**National University of Computer and Emerging Sciences**



Laboratory Manual

for

Data Structures Lab

|  |  |
| --- | --- |
| Course Instructor | Ms. Syeda Tayyaba Bukhari |
| Lab Instructor(s) | Ms. Sonia Anum  Ms. Samia Akhter |
| Section | BDS-3B |
| Semester | Fall 2022 |

**Department of Computer Science**

FAST-NU, Lahore, Pakistan

**Objectives:**

In this lab, students will practice:

1. Implementation of Heap Sort
2. Practice Huffman Encoding

**Heap Sort**

**Heap sort** is a comparison-based sorting technique based on [Binary Heap](http://www.geeksforgeeks.org/binary-heap/) data structure. It is similar to the [Selection Sort](http://www.geeksforgeeks.org/selection-sort/) where we first find the minimum element and place the minimum element at the beginning. Repeat the same process for the remaining elements.

You have to follow following steps to implement the Heap Sort algorithm:

* Build a max heap from the input data.
* At this point, the maximum element is stored at the root of the heap. Replace it with the last item of the heap followed by reducing the size of the heap by 1. Finally, heapify the root of the tree.
* Repeat step 2 while the size of the heap is greater than 1.

**Task:**

1. Implement a **Max Heap** class containing all the required data members and functions to convert a given array into a Max Heap.
   1. Data Members: int \*arr, int capacity, int size
   2. Functions:
      1. Default constructor
      2. Overloaded Constructor MaxHeap(int \_capacity)
      3. Heapify() function
      4. Insert() function
      5. Print() function
2. Input the following array from user using insert function:

**14 19 42 33 26 31 27 44 35 10**

1. Display the Output. Your output should be a unsorted max heap.
2. Create a function **heapSort()** that applies heap sort algorithm on the input array and sort it in increasing order
3. Display the Output

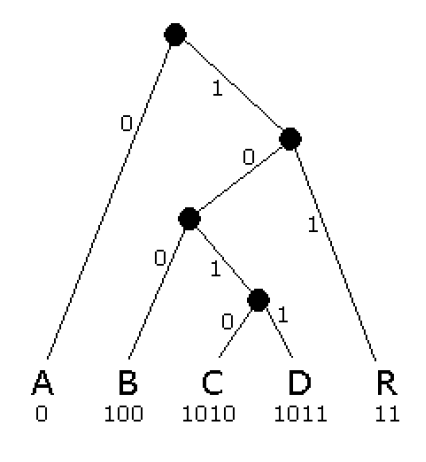
**Huffman Codes**

Huffman coding is a lossless data compression algorithm. The idea is to assign variable-length codes to input characters, lengths of the assigned codes are based on the frequencies of corresponding characters. The most frequent character gets the smallest code and the least frequent character gets the largest code.

**NOTE: Solve these tasks on paper and submit it to the instructor. No submission is allowed on Google Classroom. In case of plagiarism, whole lab work will be marked 0.**

**Task-1:**

Write the Huffman code for each of the characters in this tree.



**Task-2:**

Write Huffman Code for following text:

**Eerie eyes seen near lake.**

Show complete step wise working, otherwise the solutions will not be accepted.