Name: ANYA AGARWAL

Section: B (13)

Roll no: 2021122

1// Given an unsorted array of alphabets containing duplicate elements. Design an algorithm

// and implement it using a program to find which alphabet has maximum number of

// occurrences and print it. (Time Complexity = O(n)) (Hint: Use counting sort)

#include<iostream>

#include<vector>

#include<algorithm>

#include<fstream>

using namespace std;

void findMaxOccurrences(vector<char>&arr, int n) {

vector<int>count(26,0);

for (int i = 0; i < n; ++i) {

count[arr[i] - 'a']++;

}

char maxChar;

int maxCount = 1;

bool b = false;

for (int i = 0; i < 26; ++i) {

if (count[i] > maxCount) {

maxCount = count[i];

maxChar = 'a' + i;

b = true;

}

}

if (b) {

cout << maxChar<< "-" << maxCount <<endl;

} else {

cout << "No Duplicates Present" <<endl;

}

int main()

{

ifstream infile("input11.txt");

int t;

infile >> t;

while (t) {

int n, k;

infile >> n;

vector<char> arr(n);

for (int i = 0; i < n; i++) {

infile >> arr[i];

}

findMaxOccurrences(arr, n);

t--;

}

infile.close();

return 0;

}

//\*\*\*\*\*\*\*\*Input\*\*\*\*\*\*\*\*

// 3

// 10

// a e d w a d q a f p

// 15

// r k p g v y u m q a d j c z e

// 20

// g t l l t c w a w g l c w d s a a v c l

//\*\*\*\*\*\*\*\*Output\*\*\*\*\*\*\*\*

// a-3

// No Duplicates Present

// l-4

2.// Given an unsorted array of integers, design an algorithm and implement it using a

// program to find whether two elements exist such that their sum is equal to the given

// key element. (Time Complexity = O(n log n))

#include<iostream>

#include<vector>

#include<algorithm>

#include<fstream>

using namespace std;

void PairSum(vector<int>&arr, int n, int key) {

sort(arr.begin(),arr.end());

int left = 0;

int right = n - 1;

bool b = false;

while (left < right) {

int sum = arr[left] + arr[right];

if (sum == key) {

b = true;

cout <<arr[left] << " " << arr[right] <<endl;

break;

} else if (sum < key) {

left++;

} else {

right--;

}

if (!b) {

cout << "No Such Elements Exist" <<endl;

}

}

int main()

{

ifstream infile("input12.txt");

int t;

infile >> t;

while (t) {

int n, k;

infile >> n;

vector<int> arr(n);

for (int i = 0; i < n; i++) {

infile >> arr[i];

}

infile >> k;

PairSum(arr, n, k);

t--;

}

infile.close();

return 0;

}

//\*\*\*\*\*\*\*\*Input\*\*\*\*\*\*\*\*

// 2

// 10

// 64 28 97 40 12 72 84 24 38 10

// 50

// 15

// 56 10 72 91 29 3 41 45 61 20 11 39 9 12 94

// 302

//\*\*\*\*\*\*\*\*Output\*\*\*\*\*\*\*\*

// 10 40

// No Such Elements Exist

3.You have been given two sorted integer arrays of size m and n. Design an algorithm and implement it using a program to find list of elements which are common to both. (Time Complexity = O(m+n)).

#include <iostream>

#include <fstream>

using namespace std;

int main() {

ifstream inFile1, inFile2;

ofstream outFile;

int a[10], b[10], c[10];

int n, m, k = 0;

inFile1.open("input1.txt");

if (!inFile1) {

cerr << "Unable to open input1.txt";

return 1;

}

inFile2.open("input2.txt");

if (!inFile2) {

cerr << "Unable to open input2.txt";

return 1;

}

outFile.open("output.txt");

if (!outFile) {

cerr << "Unable to open output.txt";

return 1;

}

inFile1 >> n;

for (int i = 0; i < n; i++) {

inFile1 >> a[i];

}

inFile2 >> m;

for (int i = 0; i < m; i++) {

inFile2 >> b[i];

}

int i = 0, j = 0;

while (i < n && j < m) {

if (a[i] == b[j]) {

c[k++] = a[i];

i++;

j++;

} else if (a[i] > b[j]) {

j++;

} else {

i++;

}

}

outFile << "The new array is: ";

for (int i = 0; i < k; i++) {

outFile << c[i] << " ";

}

outFile << endl;

inFile1.close();

inFile2.close();

outFile.close();

return 0;

}

