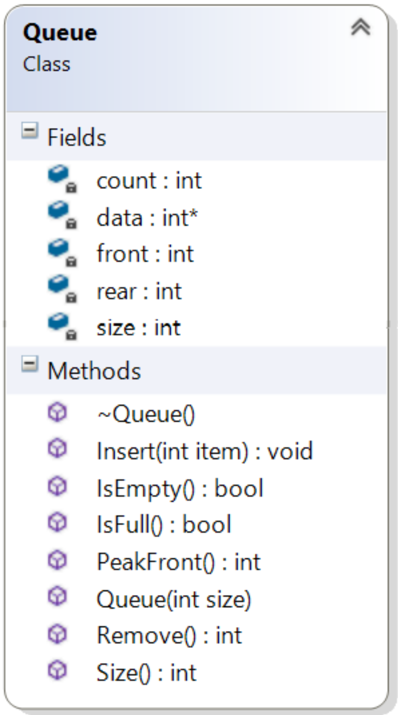
C++ Week 3 Queues

This week we shall develop two types of Queue

- The Circular Queue
- The Priority Queue

The Circular Queue

1. Create a new Visual Studio C++ Console Project and name it Queues
2. Within the project define the class below



Queue
Class

Fields
- count : int
- data : int*
- front : int
- rear : int
- size : int

Methods
- ~Queue()
- Insert(int item) : void
- IsEmpty() : bool
- IsFull() : bool
- PeakFront() : int
- Queue(int size)
- Remove() : int
- Size() : int

3. Within the constructor
   a. Instantiate the array (data) of integers, containing size elements.
   b. Set front to 0
   c. Set rear to -1
   d. Set count to 0

4. Implement PeakFront which returns the front element of data
5. Implement IsEmpty (count contains the number of occupied elements in the array).
6. Implement IsFull (size contains the number of elements in the array).
7. Implement Size (returns the number of elements in the array).

C++ Week 3 Queues

Exercise 1

Implement the Insert member function based on the pseudo code below. To simplify the code we'll assume the queue is never full.

```
If rear == size -1 then
        rear = -1
End if
Increment rear by 1
Data[ rear ] = item
Increment count by 1
```

Exercise 2

Implement the Remove member function.

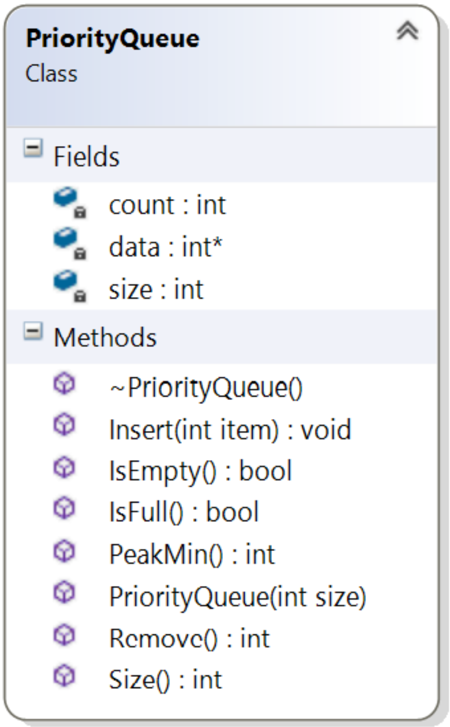
Note : It should return the removed value & remember to include the wrapping logic.

Exercise 3

Within main create a dynamic instance of the queue and insert 6 unique integers. Within a loop iterate over the contents of the queue removing its items and displaying their values in the console.

The Priority Queue

Within the same project define the PriorityQueue class below



PriorityQueue
Class

Fields
count : int
data : int*
size : int

Methods
~PriorityQueue()
Insert(int item) : void
IsEmpty() : bool
IsFull() : bool
PeakMin() : int
PriorityQueue(int size)
Remove() : int
Size() : int

C++ Week 3 Queues

Priority Queue

1. Within the constructor
    a. Instantiate the array (data) of integers, containing size elements.
    b. Set count to 0

2. Implement PeakMin which returns the count - 1 element of data
3. Implement IsEmpty (count contains the number of occupied elements in the array).
4. Implement IsFull (size contains the number of elements in the array).
5. Implement Size (returns the number of elements in the array).
6. Implement Remove which decrements count and returns the removed items

Exercise 4
Implement the pseudo code for the Insert member function

```
If queue empty then
        Set element 0 to item
        Increment count
Otherwise
        Loop j from count – 1 down to 0
                If item > looped item j
                        Move looped item to element j+1
                Otherwise
                        Exit loop
        End loop
        Set element j+1 to item
        Increment count
End if
```

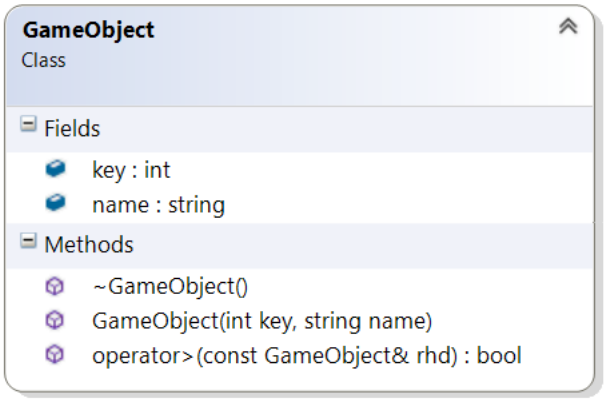
Note: In this implementation the lowest numbers are stored at the end of the array.

Exercise 5
Within main create a dynamic instance of the priority queue and insert 6 unique integers in random order. Within a loop iterate over the contents of the queue removing its items and displaying their values in the console. They should be displayed in ascending order as the lowest number is removed first.

C++ Week 3 Queues

Exercise 6

Modify the PriorityQueue class so that it can store any type of object (A Template Class). Define the GameObject Class below. The constructor assigns its parameters to the corresponding data members.



GameObject
Class

Fields

key : int
name : string

Methods

~GameObject()
GameObject(int key, string name)
operator>(const GameObject& rhd) : bool

Modify the code with main to instantiate a Template Priority Queue that stores GameObject pointers. Insert six GameObjects with unique keys and names and iterate through the queue deleting and displaying their values.
Hint: A templatesmember functions must be defined in the h file. Overload the > operator within GameObject.