CSE2216: DSA1-Lab (Spring'24)

Assignment-2 Total: 20 marks

***Replace 011XXXXXX with your student id

1. Create a cpp file and rename it with 011XXXXXX_stack.cpp
   Implement the following classes in this file
   i. **Stack:** implement stack with fixed size array
   ii. **StackD:** implement stack with dynamic size array. Initially array will have a size and the size can be modified during runtime.
   iii. **StackList:** implement stack using linked list
   The classes should have the following functions:
   a) Constructor
   b) push(int x): insert element at top
   c) pop(): return the top element and remove from stack
   d) top(): return the top element
   e) display(): display the elements
   f) destructor if needed

2. Create a cpp file and rename it with 011XXXXXX_queue.cpp
   Implement the following classes in this file
   i. **Queue:** implement queue with fixed size array
   ii. **QueueD:** implement queue with dynamic size array. Initially array will have a size and the size can be modified during runtime.
   iii. **QueueList:** implement queue using linked list

   The classes should have the following functions:
   a) Constructor
   b) enqueue(int x): insert element at rear
   c) dequeue(): return the front element and remove fromqueue
   d) display(): print the elements in the queue destructor if needed

3. Create a cpp file and rename it with 011XXXXXX_graph.cpp
   Implement the following classes in this file

   i. GraphMatrix: Implement graph using adjacency matrix

   ii. GraphList: Implement graph using adjacency list

Both classes should have the following functions:

a) Constructor
b) void addEdge(int u,int v)
c) bool isEdge(int u,int v)
d) void displayGraph()
e) int inDegree(int u)
f) int outDegree(int u)
g) destructor
h) bfs(int src)
i) dfs()
j) distance(int src,int dest)
k) path(int src,int dest)

4. Create a cpp file and rename it with 011XXXXX_tree.cpp.
Implement the following methods:
    a) insert(int x)
    b) delete(int x)
    c) search(int x)
    d) preoder, inorder and postorder traversal

5. **The Maze of Mysteries - Finding Paths with Treasures (Mark 10)**

Create a cpp file and rename it with 011XXXX_maze.cpp

You find yourself in a mysterious maze filled with twists and turns, dead ends, and hidden treasures. Your task is to navigate through the maze to find paths from the starting position to the exit that contain hidden treasures. **The maze is represented as a 2D grid** where each cell can have the following values:

0: Pathway.

1: Wall.

2: Exit.

3: Treasure.

You have to take the maze size as input as well as the above values for that maze cell.

Write a function **find_paths_with_treasures(maze, start_row, start_col)** that takes the maze grid and your starting position as inputs. **The function should print all the paths from the starting**

position to the exit (2) that contain treasures (3). If no such path is possible, the function should print "No path with treasure found."

Here is a sample maze with a size of 5x5:

```
maze = [
    [0, 1, 2, 0, 0],
    [0, 0, 0, 1, 0],
    [0, 3, 1, 0, 0],
    [0, 1, 0, 1, 0],
    [0, 0, 0, 0, 0]
]
```

For the given maze and starting position (0, 0), the function should print the following path:

Path with treasure: **(0,0)**->(1,0)->(2,0)->**(2,1)**->(1,1)->(1,2)->**(0,2)**

In this case, the path starts from the treasure at position (0, 0), follows the pathway, and ends at the exit (2) while collecting the treasure on the way.

Your function should implement either BFS or DFS to explore the maze, avoid walls (1), and identify paths containing treasures. If no such path is found, the function should print "No path with treasure found."

Here is a sample maze with a size of 5x5:

```
maze = [
    [0, 1, 0, 0, 2],
    [0, 0, 0, 1, 0],
    [0, 3, 1, 0, 0],
    [0, 1, 0, 1, 0],
    [0, 0, 0, 0, 0]
]
```

For this maze and starting position (1, 0), the function should print:

Path with treasure: **(1,0)**->(2,0)->**(2,1)**->(1,1)->(1,2)->(0,2)->(0,3)->**(0,4)** **or,**

Path with treasure: **(1,0)**->(1,1)->**(2,1)**->(2,0)->(3,0)->(4,0)->(4,1) ->(4,2)->(4,3)->(4,4)->(3,4)->(2,4)->(1,4)->**(0,4)**

Submission guideline:
Put all the cpp files into a folder and zip it. Then rename the zip file with your student id. Submit the zip file.
Submission deadline: 11 PM 29 April,2024