Rajiv Gandhi Institute of Technology, Kottayam

S4 B.Tech. in CSE (2016-2020) Batch

CS 206 – Object Oriented Design and Programming

Assignments 1 & 2

Assignment 1 – Programming Assignment (Group wise)

The objective is to get familiarized with OOP concepts and JAVA. Students are required to form groups of 10. Each group will be allotted a question from the set below in a random fashion. You are required to program for the respective question and write a brief note (not exceeding 2 pages) explaining the methods and OOP features you used, and also about the output . Send me the source code and note, latest by 2nd April 2018.

Assignment 2 – Presentation

Each group has to make a 10 minute presentation on Assignment 1. I'll let you know about the date of presentation.

Assignment 1 Questions

1. Construct a class called *Sorter* and an abstract method *sort*. Inherit classes for bubble sort, selection sort, merge sort and quick sort and override the method *sort*. Use a large input (probably read from a file) and get an estimate of speed of execution of different sorting methods.

2. Create an abstract class *Stack* with operations *push* and *pop*. Inherit *ArrayStack* and *LinkedStack* from *Stack*, overriding the above operations. Use *Stack* for infix to postfix conversion.

3. Construct a class *Node*. Create another class *LinkedList* which uses *Node*. Implement various operations for *LinkedList*. Inherit *DoublyLinkedList* from *LinkedList*.

4. Construct a class for *BinaryTree*. Implement various traversals. Inherit the class *ThreadedBinaryTree*.

5. Create an abstract class *Graph* with two abstract operations *bfs* and *dfs*. Inherit two classes

from *Graph*, one using adjacency list and the other using adjacency matrix representations. Override the above methods.

6. Implement an operating system simulation using Java threads. A thread is responsible for dynamically creating and deleting tasks (in a random fashion). Maintain two linked lists, one containing *ready* tasks and the other containing *blocked* tasks. A high priority thread, the *scheduler*, acts periodically and does a round robin scheduling. Each task may be a low priority thread.

7. Create threads which are able to read text files in parallel. Arrange the words in order of frequency (pertaining to all the documents).