**Assignment # 6**

**Saikiran Reddy Gokula**

**Output:**

Executed the main class ‘Driver’

**A screenshot of a computer

Description automatically generated**

**Console Output:**

Menu:

1. Fill

2. Clear

3. Count Nodes

4. Count ThreeDNodes

5. Sort

6. Display

7. Exit

Enter your choice:

**Output:**

Filled with 0 nodes , tried to display the nodes and tried to sort the nodes when there are 0 nodes.

A screenshot of a computer

Description automatically generated

**Console Output:**

Menu:

1. Fill

2. Clear

3. Count Nodes

4. Count ThreeDNodes

5. Sort

6. Display

7. Exit

Enter your choice: 1

Enter the size of the list: 0

Nodes filled successfully.

Menu:

1. Fill

2. Clear

3. Count Nodes

4. Count ThreeDNodes

5. Sort

6. Display

7. Exit

Enter your choice: 5

No nodes to sort.

Menu:

1. Fill

2. Clear

3. Count Nodes

4. Count ThreeDNodes

5. Sort

6. Display

7. Exit

Enter your choice: 6

No nodes to display.

Menu:

1. Fill

2. Clear

3. Count Nodes

4. Count ThreeDNodes

5. Sort

6. Display

7. Exit

Enter your choice:

**Output:**

Filled with 5 nodes, displayed the nodes, counted the ‘Nodes’ and counted the ‘ThreeDNodes’

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

**Console Output:**

Menu:

1. Fill

2. Clear

3. Count Nodes

4. Count ThreeDNodes

5. Sort

6. Display

7. Exit

Enter your choice: 1

Enter the size of the list: 5

Nodes filled successfully.

Menu:

1. Fill

2. Clear

3. Count Nodes

4. Count ThreeDNodes

5. Sort

6. Display

7. Exit

Enter your choice: 6

Nodes:

(-78,-11)

(37,-100)

(23,70)

(-82,-36)

(0, 42, -10)

Menu:

1. Fill

2. Clear

3. Count Nodes

4. Count ThreeDNodes

5. Sort

6. Display

7. Exit

Enter your choice: 3

Number of Nodes: 5

Menu:

1. Fill

2. Clear

3. Count Nodes

4. Count ThreeDNodes

5. Sort

6. Display

7. Exit

Enter your choice: 4

Number of ThreeDNodes: 1

Menu:

1. Fill

2. Clear

3. Count Nodes

4. Count ThreeDNodes

5. Sort

6. Display

7. Exit

Enter your choice:

**Output:**

Sorted the above filled nodes and displayed all the nodes.

A screenshot of a computer

Description automatically generated

**Console Output:**

Menu:

1. Fill

2. Clear

3. Count Nodes

4. Count ThreeDNodes

5. Sort

6. Display

7. Exit

Enter your choice: 5

Nodes sorted successfully.

Menu:

1. Fill

2. Clear

3. Count Nodes

4. Count ThreeDNodes

5. Sort

6. Display

7. Exit

Enter your choice: 6

Nodes:

(-82,-36)

(-78,-11)

(37,-100)

(23,70)

(0, 42, -10)

**Output:**

Cleared the filled nodes, counted the nodes , sorted the nodes and displayed the nodes when there are 0 nodes.

Exited from the menu.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

**Console Output:**

Menu:

1. Fill

2. Clear

3. Count Nodes

4. Count ThreeDNodes

5. Sort

6. Display

7. Exit

Enter your choice: 2

Nodes cleared successfully.

Menu:

1. Fill

2. Clear

3. Count Nodes

4. Count ThreeDNodes

5. Sort

6. Display

7. Exit

Enter your choice: 6

No nodes to display.

Menu:

1. Fill

2. Clear

3. Count Nodes

4. Count ThreeDNodes

5. Sort

6. Display

7. Exit

Enter your choice: 5

No nodes to sort.

Menu:

1. Fill

2. Clear

3. Count Nodes

4. Count ThreeDNodes

5. Sort

6. Display

7. Exit

Enter your choice: 3

Number of Nodes: 0

Menu:

1. Fill

2. Clear

3. Count Nodes

4. Count ThreeDNodes

5. Sort

6. Display

7. Exit

Enter your choice: 4

Number of ThreeDNodes: 0

Menu:

1. Fill

2. Clear

3. Count Nodes

4. Count ThreeDNodes

5. Sort

6. Display

7. Exit

Enter your choice: 7

Exiting...

**Output:**

Tested the ThreeDNode file by adding the main method.

A screenshot of a computer

Description automatically generated

**Console Output:**

Default ThreeDNode: (10, 20, 20)

Custom ThreeDNode: (15, 30, 45)

Copy of ThreeDNode: (15, 30, 45)

Node2 equals Node3: true

**Source Code:**  I have pushed all the files into git.

**As per the given class diagram**, I have Noticed the following:

- Class Structure: The diagram shows the classes involved in the entire source program and their relationships.

Interfaces - The 'a5::INode' interface defines common methods for nodes. It has two implementations- 'a5::Node' and 'a6::ThreeDNode'.

'a5::INode' interface defines constants 'LOWER\_LIMIT' and 'UPPER\_LIMIT' which are used for validating node coordinates. Also, Default x, y z values.

Methods - Each class includes its method constructors, getters, setters and utility methods like 'toString', 'equals' and'`compareTo'.

Relationships:

'a5::NodeFactory' is responsible for creating instances of nodes ('Node' and 'ThreeDNode').

'a6::Nodes' contains a list of nodes and provides methods for managing and interacting with them.

'a6::Driver' the main class responsible for user interaction, as it contains the 'main' method and presents a menu for user actions.

Composition: The 'a6::Nodes' class contains an 'ArrayList' of 'INode' objects, indicating a composition relationship.

Inheritance is represented by the a5::Node and a6::ThreeDNode classes implementing the a5::INode interface. Both Node and ThreeDNode inherit common behaviors and properties from the INode interface. ThreeDNode also extends the Node class, indicating that it inherits properties and behaviors from Node.

Polymorphism is achieved through method overriding in the Node and ThreeDNode classes. Both classes implement the compareTo method from the INode interface, but they provide their own implementation specific to their behavior.

Composition is demonstrated by the a6::Nodes class containing an ArrayList of INode objects. This means that the Nodes class "has-a" collection of INode objects.

Abstraction is achieved through the a5::INode interface. The interface defines a contract for node objects, specifying common methods that all node implementations must implement.