NAME- Amrit Aanand

REG NO- RA1911003010917

LAB WEEK-3

**CROSS+ROAD= DANGER**

CODE-

#include <iostream>

#include<vector>

using namespace std;

vector<int> use(10); //set 1, when one character is assigned previously

struct node {

char letter;

int value;

};

int isValid(node\* nodeList, const int count, string s1, string s2, string s3) {

int val1 = 0, val2 = 0, val3 = 0, m = 1, j, i;

for (i = s1.length() - 1; i >= 0; i--) { //find number for first string

char ch = s1[i];

for (j = 0; j < count; j++)

if (nodeList[j].letter == ch) //when ch is present, break the loop

break;

val1 += m \* nodeList[j].value;

m \*= 10;

}

m = 1;

for (i = s2.length() - 1; i >= 0; i--) { //find number for second string

char ch = s2[i];

for (j = 0; j < count; j++)

if (nodeList[j].letter == ch)

break;

val2 += m \* nodeList[j].value;

m \*= 10;

}

m = 1;

for (i = s3.length() - 1; i >= 0; i--) { //find number for third string

char ch = s3[i];

for (j = 0; j < count; j++)

if (nodeList[j].letter == ch)

break;

val3 += m \* nodeList[j].value;

m \*= 10;

}

if (val3 == (val1 + val2)) //check whether the sum is same as 3rd string or not

return 1;

return 0;

}

bool permutation(int count, node\* nodeList, int n, string s1, string s2, string s3) {

if (n == count - 1) { //when values are assigned for all characters

for (int i = 0; i < 10; i++) {

if (use[i] == 0) { // for those numbers, which are not used

nodeList[n].value = i; //assign value i

if (isValid(nodeList, count, s1, s2, s3) == 1) { //check validation

cout << "Solution found: ";

for (int j = 0; j < count; j++) //print code, which are assigned

cout << " " << nodeList[j].letter << " = " << nodeList[j].value;

return true;

}

}

}

return false;

}

for (int i = 0; i < 10; i++) {

if (use[i] == 0) { // for those numbers, which are not used

nodeList[n].value = i; //assign value i and mark as not available for future use

use[i] = 1;

if (permutation(count, nodeList, n + 1, s1, s2, s3)) //go for next characters

return true;

use[i] = 0; //when backtracks, make available again

}

}

return false;

}

bool solvePuzzle(string s1, string s2,string s3) {

int uniqueChar = 0; //Number of unique characters

int len1 = s1.length();

int len2 = s2.length();

int len3 = s3.length();

vector<int> freq(26); //There are 26 different characters

for (int i = 0; i < len1; i++)

++freq[s1[i] - 'A'];

for (int i = 0; i < len2; i++)

++freq[s2[i] - 'A'];

for (int i = 0; i < len3; i++)

++freq[s3[i] - 'A'];

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

if (freq[i] > 0) //whose frequency is > 0, they are present

uniqueChar++;

if (uniqueChar > 10) { //as there are 10 digits in decimal system

cout << "Invalid strings";

return 0;

}

node nodeList[uniqueChar];

for (int i = 0, j = 0; i < 26; i++) { //assign all characters found in three strings

if (freq[i] > 0) {

nodeList[j].letter = char(i + 'A');

j++;

}

}

return permutation(uniqueChar, nodeList, 0, s1, s2, s3);

}

int main() {

string s1 = "CROSS";

string s2 = "ROADS";

string s3 = "DANGER";

if (solvePuzzle(s1, s2, s3) == false)

cout << "No solution";

}

OUTPUT-

