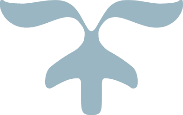


DAA WEEK – 10 SKILL – 10



# [The Coin Change Problem](https://www.hackerrank.com/contests/daa-skill-10-dynamic-programming-part-i/challenges/coin-change)

#include <stdio.h>

#include <stdlib.h>

long NCoinWay(int coin[], int n, int cost) {

long ar[n + 1][cost + 1];

for (int j = 0; j <= cost; j++) {

ar[0][j] = 0;

}

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

ar[i][0] = 1;

}

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

for (int j = 1; j <= cost; j++) {

if (coin[i - 1] > j) {

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

} else {

ar[i][j] = ar[i - 1][j] + ar[i][j - coin[i - 1]];

}

}

}

return ar[n][cost];

}

int main() {

int cost, n;

scanf("%d %d", &cost, &n);

int \*coin = (int \*)malloc(n \* sizeof(int));

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

scanf("%d", &coin[i]);

}

printf("%ld\n", NCoinWay(coin, n, cost));

free(coin);

return 0;

}

**The Coin Change Problem Test Cases**

**A screenshot of a computer

AI-generated content may be incorrect.**

# [Equal](https://www.hackerrank.com/contests/daa-skill-10-dynamic-programming-part-i/challenges/equal)

#include <stdio.h>

#include <stdlib.h>

#include <limits.h>

void solve() {

int T;

scanf("%d", &T);

while (T-- > 0) {

short N;

scanf("%hd", &N);

short A[N];

for (short i = 0; i < N; ++i) {

scanf("%hd", &A[i]);

}

if (N < 2) {

printf("0\n");

continue;

}

int minVal = A[0];

for (int i = 1; i < N; ++i) {

if (A[i] < minVal) {

minVal = A[i];

}

}

int minCount = INT\_MAX;

for (int i = 0; i <= 5; ++i) {

int count = 0;

for (short j = 0; j < N; ++j) {

short V = (short)(A[j] - (minVal - i));

count += V / 5 + (V % 5) / 2 + (V % 5) % 2;

}

if (count < minCount) {

minCount = count;

}

}

printf("%d\n", minCount);

}

}

int main() {

solve();

return 0;

}

**Equal Test Cases**

**A screenshot of a computer

AI-generated content may be incorrect.**

# [Sherlock and Cost](https://www.hackerrank.com/contests/daa-skill-10-dynamic-programming-part-i/challenges/sherlock-and-cost)

#include <stdio.h>

#include <stdlib.h>

#include <math.h>

void compute() {

int testCases;

scanf("%d", &testCases);

while (testCases-- > 0) {

int size;

scanf("%d", &size);

long \*array = (long \*)malloc(size \* sizeof(long));

for (int index = 0; index < size; index++) {

scanf("%ld", &array[index]);

}

long lowCost = 0;

long highCost = 0;

long totalCost = 0;

for (int index = 1; index < size; index++) {

long lowDifference = fmax((lowCost + 0), (highCost + labs(array[index - 1] - 1)));

long highDifference = fmax((lowCost + labs(1 - array[index])), (highCost + labs(array[index - 1] - array[index])));

totalCost = fmax(lowDifference, highDifference);

lowCost = lowDifference;

highCost = highDifference;

}

printf("%ld\n", totalCost);

free(array);

}

}

int main() {

compute();

return 0;

}

**Sherlock and Cost Test Cases**

**A screenshot of a computer

AI-generated content may be incorrect.**

# [Construct the Array](https://www.hackerrank.com/contests/daa-skill-10-dynamic-programming-part-i/challenges/construct-the-array)

#include <stdio.h>

#include <stdint.h>

long countArray(int n, int k, int x) {

int64\_t eq\_x = (x == 1), neq\_x = (x != 1), MOD = 1e9+7;

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

int64\_t new\_eq\_x = neq\_x;

neq\_x = ((k-1) \* eq\_x + (k-2) \* neq\_x) % MOD;

eq\_x = new\_eq\_x % MOD;

}

return eq\_x;

}

int main() {

int n, k, x;

