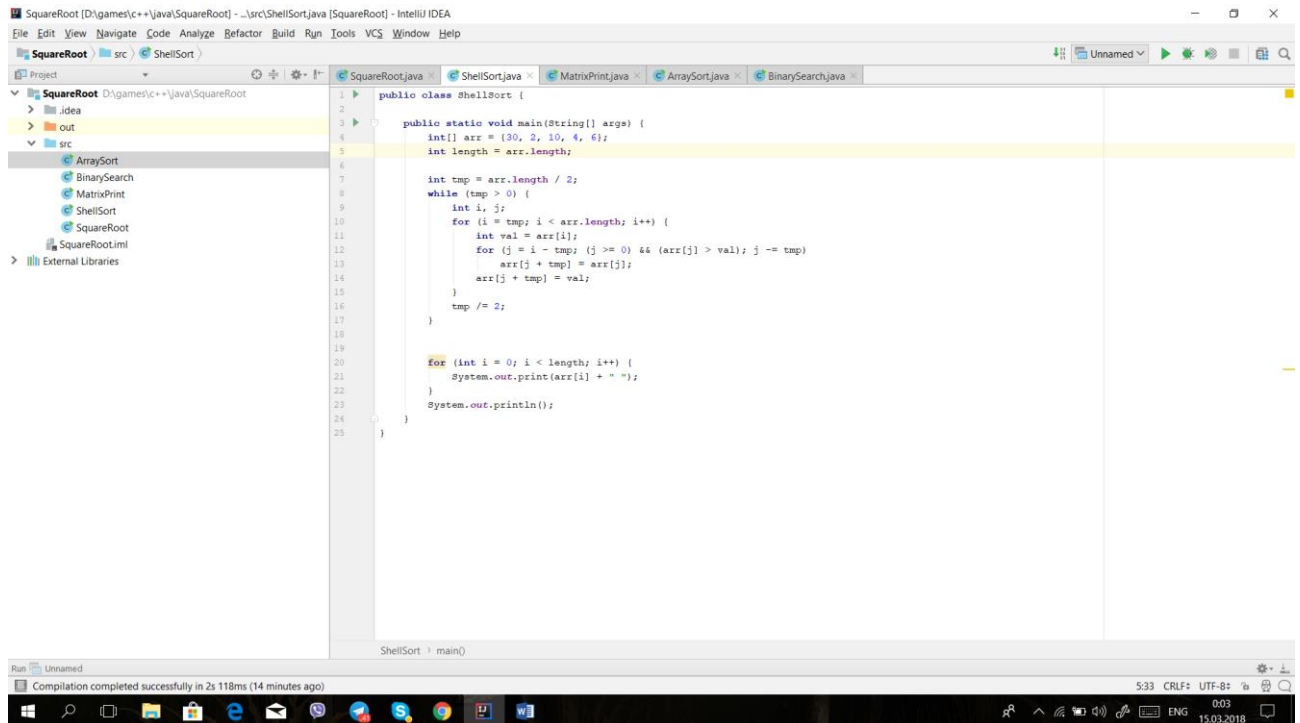
## 3.Bubble sort



```
1 public class ArraySort {
2
3     public static void main(String[] args) {
4         int[] array = {30, 2, 10, 4, 6};
5         int length = array.length;
6
7         int n = length;
8         int temp;
9         for(int j = 0; j < n - 1; j++){
10             for(int i = 0; i < n - 1; i++){
11                 if (array[i] > array[i + 1])
12                 {
13                     temp = array[i];
14                     array[i] = array[i + 1];
15                     array[i + 1] = temp;
16                 }
17             }
18         }
19
20         for (int i = 0; i < length; i++) {
21             System.out.print(array[i] + " ");
22         }
23     }
24 }
25
```

