**Name: Saurav Jaiswal**

**UID: 18BCS1047**

**Section: 18K2G1**

**Subject/Subject-code: Design and Analysis of Algorithms Lab (CSP-309)**

**PRACTICE PROBLEMS**

**Question 1 –** **Write a program to print the all possible subsets and subarrays of given array.**

**Solution:**

//Print subarrays and subsets

#include<bits/stdc++.h>

using namespace std;

void subsetUtil(int a[],int ind,int n,vector<vector<int>> &result,vector<int> &subset){

result.push\_back(subset);

int i;

for(i=ind;i<n;i++){

subset.push\_back(a[i]);

subsetUtil(a,i+1,n,result,subset);

subset.pop\_back();

}

return;

}

void subarray(int a[],int n,vector<vector<int>> &result,vector<int> &subset){

int i,j,k;

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

for(j=i;j<n;j++){

for(k=i;k<=j;k++)

subset.push\_back(a[k]);

result.push\_back(subset);

subset.clear();

}

}

}

int main(){

int a[]={1,2,3};

int n=sizeof(a)/sizeof(a[0]);

int i,j;

vector<int> subset;

vector<vector<int>> result;

cout<<"The subsets are:\n";

subsetUtil(a,0,n,result,subset);

for(i=0;i<result.size();i++){

for(j=0;j<result[i].size();j++)

cout<<result[i][j]<<" ";

cout<<"\n";

}

subset.clear();

result.clear();

cout<<"The subarrays are:\n";

subarray(a,n,result,subset);

for(i=0;i<result.size();i++){

for(j=0;j<result[i].size();j++)

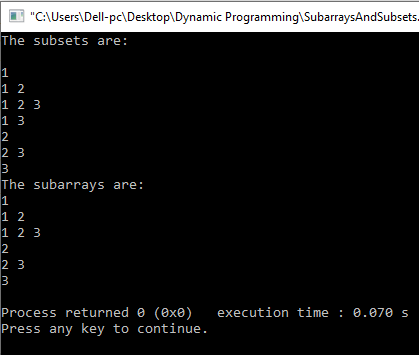
cout<<result[i][j]<<" ";

cout<<"\n";

}

}

**OUTPUT-**



**Question 2 –** **Leetcode – Coin Change**

You are given coins of different denominations and a total amount of money *amount*. Write a function to compute the fewest number of coins that you need to make up that amount. If that amount of money cannot be made up by any combination of the coins, return -1.

You may assume that you have an infinite number of each kind of coin.

**Example 1:**

**Input:** coins = [1,2,5], amount = 11

**Output:** 3

**Explanation:** 11 = 5 + 5 + 1

**Example 2:**

**Input:** coins = [2], amount = 3

**Output:** -1

**Example 3:**

**Input:** coins = [1], amount = 0

**Output:** 0

**Example 4:**

**Input:** coins = [1], amount = 1

**Output:** 1

**Example 5:**

**Input:** coins = [1], amount = 2

**Output:** 2

**Constraints:**

* 1 <= coins.length <= 12
* 1 <= coins[i] <= 231 - 1
* 0 <= amount <= 104

**Solution:**

class Solution {

public:

int coinChange(vector<int>& coins, int amount) {

int n=coins.size();

int i,j,t[n+1][amount+1];

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

for(j=0;j<=amount;j++){

if(i==0)

t[i][j]=INT\_MAX-1;

if(j==0)

t[i][j]=0;

}

}

t[0][0]=INT\_MAX-1;

for(j=1;j<=amount;j++){

if(j%coins[0]==0)

t[1][j]=j/coins[0];

else

t[1][j]=INT\_MAX-1;

}

for(i=2;i<=n;i++){

for(j=1;j<=amount;j++){

if(coins[i-1]<=j)

t[i][j]=min(1+t[i][j-coins[i-1]],t[i-1][j]);

else

t[i][j]=t[i-1][j];