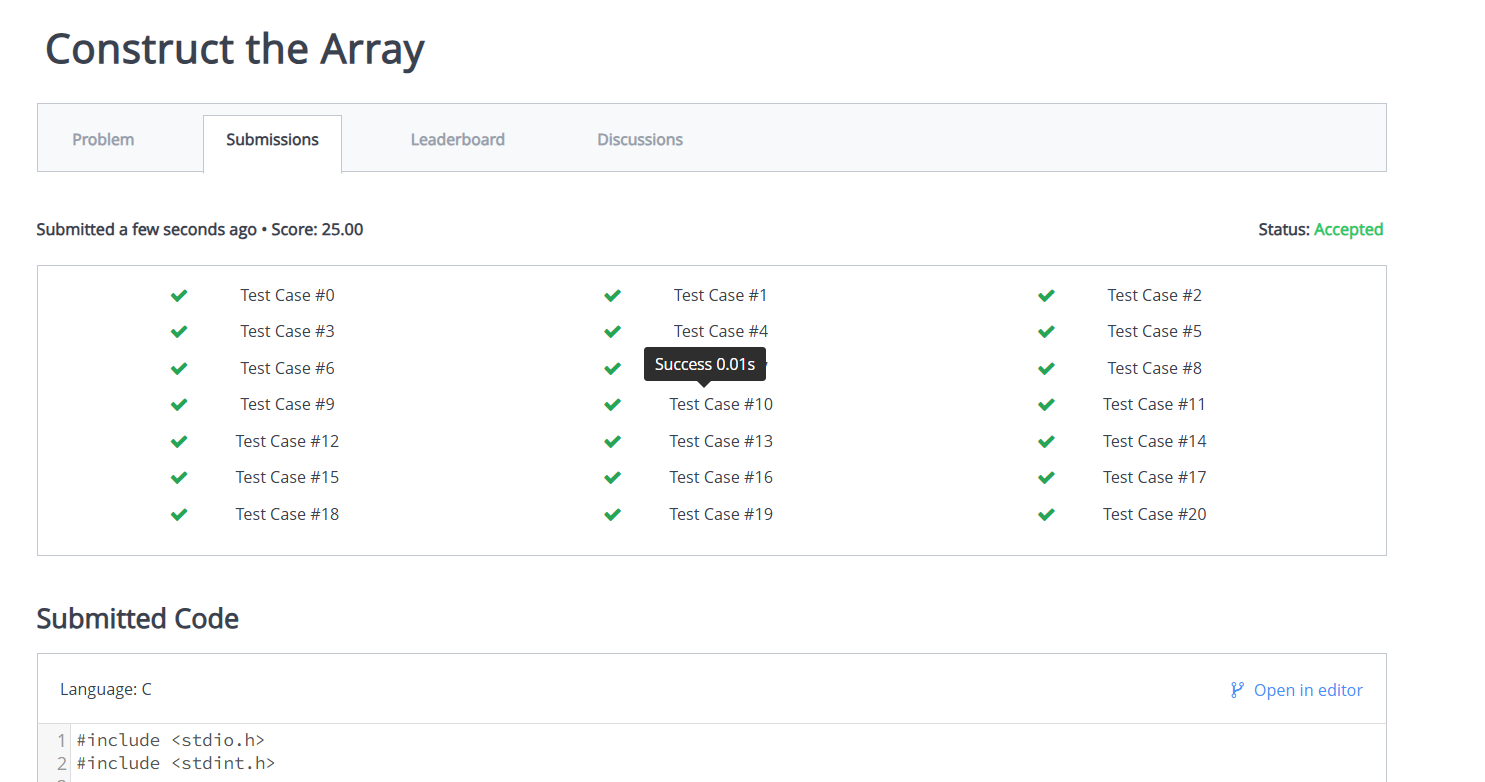
scanf("%d %d %d", &n, &k, &x);

printf("%ld\n", countArray(n, k, x));

return 0;

}

**Construct the Array Test Cases**

****

**SKILL WEEK – 10**

[**https://www.hackerrank.com/contests/daa-skill-10-dynamic-programming-part-i/challenges**](https://www.hackerrank.com/contests/daa-skill-10-dynamic-programming-part-i/challenges)