**CSE207 – Data Structures**

**Project: Queue Simulator**

**Project Description:**

An important application of queue is queue simulator, a modeling activity used to generate statistics about performance of queue. For this you have to build a single-server single queue simulator. For example, you have to build this simulator for a super shop. This shop has one window and a clerk can serve one customer at a time. The simulator needs to check three events; the arrival of customer, the start of customer’s processing and the completion of customer’s processing. For constructing queue simulator you have to do the following:

1. Use queue for constructing customer’s queue
2. For the server each call history is saved in a stack
3. Use separate structure for current customer status and simulation statistics.

In this project you have to do the following:

1. Print statistics gathered per day during the simulation along with the average queue wait time and average queue service time
2. Show the arrival time, start time, wait time, service time and the number of elements in the queue, each time a customer is completely served.

**CSE207 – Data Structures**

**Project: Mini Parser**

**Project Description:**

A parser is a software component that takes input data (frequently text) and builds a data structure – often some kind of parse tree, abstract syntax tree or other hierarchical structure – giving a structural representation of the input, checking for correct syntax in the process. In this project, you have to build a small parser that will parse branching statement (*if and if-else-else if*) and algebraic expression of a source program and check for correct syntax in the program.

For example consider the below code.

1. int main()

2. {

3. int a, b,c;

4. a= 3; b=2;

5. if[d>b)

6. c = (a+b)/a;

7. else

8. c = (a+b/b;

9. return 0;

10. }

Here we have found three errors. They are: **if [d>b)** in line no 5**, c = (a+b/b** in line no 8 **and d is an undefined variable** in line no 5. So, in this project you have to build a parser that will check syntactical error of a source code. To construct the **syntax parser** you may consider the following data structure,

1. A file that contains a small program containing mathematical expression and branching statements.
2. Stack for checking correct syntactic structure of given source code

In this project you have to do the following:

1. Build a parser that will parse algebraic expression
2. Parse structure of branching statement

If error found then provide an error message with line number of the source program

**CSE207 – Data Structures**

**Project: Rat in a Maze**

**Project Description:**

Rat in a maze is a popular puzzle game where a mouse start from a start point to final point by visiting different internal spots. To get to his final destination the mouse need to visit one point two or three times or need to backtrack.



A Maze is given as N\*N binary matrix of blocks where source block is the upper left most block i.e., maze [0][0] of the above figure and destination block is lower rightmost block i.e., maze[N-1][N-1]. A rat starts from source and has to reach destination. The rat can move in four directions: right, left, top and down. In the maze matrix, 0 means the block is dead end and 1 means the block can be used in the path from source to destination. To develop the game you may use the following data structure:

* 1. Matrix for the representation of the Maze
  2. Stack or graph to figure out the movement of the rat where a rate cannot move towards the dead block.

In this project you have to do the following:

1. Simulate a mouse movement through the maze
2. Show the path through the maze moved by the mouse

**CSE207 – Data Structures**

**Project: Postfix and Prefix Calculator**

**Project Description:**

You have been using one of very classic calculator, the HP-35, for a long time. It was the first handheld calculator manufactured by Hewlett Packard in 1972. However, after a disastrous accident (dropped it in a sink), it is no longer functional. You miss this calculator so much You finally decided to implement its special form of postfix calculation yourself. For constructing the postfix calculator you have to do the following:

1. Use stack for converting infix expression to postfix and prefix expression
2. Use stack for calculating value of postfix expression
3. Use queue to calculate value of prefix expression

In this project you have to do the following:

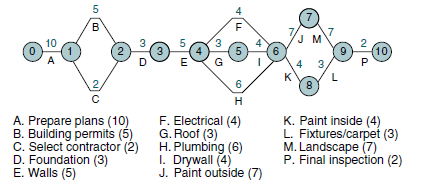
* 1. Take infix expression as input
  2. Convert postfix and prefix expression
  3. Evaluate value of postfix and prefix expression

**CSE207 – Data Structures**

**Project: CPM Manager**

**Project Description:**

One of the tools used to manage large projects is known as the critical path method (CPM). In CPM the manager builds a network of all phases of a project and then evaluates the network to determine critical aspects of the project. In a CPM network, each vertex is an event, such as the start or completion of a task. The arcs connecting the vertices represent the duration of the activity. Unlike the examples in the text, they also store the name of the activity. To better understand the concept, let’s look at a possible CPM plan to build a house. The network for this project is shown in below figure:



In the plan we see that it will take 10 days to prepare the building plan (A) and 5 days to get it approved (B). Furthermore, we can’t start building until we have selected the contractor (C).