}

}

if(t[n][amount]!=INT\_MAX-1)

return t[n][amount];

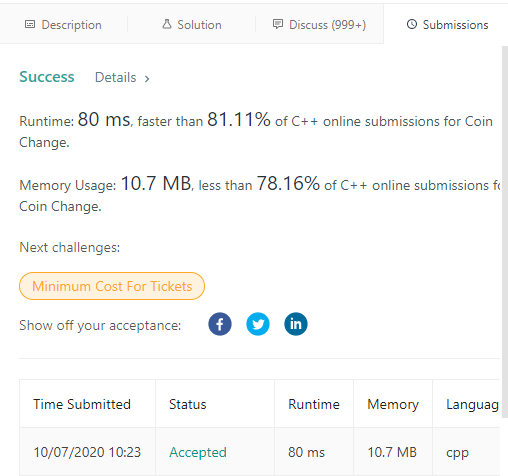
else

return -1;

}

};

**OUTPUT-**



**Question 3 –** **Leetcode – Coin Change 2**

You are given coins of different denominations and a total amount of money. Write a function to compute the number of combinations that make up that amount. You may assume that you have infinite number of each kind of coin.

**Example 1:**

**Input:** amount = 5, coins = [1, 2, 5]

**Output:** 4

**Explanation:** there are four ways to make up the amount:

5=5

5=2+2+1

5=2+1+1+1

5=1+1+1+1+1

**Example 2:**

**Input:** amount = 3, coins = [2]

**Output:** 0

**Explanation:** the amount of 3 cannot be made up just with coins of 2.

**Example 3:**

**Input:** amount = 10, coins = [10]

**Output:** 1

**Note:**

You can assume that

* 0 <= amount <= 5000
* 1 <= coin <= 5000
* the number of coins is less than 500
* the answer is guaranteed to fit into signed 32-bit integer

**Solution:**

class Solution {

public:

int change(int amount, vector<int>& coins) {

int n=coins.size();

int i,j,t[n+1][amount+1];

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

for(j=0;j<=amount;j++){

if(i==0)

t[i][j]=0;

if(j==0)

t[i][j]=1;

}

}

for(i=1;i<=n;i++){

for(j=1;j<=amount;j++){

if(coins[i-1]<=j)

t[i][j]=t[i][j-coins[i-1]]+t[i-1][j];

else

t[i][j]=t[i-1][j];

}

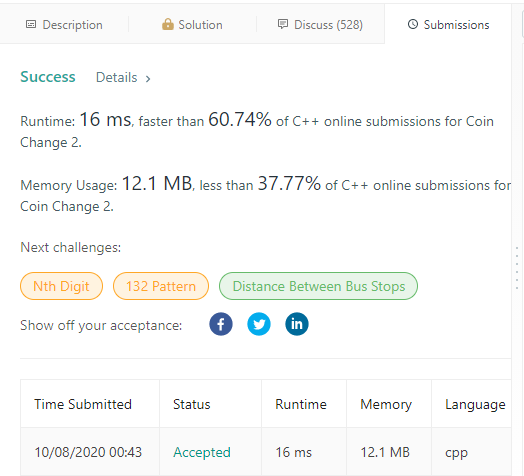
}

return t[n][amount];

}

};

**OUTPUT-**



**Question 4 –** **Write a program to print the set of coins in minimum number of coins problem.**

**Solution:**

//Coin Change-2 -> Minimum number of coins

/\*Input: coins[] = {25, 10, 5}, V = 30

Output: Minimum 2 coins required

We can use one coin of 25 cents and one of 5 cents

Input: coins[] = {9, 6, 5, 1}, V = 11

Output: Minimum 2 coins required

We can use one coin of 6 cents and 1 coin of 5 cents\*/

#include<bits/stdc++.h>

using namespace std;

int minNoOfCoins(int coins[],int sum,int n){

int i,j;

int t[n+1][sum+1];

