**Assignment # 5 – Class Node**

**Sai Kiran Reddy Gokula**

**Output:**

Given Default Constructor as Node 1

Parameter Constructor as Node 2 with X,Y values [5,40]

Copied Node 3 with Node 2 as copy constructor

Setting and getting X,Y Coordinates for node 1 from the user

A screenshot of a computer

Description automatically generated

**Console Output:**

Default Constructor Node 1: (0, 0)

Node 2 Parameter Constructor: (5, 40)

Copy Constructor: Node 3 (copy of Node 2): (5, 40)

\*\*\*Testing of Setter & Getter methods\*\*\*

Enter X coordinate Value in Range [-100, 100]:

**Output:**

Node 1 displayed with new coordinates X,Y

Adding Node 1 to Node 2

Similarly Adding Node 3 to Node 2

Check Node Equality for the 3 Nodes

**A screenshot of a computer

Description automatically generated**

**Console Output:**

Default Constructor Node 1: (0, 0)

Node 2 Parameter Constructor: (5, 40)

Copy Constructor: Node 3 (copy of Node 2): (5, 40)

\*\*\*Testing of Setter & Getter methods\*\*\*

Enter X coordinate Value in Range [-100, 100]:10

Enter Y coordinate Value in Range [-100, 100]:30

Node 1 X coordinate: 10

Node 1 Y coordinate: 30

Node 1 after setting new coordinates (x,y): (10, 30)

\*\*\*Adding Node 1 to Node 2\*\*\*

Node 1 after adding Node 2: (15, 70)

\*\*\*Adding Node 3 to Node 2\*\*\*

Node 3 after adding Node 2: (10, 80)

\*\*\*Nodes Equality Check\*\*\*

Node 1 and Node 2 are not equal

Node 2 and Node 3 are not equal

Node 1 and Node 3 are not equal

**Output:**

Exceptions handled for range [-100,100]

**A screenshot of a computer

Description automatically generated**

**Console Output:**

Default Constructor Node 1: (0, 0)

Node 2 Parameter Constructor: (5, 40)

Copy Constructor: Node 3 (copy of Node 2): (5, 40)

\*\*\*Testing of Setter & Getter methods\*\*\*

Enter X coordinate Value in Range [-100, 100]:-200

java.lang.IllegalArgumentException: Invalid value given for x-coordinate. Must be in the range [-100, 100]

**Output:**

Exceptions handled for range [-100,100]

**A screenshot of a computer

Description automatically generated**

**Console Output:**

Default Constructor Node 1: (0, 0)

Node 2 Parameter Constructor: (5, 40)

Copy Constructor: Node 3 (copy of Node 2): (5, 40)

\*\*\*Testing of Setter & Getter methods\*\*\*

Enter X coordinate Value in Range [-100, 100]:100

Enter Y coordinate Value in Range [-100, 100]:101

java.lang.IllegalArgumentException: Invalid value given for y-coordinate. Must be in the range [-100, 100]

**Output:**

Exceptions handled for added Node in range [-100,100]

**A screenshot of a computer

Description automatically generated**

**Console Output:**

Default Constructor Node 1: (0, 0)

Node 2 Parameter Constructor: (5, 40)

Copy Constructor: Node 3 (copy of Node 2): (5, 40)

\*\*\*Testing of Setter & Getter methods\*\*\*

Enter X coordinate Value in Range [-100, 100]:25

Enter Y coordinate Value in Range [-100, 100]:75

Node 1 X coordinate: 25

Node 1 Y coordinate: 75

Node 1 after setting new coordinates (x,y): (25, 75)

\*\*\*Adding Node 1 to Node 2\*\*\*

java.lang.IllegalArgumentException: Addition results in out of range.Coordinates must be in the range [-100, 100]

**Output:**

Equality check between the nodes

**A screenshot of a computer

Description automatically generated**

**Console Output:**

Default Constructor Node 1: (0, 0)

Node 2 Parameter Constructor: (5, 40)

Copy Constructor: Node 3 (copy of Node 2): (5, 40)

\*\*\*Testing of Setter & Getter methods\*\*\*

Enter X coordinate Value in Range [-100, 100]:5

Enter Y coordinate Value in Range [-100, 100]:40

Node 1 X coordinate: 5

Node 1 Y coordinate: 40

Node 1 after setting new coordinates (x,y): (5, 40)

\*\*\*Adding Node 1 to Node 2\*\*\*

Node 1 after adding Node 2: (10, 80)

\*\*\*Adding Node 3 to Node 2\*\*\*

Node 3 after adding Node 2: (10, 80)

\*\*\*Nodes Equality Check\*\*\*

Node 1 and Node 2 are not equal

Node 2 and Node 3 are not equal

Node 1 equals Node 3

**Source Code:**

package a5;

import java.util.Scanner;

/\*\*

\* CPSC 24500-001- Object-Oriented Programming

\* Assignment 5

\*/

public class Node {

private static final int MIN\_VALUE = -100;

private static final int MAX\_VALUE = 100;

private int x;

private int y;

//Default constructor with x and y initialized to zero.

public Node()

{

this(0, 0);

}

/\*\*

\* Copy constructor.

