SHEET 3 SOLUTIONS

SHEHAB MAHMOUD SALAH | 2100320

GENERAL NOTE: to **copy** code **from** this PDF document, **copy each block of code** separately to **not** lose the code's formatting, alternatively you can find all the **source code** to **PROGRAMMING EXERCISES** on my **GitHub**: https://bit.ly/CSE231Sheets , **happy compiling!**

1.

(**Vector.java** on GitHub)

```java
import java.util.Scanner;
public class Vector {
    double x;
    double y;

    // Default constructor
    public Vector(){
        x = 0;
        y = 0;
    }
    // Parametrized constructor
    public Vector(double x, double y){
        this.x = x;
        this.y = y;
    }
    // x & y getters
    public double getX(){
        return x;
    }
    public double getY(){
        return y;
    }
    // x & y setters
    public void setX(double x){
        this.x = x;
    }
    public void setY(double y){
        this.y = y;
    }
    // Magnitude method
    double magnitude(){
        return Math.sqrt((Math.pow(x,2) + Math.pow(y,2)));
    }
    // Angle method
    double angle(){
        return Math.toDegrees(Math.atan(y / x));
    }
    // printing method
    void print(){
        System.out.println("Vector in cartesian form: " + x + "i + " + y + "j");
        System.out.println("Vector in polar form: " + magnitude() + "[" + angle()
+ "]");
    }

    // toString method ->> will be called in Line2D.java
    @Override
    public String toString() {
        return "Vector: (" + x + ", " + y + ")";
    }
    // Vector STATIC addition method
    static Vector add(Vector v1, Vector v2){
        double newX = v1.x + v2.x;
        double newY = v1.y + v2.y;
        return new Vector(newX, newY);
    }
    // Vector STATIC subtraction method
    static Vector sub(Vector v1, Vector v2){
        double newX = v1.x - v2.x;
        double newY = v1.y - v2.y;
        return new Vector(newX,newY);
    }
    // Read method
    void read(double x,double y){
        this.x = x;
        this.y = y;
    }

    /** Main method */
    public static void main(String[] args){
        // test cases
        Vector myVector = new Vector(4,5); // vector declaration
        double xValue = myVector.getX(); // x value getter
        double yValue = myVector.getY(); // y value getter
        System.out.println("x: " + xValue + " y: " + yValue); // print x & y
values
        myVector.print(); // print method
        myVector.setX(3); // x value setter
        myVector.setY(2); // y value setter
        System.out.println("new x: " + myVector.getX() + " new y: " +
myVector.getY()); // print new x & y values
        myVector.print(); // print method
        double magValue = myVector.magnitude(); // magnitude method
        double angleValue = myVector.angle(); // angle method
        System.out.println("magnitude = " + magValue + " angle = " + angleValue);
// print magnitude & angle
        Vector myVector2 = new Vector(1,1); // new vector declaration
        Vector summedVector = add(myVector, myVector2); // STATIC vector addition
method
        summedVector.print(); // print method
        Vector subbedVector = sub(myVector, myVector2); // STATIC vector
subtraction method
        subbedVector.print(); // print method
        myVector2.read(5,9); // read method
        myVector2.print();

        System.out.println("-------------------------------------------------");
        // test for a user input:
        Scanner input = new Scanner(System.in);
        System.out.println("Enter x value: ");
        double x = input.nextDouble();
        System.out.println("Enter y value: ");
        double y = input.nextDouble();
        input.close();
        Vector userVector = new Vector(x,y);
        userVector.print();
    }
}
```

2.

(**Line2D.java** on GitHub)

```java
import javafx.geometry.Point2D;

public class Line2D {
    private Point2D position;
    private Vector direction;

    // Constructor with Point2D and Vector objects
    public Line2D(Point2D position, Vector direction) {
        this.position = position;
        this.direction = direction;
    }

    // Constructor with coordinates for position and direction
    public Line2D(double xPos, double yPos, double xDir, double yDir) {
        this.position = new Point2D(xPos, yPos);
        this.direction = new Vector(xDir, yDir);
    }

    // Print method
    public void print() {
        System.out.println("Line: Position = " + position + ", Direction =
" + direction.toString());
    }

    // Main method for testing
    public static void main(String[] args) {
        // Create lines using different constructors
        Line2D line1 = new Line2D(new Point2D(2, 3), new Vector(1, 2));
        Line2D line2 = new Line2D(5, 6, 3, 4);

        // Print line information
        line1.print();
        line2.print();
    }
}
```

