DAA Assignment 3

Name: Digvijay Pawar
Class: TY.Btech Comp B2
Gr.No: 21810344
Roll No: 322043

Greedy approach (Huffman Coding)

Code Implementation :

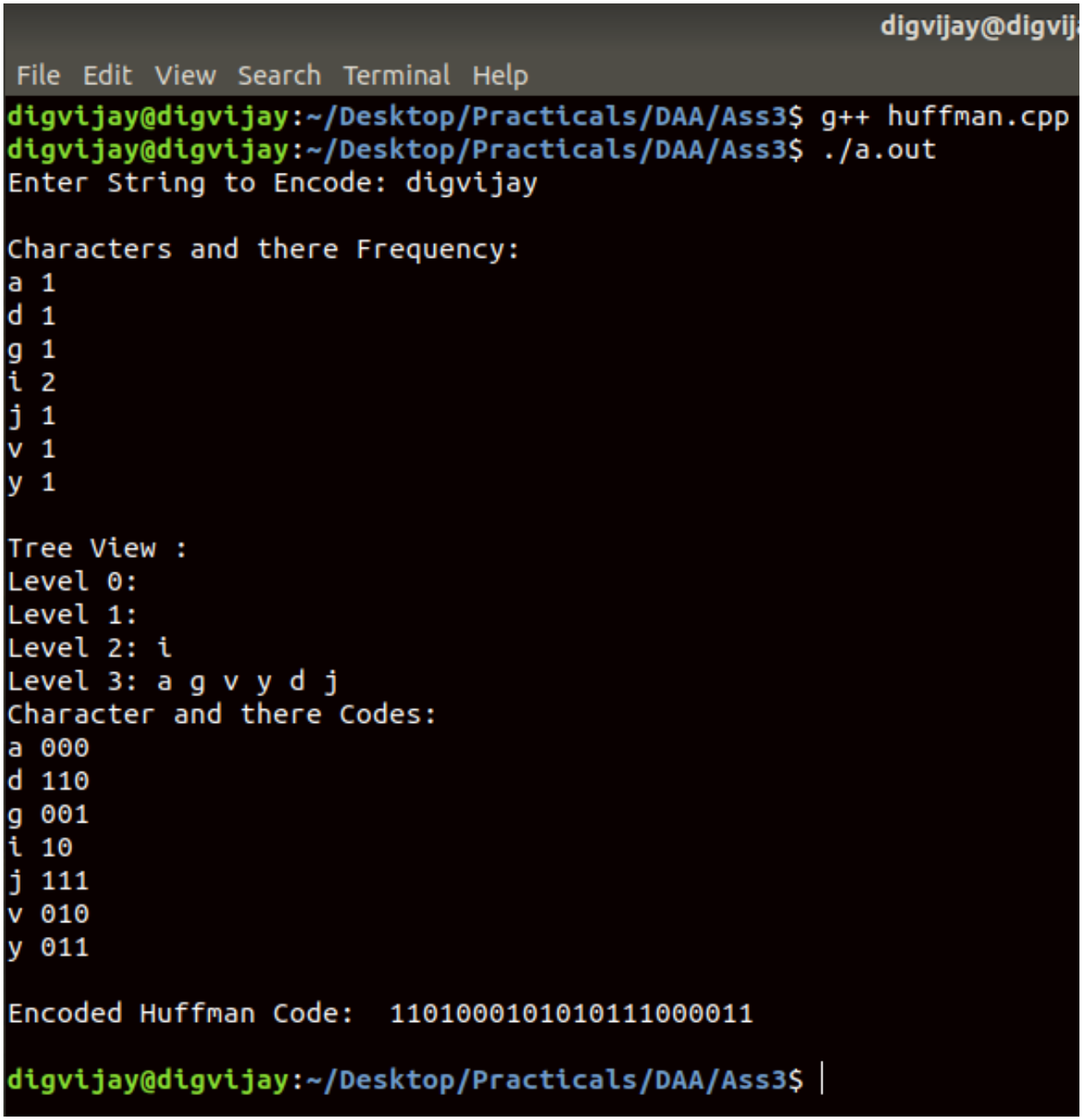
```
#include <iostream>
#include <vector>
#include <map>
#include <algorithm>
#include <iterator>
#include <bits/stdc++.h>
using namespace std;

map<char, string> codes;

struct Node
{
    char data;
    int freq;
    Node *left, *right;
    Node(char data, int freq)
    {
        left = right = NULL;
        this->data = data;
        this->freq = freq;
    }
};
```

```
struct comp
{
    bool operator()(Node* r, Node* l)
    {
        return (r->freq > l->freq);
    }
};

priority_queue<Node*, vector<Node*>, comp> tree;

void storeCodes(struct Node* root, string str)
{
    if (root==NULL)
        return;
    if (root->data != '$')
        codes[root->data]=str;
    storeCodes(root->left, str + "0");
    storeCodes(root->right, str + "1");
}

void huffman (map<char,int> freq,int size)
{
  struct Node *left, *right, *top;
  for (map<char,int>::iterator v=freq.begin(); v!=freq.end(); v++)
      tree.push(new Node(v->first, v->second));
  while (tree.size() != 1)
  {
      left = tree.top();
      tree.pop();
      right = tree.top();
      tree.pop();
      top = new Node('$', left->freq + right->freq);
      top->left = left;
      top->right = right;
      tree.push(top);
```

```
  }
  storeCodes(tree.top(), "");
}

void printLevel(Node* root)
{
    if (root == NULL)
      return;
    queue<Node *> q;
    q.push(root);
    int i=0;
    while (q.empty() == false)
    {
        int nodeCount = q.size();
        if(i>0)
          std::cout << "Level "<<i<<": ";
        while (nodeCount > 0)
        {
            Node *node = q.front();
            if(node->data!='$')
              cout << node->data << " ";
            q.pop();
            if (node->left != NULL)
                q.push(node->left);
            if (node->right != NULL)
                q.push(node->right);
            nodeCount--;
        }
        if(i>0)
          std::cout<<"\n";
        i++;
    }
    std::cout<< '\n';
}
```

```
int main()
{
  string s,encodedString;
  std::cout << "Enter String to Encode: ";
  std::cin >> s;

  map<char, int> map;
  for (int i = 0; i < s.length(); i++) {
    map[s[i]]++;
  }

  std::cout << "\nCharacters and there Frequency: " << '\n';
  for(auto it:map)
  {
    std::cout <<it.first<<" "<<it.second<< '\n';
  }

  std::cout << "\nTree View : " << '\n';

  huffman(map,s.length());
  printLevel(tree.top());
  cout << "Character and there Codes:\n";
  for (auto v=codes.begin(); v!=codes.end(); v++)
      cout << v->first <<' ' << v->second << endl;

  for (auto i: s)
        encodedString+=codes[i];

  cout << "\nEncoded Huffman Code:  " << encodedString << endl;
  std::cout<< '\n';
  return 0;
}
```

Output:


digvijay@digvij
File Edit View Search Terminal Help
digvijay@digvijay:~/Desktop/Practicals/DAA/Ass3$ g++ huffman.cpp
digvijay@digvijay:~/Desktop/Practicals/DAA/Ass3$ ./a.out
Enter String to Encode: digvijay
Characters and there Frequency:
a 1
d 1
g 1
i 2
j 1
v 1
y 1
Tree View :
Level 0:
Level 1:
Level 2: i
Level 3: a g v y d j
Character and there Codes:
a 000
d 110
g 001
i 10
j 111
v 010
y 011
Encoded Huffman Code: 11010001010101110000011
digvijay@digvijay:~/Desktop/Practicals/DAA/Ass3$