\* @param 'copyNode' to copy the other Node

\*/

public Node(Node copyNode)

{

this(copyNode.x, copyNode.y);

}

/\*\*

\* Constructor with given x and y coordinates.

\* @param x to set x-coordinate

\* @param y to set y-coordinate

\* @throws exception if x or y is out of range

\*/

public Node(int x, int y)

{

setX(x);

setY(y);

}

/\*\*

\* Get the x-coordinate.

\*\* @returns the x value

\*/

public int getX()

{

return x;

}

/\*\*

\* Set the x-coordinate.

\* @param x to set x-coordinate value

\* @throws IllegalArgumentException if x is out of range

\*/

public void setX(int x)

{

if (!isValidCoordinateRange(x)) {

throw new IllegalArgumentException("Invalid value given for x-coordinate. Must be in the range [-100, 100]\n");

}

this.x = x;

}

/\*\*

\* Get the y-coordinate.

\* @returns the y value

\*/

public int getY()

{

return y;

}

/\*\*

\* Set the y-coordinate.

\* @param y to set y-coordinate value

\* @throws IllegalArgumentException if y is out of range

\*/

public void setY(int y)

{

if (!isValidCoordinateRange(y)) {

throw new IllegalArgumentException("Invalid value given for y-coordinate. Must be in the range [-100, 100]\n");

}

this.y = y;

}

/\*\*

\* Add another node to this node.

\* @param 'givenNode' to add to the caller node

\* @throws IllegalArgumentException if the result is out of range

\*/

public void add(Node givenNode)

{

int addX = this.x + givenNode.x;

int addY = this.y + givenNode.y;

if (!isValidCoordinateRange(addX) || !isValidCoordinateRange(addY)) {

throw new IllegalArgumentException("Addition results in out of range.Coordinates must be in the range [-100, 100]");

}

this.x = addX;

this.y = addY;

}

/\*\*

\* Override toString method to return a string that represents the Node.

\* @return string representation of the node

\*/

@Override

public String toString() {

return "(" + x + ", " + y + ")";

}

/\*\*

\* Override equals method to check for nodes equality.

\* Two nodes are equal if they have the same values for x and y.

\* @param obj the object to compare

\* @return true if the objects are equal, false otherwise

\*/

@Override

public boolean equals(Object obj)

{

if (this == obj)

return true;

if (obj == null || !(obj instanceof Node) )

return false;

Node other = (Node) obj;

return x == other.x && y == other.y;

}

private boolean isValidCoordinateRange(int coVal) {

return coVal >= MIN\_VALUE && coVal <= MAX\_VALUE;

}

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

try {

// Create a node using default constructor

Node node1 = new Node();

System.out.println("Default Constructor Node 1: " + node1);

// Created a node using constructor with specified coordinates

Node node2 = new Node(5, 40);

System.out.println("Node 2 Parameter Constructor: " + node2);

// Create a copy of node2 using copy constructor

Node node3 = new Node(node2);

System.out.println("Copy Constructor: Node 3 (copy of Node 2): " + node3);

// Test getter and setter methods

System.out.println("\*\*\*Testing of Setter & Getter methods\*\*\*");

System.out.print("Enter X coordinate Value in Range [-100, 100]:");

int x = sc.nextInt();

node1.setX(x);

System.out.print("Enter Y coordinate Value in Range [-100, 100]:");

int y = sc.nextInt();

node1.setY(y);

System.out.println("Node 1 X coordinate: " + node1.getX());

System.out.println("Node 1 Y coordinate: " + node1.getY());

System.out.println("Node 1 after setting new coordinates (x,y): " + node1.toString());

// Test add method & String method

System.out.println("\*\*\*Adding Node 1 to Node 2\*\*\*");

node1.add(node2);

System.out.println("Node 1 after adding Node 2: " + node1.toString());

System.out.println("\*\*\*Adding Node 3 to Node 2\*\*\*");

node3.add(node2);

System.out.println("Node 3 after adding Node 2: " + node3.toString());

// Test nodes equality

System.out.println("\*\*\*Nodes Equality Check\*\*\*");

if (node1.equals(node2))

System.out.println("Node 1 equals Node 2 ");

else

System.out.println("Node 1 and Node 2 are not equal");

if (node2.equals(node3))

System.out.println("Node 2 equals Node 3 ");

else

System.out.println("Node 2 and Node 3 are not equal");

if (node1.equals(node3))

System.out.println("Node 1 equals Node 3 ");

else

System.out.println("Node 1 and Node 3 are not equal");

} catch (Exception e) {

System.out.print(e);

}

}

}