Data Structures and Algorithms Lab

Lab 03 **Marks 05**

Instructions

Work in this lab individually. You can use your books, notes, handouts etc. but you are not allowed to borrow anything from your peer student.

Marking Criteria

Show your work to the instructor before leaving the lab to get some or full credit.

What you must do

Program the following task in your C++ compiler and then compile and execute them. *Write **main** function first and keep on testing the functionality of each function once created.*

ADT: PointList

Implement a **PointList** class to manage a list of **Point3D** objects. This class should support various operations for managing and manipulating the list.

Point3D structure:

```
struct Point3D
{
   float x, y, z;
};
```

The **PointList** class should have the following private data members:

1. A pointer to a dynamically allocated array of **Point3D** structures.
2. An integer **maxSize** to specify the maximum capacity of the list.
3. An integer **curSize** to track the current number of elements in the list.
4. An integer **cursor** to maintain the current position in the list. It should be initialized to **-1** if the list is empty.

Operations Supported by **PointList**:

A. **Constructor:** Accept an integer **maxSize** as an argument and initialize the list as an empty list with the specified capacity.

B. **Destructor:** Release any memory resources occupied by the **PointList** object.

C. **isEmpty():** Return true if the list is empty; otherwise, return false.

D. **isFull():** Return true if the list is empty; otherwise, return false.

E. **insert(Point3D newPoint):** Insert **newPoint** into the list. If the list is not empty, insert **newPoint** at the end; otherwise, insert it as the first (and only) data item. Move the cursor to **newPoint** and increment the current size.

F. **showStructure():** Output the data items in the list. If the list is empty, display "Empty list."

G. **getCursor():** If the list is not empty, return a copy of the data item marked by the cursor. Display an error message and return a **Point3D** object with all values set to **-1** if the list is empty.

H. **void gotoBeginning():** If the list is not empty, move the cursor to the beginning of the list.

I. **gotoEnd():** If the list is not empty, move the cursor to the data item at the end of the list (current size).

J. **gotoNext():** If the cursor is not at the end of the list, move it to the next data item and return true. Otherwise, return false.

K. **gotoPrior():** If the cursor is not at the beginning of the list, move it to the previous item and return true. Otherwise, return false.

L. **clear():** make the list empty *(a list having no elements)*.

M. **replace(Point3D newPoint):** Replace the data item marked by the cursor with **newPoint** and return true. Return false if the cursor is at an invalid position.

N. **remove():** Remove the data item marked by the **cursor** from the list. If the list is not empty after removal, keep the cursor in its current position. If the removed item was at the end of the list, move the cursor to the beginning.

Note: Ensure that you handle errors gracefully and provide meaningful error messages when necessary.

Demonstration: In the main function, create instances of the **PointList** class and demonstrate the functionality of each function clearly.

☺ ☺ ☺ **BEST OF LUCK** ☺ ☺ ☺