Homework assignment 3 on Inheritance

Offline #1

You have become familiar with the stack data structure with allows storing and retrieving data in last in first out manner (see week#1 contents). Consider a **Stack** class for integer elements with the following members:

- Protected
  - int buffer[100]: for storing all items (assuming maximum size 100)
  - int top: for pointing to the recently stored item, used to store/retrieve the next item
- Public
  - void init(): initializes to an empty stack
  - int getSize(): returns the current size of the stack
  - void push(int item): stores an item to the end of the stack, shows error message when the stack is full
  - int pop(): removes and returns the recently stored item, , shows error message when the stack is empty
  - bool isEmpty(): shows whether the stack is empty

Task 1 [2 marks]: Implement the **Stack** class according to the above specification.

There is another data structure named queue which allows storing and retrieving data in first in first out manner (the oldest item will be retrieved first). A very simple implementation of **Queue** class contains the following members:

- Private
  - int buffer[100]: for storing all items (assuming maximum size 100)
  - int head: for pointing to the recently stored item, used to store the next item
  - int tail: for pointing to the oldest item, used to retrieve the next item
- Public
  - void init(): initializes to an empty queue
  - int getSize(): returns the current size of the queue
  - void enqueue(int item): stores an item to the end of the queue, shows error message when the queue is full
  - int deque(): removes and returns the oldest item from the start of the queue, shows error message when the queue is empty
  - bool isEmpty(): shows whether the queue is empty

Task 2 [15 marks]: Now you have to implement the **Queue** class according to the above specification by **inheriting** your written **Stack** class. Note that:

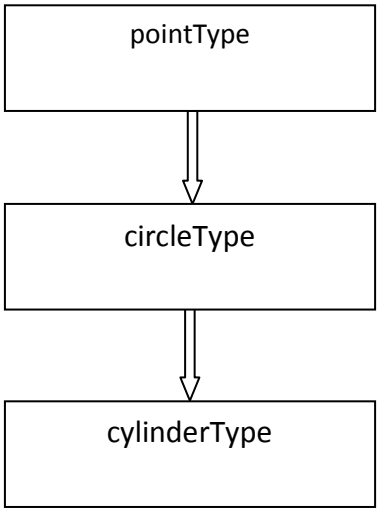
- Your **Queue** class must contain the minimal amount of code.
- You must ensure the maximum reuse of your **Stack** class.
- You can not modify the given definition of **Stack** class in anyway.

Task 2 [3 marks]: Write a main function properly demonstrating all functionalities of your **Queue** class.

Homework assignment 3 on Inheritance

Offline #2

Consider the following hierarchy:



In the "Offline_Skeleton.cpp" file you have the implementation of the pointType class and the main() function.

Task 1 [8 marks]: Implement the circleType class and its necessary functions.

Task 1 [12 marks]: Implement the cylinderType class and its necessary functions.

Implement these classes according to the above hierarchy so that the program runs smoothly and generates the desired output. The output of the program is given in the "output.txt" file.

Restrictions:

- You can not modify the pointType class and main() function.
- circleType class must inherit pointType class.
- cylinderType class must inherit circleType class.
- cylinderType class should have its own print() function.
- Write functions of circleType and cylinderType according to OOP principles. (For example, while printing base centre you should call the already available print() function of pointType). Try to reuse the codes you have written.