CENG311 **Fall 2021**

PROGRAMMING ASSIGNMENT 1

Due date: 29.10.2021 23:00

In this assignment, you are required to implement priority queue construction in C and compare the performance of your implementation with Java-based implementation.

In your C implementation, you are required to use max binary heap, where the value of parent node will always be greater than the value of child node, and the value at the root must be maximum among all the values. Specifically, you need to use array representation of binary heap in your implementation with the following properties:

The root element will be at Arr[0]

Arr[(i-1)/2]	Returns the parent node
Arr[(2*i)+1]	Returns the left child node
Arr[(2*i)+2]	Returns the right child node

You only need to implement the *queue* function, which inserts a new element into the queue (no need for dequeue etc. operations).

You are free to choose the way in your Java code, but it is recommended to use PriorityQueue class.

You can assume that your queue can include up to 2^28 elements. To be able to store that big array, you may need to create your array by malloc function instead of an array type (If you have limitations due to your computer settings, you can reduce the array size).

Additionally, in your C code, you will implement a *convert* function, which converts the array representation into a data structure represented by pointers and links between the elements. Each element is to be represented by the following structure:

```
struct node
{
   int data;
   struct node* left;
   struct node* right;
};
```

Your program flow should be as follows:

1) Read a set of unordered integer values (separated by space) from a text file into an array.

Example: A= [45 20 14 12 31 7 11 13]

2) Build the priority queue in an array by processing the integer values one by one. Add each integer by considering that the value is representing the priority. Note that at the end of each insertion, you should have a priority queue. That is you should not have all values in some structure (like a binary tree) first, then convert into priority queue.

Example:

B= [45]
B= [45 20]
B= [45 20 14]

B= [45 20 14 12]
B= [45 31 14 12 20]
…
B= [45 31 14 13 20 7 11 12]

3) Convert your array representation into a data structure with pointers.

Example:

Node1:	data = 45, left = Node2, right = Node3
Node2:	data = 31, left = Node4, right = ...
Node3:	data = 14 ...
Node4:	data = 13 ...

…..

4) Traverse the structure, save your array in a text file.

Example file content:

45 31 14 13 20 7 11 12

Your program should measure and report the execution time for building the array structure. The measured time should not include the I/O operations. Time measurement should be done in the following order:

```
read the input file into an array          // file read operation
start time
build queue (array representation)                // the operation we want to measure
stop time
convert your queue array into structure with pointers
print resulting structure by breadth first traversal to the output file          // file write operation
```

You are required to write a report and include both functionality and performance results: 1) A simple example including ~20 integers showing that your program is building the queue correctly. 2) A test case with ~2^28 integers and timing information for that case.

Requirements:

- You are required to submit source code and report that includes implementation details and performance results.
- You need to work individually, no group work is allowed.
- You should develop your own code. For any part, if you use any source code available on the web, give the reference in your report.
- For the file operations, you can directly use the code from the following link: https://www.geeksforgeeks.org/basics-file-handling-c/
- No late homework will be accepted.

Submission: You are required to submit your commented source code and report to cloud-lms. Please create a compressed file including all source files and report; and name it as yourstudentnumber_P1.zip.