TECHNICAL TRAINING DSA - CODING PRACTICE PROBLEMS

Name: Rishi Kumar S
Dept: CSBS
Date: 18-11-2024

Problem 1: **Bubble Sort**

Code:

```
#include <iostream>
#include <vector>
using namespace std;

void bubblesort(vector<int> &arr){
    int n=arr.size();
    for(int i=0;i<n-1;i++){
        for(int j=i+1;j<n;j++){
            if(arr[j]<arr[i]) swap(arr[i],arr[j]);
        }
    }
}

int main(){
    int n;
    cout<<"Enter length: ";
    cin>>n;
    vector<int> arr(n);
    for(int i=0;i<n;i++){
        cin>>arr[i];
    }

    bubblesort(arr);

    for(int x:arr){
        cout<<x<<" ";
    }

    return 0;
}
```

Output:

```
PS C:\Users\rishi\Documents\DSA\18.11.24> g++ bubblesort.cpp -o bubblesort.exe
PS C:\Users\rishi\Documents\DSA\18.11.24> .\bubblesort.exe
Enter length: 6
6 1 8 3 7 3
1 3 3 6 7 8
PS C:\Users\rishi\Documents\DSA\18.11.24>
```

Time Complexity: O(n^2)
Space Complexity: O(1)

Problem 2: **Quick Sort**

Code:

```
#include <iostream>
#include <vector>
using namespace std;

int part(vector<int> &arr,int left, int right){
    int pivot=arr[right];
    int i=left-1;
    for(int j=left;j<right;j++){
        if(arr[j]<pivot){
            i++;
            swap(arr[i],arr[j]);
        }
    }
    swap(arr[i+1],arr[right]);
    return i+1;
}

void quicksort(vector<int> &arr, int left, int right){
    if(left<right){
        int pivot=part(arr,left,right);

        quicksort(arr,left,pivot-1);
        quicksort(arr,pivot+1,right);
    }
}

int main(){
    int n;
    cout<<"Enter length: ";
    cin>>n;
    vector<int> arr(n);
```

```
    for(int i=0;i<n;i++){
        cin>>arr[i];
    }

    quicksort(arr,0,n-1);

    for(int x:arr){
        cout<<x<<" ";
    }

    return 0;
}
```

Output:

```
PS C:\Users\rishi\Documents\DSA\18.11.24> g++ quicksort.cpp -o quicksort.exe
PS C:\Users\rishi\Documents\DSA\18.11.24> .\quicksort.exe
Enter length: 6
49 24 15 32 77 31
15 24 31 32 49 77
PS C:\Users\rishi\Documents\DSA\18.11.24>
```

Time Complexity: O(n log n)
Space Complexity: O(n) #Recursion stack space

Problem 3: **Non Repeating Character**

Given a string s consisting of lowercase Latin Letters. Return the first non-repeating character in s. If there is no non-repeating character, return '$'.
Note: When you return '$' driver code will output -1.

Code:

```
#include <iostream>
#include <vector>
#include <string>
using namespace std;

const int maxi=26;
    char nonRepeatingChar(string &s) {
        vector<int> maxchar(maxi,0);
        for(char ch:s){
            maxchar[ch-'a']++;
```

```
        }
        for(char ch:s){
            if(maxchar[ch-'a']==1) return ch;
        }
        return '$';
}

int main(){
    string s;
    cout<<"Enter String: ";
    cin>>s;
    cout<<"Answer: "<<nonRepeatingChar(s);
    return 0;
}
```

OUTPUT:

```
PS C:\Users\rishi\Documents\DSA\18.11.24> g++ non_rep.cpp -o non_rep.exe
PS C:\Users\rishi\Documents\DSA\18.11.24> .\non_rep.exe
Enter String: geeksforgeeks
Answer: f
PS C:\Users\rishi\Documents\DSA\18.11.24>
```

Time Complexity: O(n)
Space Complexity: O(26)

Problem 4: **K largest elements**

Given an array arr[] of positive integers and an integer k, Your task is to return k largest elements in decreasing order.

CODE:

```
// C++ program to find the k largest elements in the
// array using min heap

#include <iostream>
#include <vector>
#include <queue>
#include <algorithm>
using namespace std;

vector<int> kLargest(vector<int> &arr, int k) {
    priority_queue<int, vector<int>, greater<int>>
```

```
            minH(arr.begin(), arr.begin() + k);
    for (int i = k; i < arr.size(); i++) {
        if(minH.top() < arr[i]) {
            minH.pop();
            minH.push(arr[i]);
        }
    }


    vector<int> res;
    while (!minH.empty()) {
        res.push_back(minH.top());
        minH.pop();
    }
    reverse(res.begin(), res.end());
    return res;
}

int main(){
    int n;
    cout<<"Enter length: ";
    cin>>n;
    vector<int> arr(n);
    for(int i=0;i<n;i++){
        cin>>arr[i];
    }

    cout<<"Enter value of k: ";
    int k;
    cin>>k;

    vector<int> ans=kLargest(arr,k);

    for(int x:ans){
        cout<<x<<" ";
    }

    return 0;
}
```

Output:

```
PS C:\Users\rishi\Documents\DSA\18.11.24> g++ large_k.cpp -o large_k.exe
PS C:\Users\rishi\Documents\DSA\18.11.24> .\large_k.exe
Enter length: 6
Enter array elements: 12 42 75 35 86 34
Enter value of k: 3
86 75 42
PS C:\Users\rishi\Documents\DSA\18.11.24>
```

Time Complexity: O(n * log(k))
Space Complexity: O(k)