LAB-14

Name: K V Jaya Harsha

Roll no: CS23B1034 Date: 06-11-2024

Q1. Merge Sort (in cpp)

```cpp
//K V Jaya Harsha
//CS23B1034
#include <iostream>
using namespace std;
void merge(char arr[], int beg, int mid, int end) {
    int i = beg, j = mid + 1;
    int index = 0;
    char temp[end - beg + 1];
    while (i <= mid && j <= end) {
        if (arr[i] < arr[j]) {
            temp[index] = arr[i];
            i++;
        } else {
            temp[index] = arr[j];
            j++;
        }
        index++;
    }
    while (i <= mid) {
        temp[index] = arr[i];
        i++;
        index++;
    }
    while (j <= end) {
        temp[index] = arr[j];
        j++;
        index++;
    }
    for (int k = 0; k < index; k++) {
        arr[beg + k] = temp[k];
    }
}
void merge_sort(char arr[], int beg, int end) {
    if (beg < end) {
        int mid = (beg + end) / 2;
        merge_sort(arr, beg, mid);
        merge_sort(arr, mid + 1, end);
        merge(arr, beg, mid, end);
    }
}
int main() {
    char sdtarray[10]={'c','a','l','m','e','k','v','j','b','y'};
    merge_sort(sdtarray, 0, 9);
    cout << "Sorted characters: ";
    for (int i = 0; i < 10; i++) {
        cout << sdtarray[i] << " ";
    }
    cout << endl;
    return 0;
}
```

```
t } ; if ($?) { .\mergesort }
Sorted characters: a b c e j k l m v y
PS C:\Users\harsh\OneDrive\Documents\Desktop\challenge\
bs\lab-14>
```

Q2. Double hashing. (in cpp)

```cpp
//K V Jaya Harsha
//CS23B1034
#include <iostream>
#include <vector>
using namespace std;
struct DoubleHashing {
    int buckets;
    int p;
    vector<int> table;
    DoubleHashing(int m, int p) : buckets(m), p(p), table(m, -1) {}
    int hash1(int key) {
        return key % buckets;
    }
    int hash2(int key) {
        return p - (key % p);
    }
    void insert(int key) {
        int index = hash1(key);
        if (table[index] != -1) {
            int step = hash2(key);
            int i = 1;
            while (table[(index + i * step) % buckets] != -1) {
                i++;
            }
            index = (index + i * step) % buckets;
        }
        table[index] = key;
    }
    void display() {
        for (int i = 0; i < buckets; i++) {
            if (table[i] != -1)
                cout << i << " --> " << table[i] << endl;
            else
                cout << i << " --> " << "EMPTY" << endl;
        }
    }
};
int main() {
    DoubleHashing hashTable(10, 7);
    vector<int> keys = {73, 42, 97, 104, 10, 92, 17, 37};
    for (int key : keys) {
        hashTable.insert(key);
    }
    cout << "Hash table after inserting elements:\n";
    hashTable.display();
    return 0;
}
```

```
Hash table after inserting elements:
0 --> 10
1 --> 17
2 --> 42
3 --> 73
4 --> 104
5 --> EMPTY
6 --> 37
7 --> 97
8 --> 92
9 --> EMPTY
PS C:\Users\harsh\OneDrive\Documents\
```