Our plan is to construct the shortest path from the start to the end for our plan. For constructing the shortest path you have to do the following:

* 1. Build a graph for the network
  2. Build Minimum spanning tree from the graph to find shortest possible completion path

In the project you have to do the following:

1. What is the shortest possible completion time (SPCT)? The SPCT is the shortest path through the graph from beginning to end.
2. What is the earliest start time (EST) for each activity? The EST is the sum of the weights in the minimum spanning tree up to the activity.
3. What is the latest start time (LST) for each activity? The LST is the SPCT for the whole project minus the SPCT for the rest of the project
4. What is the critical path for the project? (The critical path is the subgraph consisting of the minimum spanning tree.)

**CSE207 – Data Structures**

**Project: Resolving Overbooked Flight**

**Project Description:**

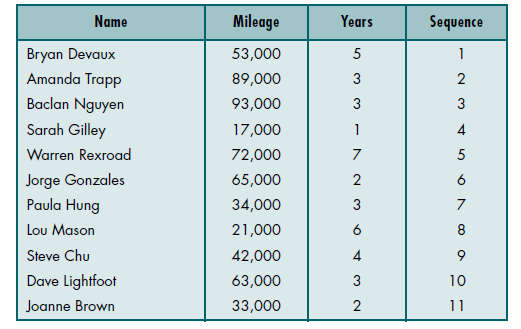
An airline company uses the formula shown below to determine the priority of passengers on the waiting list for overbooked flights. Priority number = A / 1000 + B – C where,

A is the customer’s total mileage in the past year

B is the number of years in his or her frequent flier program

C is a sequence number representing the customer’s arrival position when he or she booked the flight.

Given a file of overbooked customers as shown in below Table, write a program that reads the file and determines each customer’s priority number. You have to build an application for the airline company that builds a priority queue using the priority number and prints a list of waiting customers in priority sequence.



For constructing the application you have to do the following:

1. Build a priority queue( binary heap) based on the priority number
2. Build a stack to store information of customer who will get flight ticket later

In the project you have to do the following:

1. Builds a priority queue using the priority number

2. Prints a list of waiting customers in priority sequence.

3. Print details of customer who took flight ticket later

**CSE207 – Data Structures**

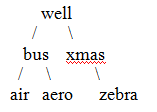
**Project:** Word Dictionary

**Project Description:**

Dictionary means a reference book listing alphabetically terms or names important to a particular subject or activity along with discussion of their meanings and applications. Dictionary should have different functionalities such as add new word, search word, delete word and find similar meaning word. For constructing dictionary you have to do the following:

1. Use Binary Search Tree to store the list of 50 words.
2. Each node of BST stores a key of a dictionary. Key 'k' of a node is always greater than the keys present in its left sub tree. Similarly, key 'k' of a node is always lesser than the keys present in its right sub tree.

**Example:**



In the above example, keys present at the left sub-tree of the root node are lesser than the key of the root node.  And also, the keys present at the right sub-tree are greater than the key of the root node.

In this project you have to do the following,

1. Add new word, search word, delete word and find similar meaning word.
2. All the data may input from a text file where each entry include a word and its definition(meaning)
3. When you search for a particular word then it will give you suggestion of nearest words.

Example: if you search for "Carn"  
Output: the word "carn" was not found,  
Did u mean "cart", " card ", " carrot", "carrom"

**CSE207 – Data Structures**

**Project: Call distributor in Call center**

**Project Description:**

A call center is a centralized office used for receiving or transmitting a large volume of requests by telephone. This project will contain multiple servers that will solve different problem over phone and one queue that will hold call of customer. Each of the servers can attend call of customer for 10 min. After finishing the call, another call is given to the free server.For constructing virtual call center you have to do the following:

1. Use linked list for constructing multiple server
2. For each server each call history is saved in a stack
3. Customer call will store in a single queue

In this project you have to do the following:

1. Forward call to different server
2. Count total call received by per server during operating system
3. Total time per customer waiting for their calls