Lab 10: Heap Lab.

Dr. Tanvir Ahmed

Attend the lab and complete it with the TA! No need to keep working after the lab if you can just follow and work actively during the lab.

Is it a Heap?

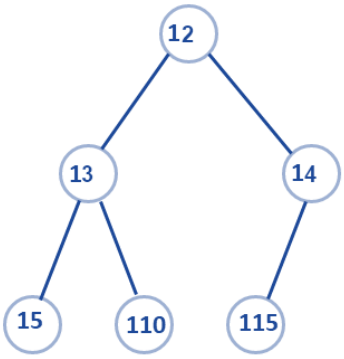
In this lab we will write a code that will check whether an array represents a minheap or not.

For example:

Is this array representing a minheap?

12	13	14	15	110	115

Difficult to say, without looking it like a tree. Let's see the tree:

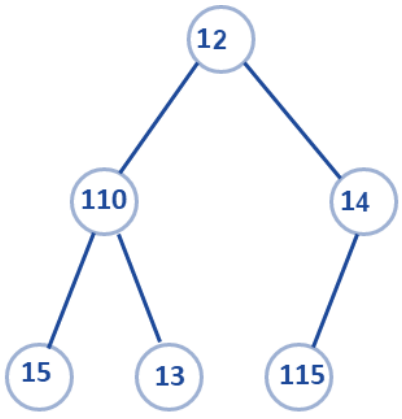


Yes, it seems the array represents a minheap as for any node, all the nodes of the sub tree is smaller than the data at that node! Additionally, it is also a complete binary tree

How about this one?

12	110	14	15	13	115

Let's see the tree:



However, this is not a minheap as 110 > its children (15 and 13)

Now, if we want to write a code to check this, we have to keep checking from the root and compere it with the children. In our implementation of heap, we have started with the array index 1 and kept our root in the index 1. However, a given array will start with 0 and we have to modify our formula to accommodate that.

So, in this case, as the first item is located at index 0 and we are considering that item as the root of the heap, its left child will be at index 1 and right child will be at index 2.

So, generalizing, left child will be at 2*i+1 and right child will be at 2*i+2

For this type of implementation where root starts at index 0, the parent of a node at index i will be at

(i-1)/2

We will try to write **a recursive version** and **one iterative version**.

For a recursive version the function prototype would be:

```
int isHeap_recursive(int arr[], int I, int n); //i is the starting point of the array
```

The main idea is to compare the left child with the parent and then right child with the parent and keep checking until it satisfies the properly. If it **does not satisfy** at any point, return 0. If it keeps satisfying the property and if you see you have reached outside of the array and your code has not returned false, it means it is really a min heap. The final result depends on the result from left sub tree and result from the right sub tree. Use && operation to get your final result.

Try this!

Also try an iterative approach:

The function prototype would be:

```
int isHeap_iter(int arr[],  int n);
```

More hints: the loop will be like this: for (int i=0; i<=(n-2)/2; i++) //start from root and end at last parent node. Why (n-2)/2? Because last index is n-1. So, parent is at (n-2)/2