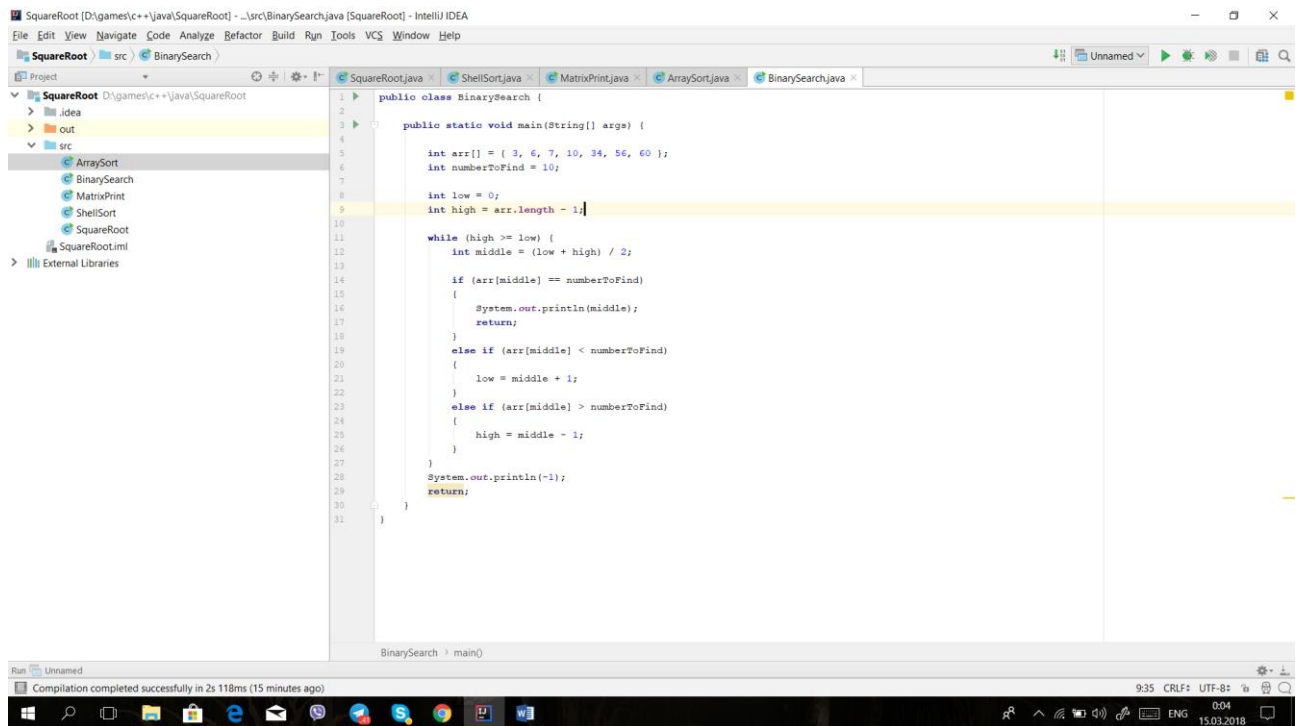
Run: Unnamed  
Local variable 'n' is redundant

## 4.Shell sort



```
1 public class ShellSort {
2
3     public static void main(String[] args) {
4         int[] arr = {30, 2, 10, 4, 6};
5         int length = arr.length;
6
7         int tmp = arr.length / 2;
8         while (tmp > 0) {
9             for (int i = tmp; i < arr.length; i++) {
10                 int val = arr[i];
11                 for (int j = i - tmp; j >= 0 && (arr[j] > val); j -= tmp) {
12                     arr[j + tmp] = arr[j];
13                     arr[j + tmp] = val;
14                 }
15                 tmp /= 2;
16             }
17         }
18
19         for (int i = 0; i < length; i++) {
20             System.out.print(arr[i] + " ");
21         }
22         System.out.println();
23     }
24 }
25 }
```

## 5.Binary Search



```
1 public class BinarySearch {
2
3     public static void main(String[] args) {
4         int arr[] = { 3, 6, 7, 10, 34, 56, 60 };
5         int numberToFind = 10;
6
7         int low = 0;
8         int high = arr.length - 1;
9
10        while (high >= low) {
11            int middle = (low + high) / 2;
12
13            if (arr[middle] == numberToFind) {
14                System.out.println(middle);
15                return;
16            }
17            else if (arr[middle] < numberToFind) {
18                low = middle + 1;
19            }
20            else if (arr[middle] > numberToFind) {
21                high = middle - 1;
22            }
23        }
24        System.out.println(-1);
25        return;
26    }
27 }
28 }
```

Всі коди доступні на [GitHub](#).