Assignment 6(Huffman coding):

Cpp code

```cpp
#include <bits/stdc++.h>
using namespace std;

struct Node
{
    char character;
    unsigned frequency;
    Node *leftPointer, *rightPointer;
    Node(char character, unsigned frequency)
    {
        leftPointer = rightPointer = NULL;
        this->character = character;
        this->frequency = frequency;
    }
};

struct heapObjCompare
{
    bool operator()(Node *l, Node *r){return (l->frequency >
r->frequency);}
};

void printHuffman(struct Node *root, string s)
{
    if (!root)
        return;

    if (root->character != '$')
        cout << root->character << ": " << s << "\n";

    printHuffman(root->leftPointer, s + "0");
    printHuffman(root->rightPointer, s + "1");
}

void createHeap(char character[], int frequency[], int size)
{
    struct Node *leftPointer, *rightPointer, *top;
    priority_queue<Node *, vector<Node *>, heapObjCompare> minHeap;
    for (int i = 0; i < size; ++i)
        minHeap.push(new Node(character[i], frequency[i]));
```

```
    while (minHeap.size() != 1)
    {
        leftPointer = minHeap.top();
        minHeap.pop();
        rightPointer = minHeap.top();
        minHeap.pop();
        top = new Node('$', leftPointer->frequency +
rightPointer->frequency);
        top->leftPointer = leftPointer;
        top->rightPointer = rightPointer;
        minHeap.push(top);
    }
    printHuffman(minHeap.top(), "");
}

int main()
{
    char arr[] = {'a', 'b', 'c', 'd', 'e', 'f'};
    int frequency[] = {5, 9, 12, 13, 16, 45};
    createHeap(arr, frequency, (int)(sizeof(arr) / sizeof(arr[0])));
    return 0;
}
```

- Test cases passed
- Completed on 16/3/23

Q/A:

1. How long did you spend on this assignment?
    a. 1day
2. Based on your effort, what letter grade would you say you earned?
    a. On a scale of 1 to 10. I would grade this as 10/10.
3. Based on your solution, what letter grade would you say you earned?
    a. On a scale of 1 to 10. I would grade this as 9/10.
4. Provide a summary of what doesn't work in your solution, along with an explanation of how you attempted to solve the problem and where you feel you struggled?
    a. In this code, the Huffman tree is created using a priority queue (min heap) data structure. First, the input characters and their frequencies are inserted into the priority queue as nodes of the Huffman tree. Then, the two nodes with the lowest frequencies are extracted from the priority queue, and a new internal node is created with their sum as the frequency. This new node is then inserted back into the priority queue. This process is repeated until there is only one node left in the priority queue, which becomes the root of the Huffman tree.