3. (MyStringBuffer.java on GitHub)

```
public class MyStringBuffer {
    private char[] data; // character array to store the string

    // Default constructor
    public MyStringBuffer(String str){
        data = str.toCharArray(); // initialize with passed string
    }

    // Print method -> prints current string
    public void print(){
        System.out.println(data);
    }
    // indexOf method -> Returns the index of the specified character
in the current object or -1 if not found.
    public int indexOf(char c){
        for (int i = 0; i < data.length; i++) {
            if (data[i] == c) {
                return i;
            }
        }
        return -1; // Not found
    }
    // length method -> Returns the length of the string
    public int length() {
        return data.length;
    }
    // erase method -> Modifies the current object by erasing a
substring whose position and length are specified by the passed
values.
    public void erase(int index, int length) {
        if (index < 0 || index >= data.length || length < 0) {
            throw new IndexOutOfBoundsException();
        }
        char[] newData = new char[data.length - length]; // Create new
array with length - length
        System.arraycopy(data, 0, newData, 0, index); // Copy data
before index
        System.arraycopy(data, index + length, newData, index,
data.length - index - length); // Copy data after index + length
        data = newData; // Set data to new array
    }
    // insert method -> Modifies the current object by inserting the
specified string at the specified position.
    public void insert(int index, String str) {
        if (index < 0 || index > data.length) {
            throw new IndexOutOfBoundsException();
        }
        char[] newData = new char[data.length + str.length()]; //
Create new array with length + str.length
        System.arraycopy(data, 0, newData, 0, index); // Copy data
before index
        System.arraycopy(str.toCharArray(), 0, newData, index,
str.length()); // Copy str at index position in newData array
```

```
        System.arraycopy(data, index, newData, index + str.length(),
data.length - index); // Copy data after index
        data = newData;
    }

    // Example usage
    public static void main(String[] args) {
        MyStringBuffer buffer = new MyStringBuffer("Hello World");

        // Testing print method
        buffer.print();  // Expected: Hello World

        // Testing indexOf method
        int index_of_o = buffer.indexOf('o');  // Expected: 4

        // Testing length method
        int length = buffer.length();  // Expected: 11

        // Testing erase method
        buffer.erase(5, 6);  // Should remove " World"
        buffer.print();  // Expected: Hello

        // Testing insert method
        buffer.insert(5, ", Universe!");
        buffer.print();  // Expected: Hello, Universe!

        // Displaying indexOf and length results
        System.out.println("Index of 'o': " + index_of_o + ", Length:
" + length);
}

}

// Congrats, we just re-invented the wheel :|
```

4. (**MyString.java** on GitHub)

```
public class MyString {
    private final char[] data; // Character array to store the string
(final for immutability)

    // Constructor
    public MyString(String str) {
        data = str.toCharArray();
    }

    // Print the string
    public void print() {
        System.out.println(data);
    }

    // Find the index of a character
    public int indexOf(char c) {
        for (int i = 0; i < data.length; i++) {
            if (data[i] == c) {
                return i;
            }
        }
        return -1; // Not found
    }

    // Get the length of the string
    public int length() {
        return data.length;
    }

    // Extract a substring (returns a new MyString object)
    public MyString substring(int beginIndex, int endIndex) {
        if (beginIndex < 0 || endIndex > data.length || beginIndex >
endIndex) {
            throw new IndexOutOfBoundsException();
        }
        return new MyString(new String(data, beginIndex, endIndex -
beginIndex));
    }

    // Example usage
    public static void main(String[] args) {
        MyString str = new MyString("Hello, World!");

        str.print(); // Output: Hello, World!

        int index = str.indexOf('o');
        System.out.println("Index of 'o': " + index); // Output: 4

        MyString subStr = str.substring(7, 13); // Extract "World"
        subStr.print(); // Output: World
    }
}
```

This concludes Sheet (3) Solutions, this document + source code to all programming exercises available on https://bit.ly/CSE231Sheets .