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

for(j=0;j<=sum;j++){

if(i==0)

t[i][j]=INT\_MAX-1;

if(j==0)

t[i][j]=0;

}

}

t[0][0]=INT\_MAX-1;

//Initialize second row

for(j=1;j<=sum;j++){

if(j%coins[0]==0)

t[1][j]=j/coins[0];

else

t[1][j]=INT\_MAX-1;

}

for(i=2;i<=n;i++){

for(j=1;j<=sum;j++){

if(coins[i-1]<=j)

t[i][j]=min((t[i][j-coins[i-1]]+1),t[i-1][j]);

else

t[i][j]=t[i-1][j];

}

}

i=n;

j=sum;

vector<int> result;

while(i>0&&j>0){

if((t[i][j]-t[i][j-coins[i-1]])==1){

result.push\_back(coins[i-1]);

j=j-coins[i-1];

}

else

i=i-1;

}

cout<<"Set of coins:\n";

for(i=0;i<result.size();i++)

cout<<result[i]<<" ";

cout<<"\n";

return t[n][sum];

}

int main(){

int coins[]={9, 6, 5, 1};

int sum=11;

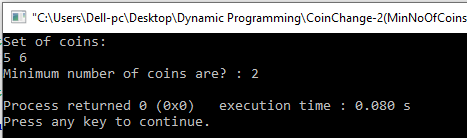
int n=sizeof(coins)/sizeof(coins[0]);

int numOfCoins=minNoOfCoins(coins,sum,n);

cout<<"Minimum number of coins are? : "<<numOfCoins<<"\n";

}

**OUTPUT-**



**Question 5 –** **Write a program to print the set of coins in maximum number of ways in coin change problem.**

**Solution:**

//Coin Change-1 -> Maximum number of ways

/\*Given a value N, if we want to make change for N cents, and we have infinite supply of each of S = { S1, S2, .. , Sm} valued coins,

how many ways can we make the change? The order of coins doesn’t matter.

For example, for N = 4 and S = {1,2,3}, there are four solutions: {1,1,1,1},{1,1,2},{2,2},{1,3}. So output should be 4. For N = 10 and S = {2, 5, 3, 6},

there are five solutions: {2,2,2,2,2}, {2,2,3,3}, {2,2,6}, {2,3,5} and {5,5}. So the output should be 5.\*/

#include<bits/stdc++.h>

using namespace std;

int maxNoOfWays(int coins[],int sum,int n){

int i,j;

int t[n+1][sum+1];

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

for(j=0;j<=sum;j++){

if(i==0)

t[i][j]=0;

if(j==0)

t[i][j]=1;

}

}

for(i=1;i<=n;i++){

for(j=1;j<=sum;j++){

if(coins[i-1]<=j)

t[i][j]=t[i][j-coins[i-1]]+t[i-1][j];

else

t[i][j]=t[i-1][j];

}

}

return t[n][sum];

}

void printAll(int coins[],int sum,int n){

vector<int> v;

vector<vector<int>> result;

vector<int> temp;

int i,j,freq,curr=0;

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

freq=sum/coins[i];

v.insert(v.end(),freq,coins[i]);

}

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

int l=v.size();

for(i=0;i<l;i++){

curr=0;

temp.clear();

for(j=i;j<l;j++){

if(curr<sum){

curr=curr+v[j];

temp.push\_back(v[j]);

if(curr==sum){

result.push\_back(temp);

break;

}

}else if(curr>sum){

break;

}

}

}

for(i=0;i<result.size();i++){

for(j=0;j<result[i].size();j++)

cout<<result[i][j]<<" ";

cout<<"\n";

}

}

int main(){

int coins[]={2, 1, 5};

int sum=5;

int n=sizeof(coins)/sizeof(coins[0]);

cout<<"Number of ways for change are? : "<<maxNoOfWays(coins,sum,n)<<"\n";

printAll(coins,sum,n);

}

**OUTPUT-**

