**Data Structure and Algorithms**

**Class Assignment # 1**

**Total Marks: 30 Deadline: 05-04-24: till 11:59 PM**

**Instructor Name: Faiza Khadim**

**Instructions:**

* You are required to submit CPP files on CMS before the due date.
* Implement all data structures using techniques of object-oriented programming. Make an abstract class having pure virtual functions that will be over-ridden/implemented in the derived class. Further, use template classes (generic data types).
* All program codes should be written in C/C++. Students should use Visual studio compiler for coding.
* Plagiarism in the work is Professional dishonesty which leads to zero in grading of assignment.

**Learning Objectives: Implement various data structures and apply them in implementing simple applications [CLO-02]**

**Question 1:**

Let us discuss Rat in a MAZE as example problem that can be solved using a Stack. 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] 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: back, forward, up 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. Visited path (visited index) should be marked as -1. Note that this is a simple version of the typical Maze problem. For example, a more complex version can be with limited number of moves.

All inputs and outputs will be done using filing. Input file name will be ***input.txt*** and output will be in ***output.txt***.

Following is an example maze.

Gray blocks are dead ends (value = 0).

[](http://www.geeksforgeeks.org/wp-content/uploads/ratinmaze_filled11.png)

Following is binary matrix representation of the above maze (in file “***input.txt***”). Each row is in a new line; numbers in each row are separated by white spaces.

1 0 0 0

1 1 0 1

0 1 0 0

1 1 1 1

Following is maze with highlighted solution path.

[](http://www.geeksforgeeks.org/wp-content/uploads/ratinmaze_filled_path1.png)

Following is the solution matrix (output of program) for the above input matrix. The output will be placed in “***output.txt***”

1 0 0 0

1 1 0 0

0 1 0 0

0 1 1 1

Note: In case of multiple paths, the very first path found should be saved. If the solution does not exist, that is, you are not able to find a path, write: **PATH NOT FOUND** in the file “***output.txt***”.

**Question 2:**

In this task, you have to create abstract data type (ADT) for **Set**, which will implement the basic operations of **Set** concept in mathematics. The data members and operations needed for this ADT are given below.

**Data Members:**

* T \*data; //pointer to an array of type T which will be treated as set of given type of elements //
* int noOfElements; // number of elements in the Set
* int capcity; // maximum possible number of elements that can be stored in the Set //

**Supported functions:**

1. Set( int cap = 0 ) // default parameterized constructor
2. Set( const Set<T> & ref) // copy constructor
3. ~Set() // destructor
4. void insert (T element) // insert function which stores the element in the array
5. void remove (T element) // this function removes the element from array
6. int getCardinality() // returns number of elements in the array
7. Set<T> calcUnion ( const Set<T> & s2 ) // returns the union of s2 and calling object
8. int isMember ( T val ) // returns 1 is val is the member of the array otherwise return 0
9. void resize (int newCapacity) // resize the array to new capacity. Make sure that elements in the current array should be preserved in the new array.

**Question 3:**

Design a **stack-based algorithm** to determine whether the parentheses in a given expression are **balanced**. Write a description of the algorithm and its implementation details. Provide examples of input expressions along with the expected output to demonstrate the functionality of the algorithm.

**Sample Output:**

sample ionput: (a+b+c)

output: balanced expression.

sample input: (a+b

output: not balanced

sample input: ((a+c)+b

output: not balanced

sample input: ((a+c)+b)))

output: not balanced