National University of Computer and Emerging Sciences, Lahore Campus

Course:	Data Structures	Course Code:	CS 2001
Program:	BS(Computer Science)	Semester:	Fall 2021
Due Date	25-Dec-2021 at 11:59 pm	Total Marks:	10
Type:	Assignment 5		
Sections	E and F		

Important Instructions:

1. Submit your source files in a zipped file named as your roll number, i.e., 20L-1111.zip.
2. You are not allowed to copy solutions from other students. We will check your code for plagiarism using plagiarism checkers. If any sort of cheating is found, negative marks will be given to all students involved.
3. Late submission of your solution is not allowed.

Question :

Implement a class Huffman which contains the root Node of Huffman binary tree. Each node in Huffman is of type HNode.

```
struct HNode
{
  int freq;
  char character;
  HNode *left, *right;
};
```

Note:

a. A valid character is stored only in leaf nodes in a Huffman Tree.
b. In a leaf node, the value in freq variable corresponds to the frequency of the character that the node represents.
c. In a non-leaf node, the value in freq variable is the sum of freq variables stored in its left child and right child.

1. Implement a member function createHuffman that is passed as parameter a filename. The createHuffman function first finds the frequency of all unique characters in the file. Frequencies can be found efficiently by using a hash map (You can either use STL implementation of hash map or the hash map class that you will create in the lab) that stores character as key and its frequency as value. For example while reading the file, if you encounter a new character in the file, add the character in the hash map with character as key and 1 as value. However, if the character was encountered previously, then simply get the value (frequency) from the hashmap and add 1 to it (we are incrementing the frequency count). After the frequencies of all characters have been found, generate the Huffman binary tree that we discussed in the class using min heap (You can use min heap implementation available in STL, or your own code implemented in the lab). `void createHuffman(char *fileName)`

2. Implement a member function printHuffman which prints the Huffman code of every character along with the character and its frequency. `void printHuffman() const`

3. Destructor

4. Create a main function which creates a Huffman object, calls the createHuffman function and passes the name of a text file to it. The main function then calls the printHuffman function to print Huffman code of all unique characters.

Your program should run on any text file containing ASCII characters. The file can have any number of characters.