DTS Lab2  
0-indexed binary heap using inheritance

Contents

[Objective 2](#_Toc358034502)

[Prototypes 2](#_Toc358034503)

[Desired Output 3](#_Toc358034504)

[Submission 4](#_Toc358034505)

# Objective

Implement the Binary Heap using either protected or private inheritance from the DynArray class rather than containment.   
For a 0-indexed binary heap, the formulas for parents and children are as follows:  
 I\*2+1 (left child)  
 I\*2+2 (right child)  
 (I-1)/2 (parent)

# Prototypes

The following functions comprise the BinaryHeap’s public interface:

/////////////////////////////////////////////////////////////////////////////  
// Function : enqueue  
// Parameters : v - the item to add to the heap  
// Notes : after the new item is added, this function ensures that the   
// smallest value in the heap is in [0]  
/////////////////////////////////////////////////////////////////////////////  
void enqueue(const Type &v)  
  
/////////////////////////////////////////////////////////////////////////////  
// Function : dequeue   
// Return : the item that was located at [0]  
// Notes : after the smallest item is dequeued, this function ensures that   
// the smallest item is in [0]. DynArray may need to be modified to change  
// Size's access type to protected so it can be decremented here  
/////////////////////////////////////////////////////////////////////////////  
Type dequeue()  
  
///////////////////////////////////////////////////////////////////////////////  
// Function : operator[]  
// Parameters : index - the index to access  
// Return : Type& - the item in the index  
// Notes: calls parent operator[]  
/////////////////////////////////////////////////////////////////////////////  
Type& operator[](int index)  
  
/////////////////////////////////////////////////////////////////////////////  
// Function : clear   
// Notes : calls parent clear()  
/////////////////////////////////////////////////////////////////////////////  
void clear()  
  
///////////////////////////////////////////////////////////////////////////////  
// Function : size  
// Return : the number of valid items in the heap  
// Notes: calls parent size()  
/////////////////////////////////////////////////////////////////////////////  
unsigned int size()

# Desired Output

Compile and run your code with the DTSLab2.cpp file provided via FSO. Your console output should match the following block identically:

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
\*\* LAB 2 CHALLENGE : \*\*  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* TEST 1 \*\*\*  
Heap Contents :   
Heap Contents : 7   
Heap Contents : 2 7   
Heap Contents : 2 7 3   
Heap Contents : 2 5 3 7   
Heap Contents : 2 5 3 7 8 \*\*\* TEST 2 \*\*\*  
Dequeue : 2 : Heap Contents : 3 5 8 7 10   
Dequeue : 3 : Heap Contents : 5 7 8 10 10   
Dequeue : 5 : Heap Contents : 7 10 8 10 10   
Dequeue : 7 : Heap Contents : 8 10 10 10 10   
Dequeue : 8 : Heap Contents : 10 10 10 10 10 Heap Contents :\*\*\* TEST 3 \*\*\*  
Heap Contents : 9   
Heap Contents : 0 9   
Heap Contents : 0 9 1   
Heap Contents : 0 6 1 9   
Heap Contents : 0 4 1 9 6   
Heap Contents : 0 4 1 9 6 7   
Heap Contents : 0 4 1 9 6 7 2   
Heap Contents : 0 3 1 4 6 7 2 9   
Heap Contents : 0 3 1 4 6 7 2 9 5   
Heap Contents : 0 3 1 4 6 7 2 9 5 8   
Dequeue : 0 : Heap Contents : 1 3 2 4 6 7 8 9 5   
Dequeue : 1 : Heap Contents : 2 3 5 4 6 7 8 9   
Dequeue : 2 : Heap Contents : 3 4 5 9 6 7 8   
Dequeue : 3 : Heap Contents : 4 6 5 9 8 7   
Dequeue : 4 : Heap Contents : 5 6 7 9 8   
Dequeue : 5 : Heap Contents : 6 8 7 9   
Dequeue : 6 : Heap Contents : 7 8 9   
Dequeue : 7 : Heap Contents : 8 9   
Dequeue : 8 : Heap Contents : 9   
Dequeue : 9 : Heap Contents :

# Submission

To submit the lab assignment:

1. Clean, build, and run DTSLab1.cpp with your DynArray.h file in Visual Studio (debug mode).
   1. clear up any warnings you encounter.
   2. verify that your output is correct by comparing it to the lab document's Desired Output section, line-by-line.
   3. ensure there are no memory leaks.

2) Submit the 'DynArray.h' & 'BinaryHeap.h' files via FSO.