

Experiment-3.1

Student Name: Mukund Jaiswal UID: 21BCS3407

Branch: CSE Section/Group: 602-B

Semester: 5th Date of Performance: 19/10/23 Subject Name: AP-I Subject Code: 21CSP-314

<u>Aim</u>- Dynamic Programming: Implement the problems based on Dynamic Programming

<u>**Objectives**</u>- The objective of this experiment is to understand the concept of dynamic programming problems.

Problem1: https://www.hackerrank.com/challenges/construct-the-array/problem?isFullScreen=true

Problem2: https://www.hackerrank.com/challenges/equal/problem?isFullScreen=true

Description-

Dynamic programming is a powerful technique in computer science and mathematics for solving problems by breaking them down into smaller subproblems and solving each subproblem only once, storing the results for future reference.

The definition of dynamic programming says that it is a technique for solving a complex problem by first breaking into a collection of simpler subproblems, solving each subproblem just once, and then storing their solutions to avoid repetitive computations.

Code:

1.

Discover, Learn, Empower,



```
Change Theme
                                            Language C++14
                                                                                 0
    #include <bits/stdc++.h>
    using namespace std;
    string ltrim(const string &);
    string rtrim(const string &);
 7
    vector<string> split(const string &);
 8
    const int MOD = 10000000007;
10 \vee long countArray(int n, int k, int x) {
         long ways[2][2];
      ways[0][0] = 1;
      ways[0][1] = 0;
14
       bool fillSecond = true;
15 🗸
      for (int i = 0; i < n-1; i++) {
        ways[fillSecond][0] = (ways[!fillSecond][1] * (k - 1)) % MOD;
        ways[fillSecond][1] = (ways[!fillSecond][1] * (k - 2) + ways[!fillSecond][0]) % MOD;
         fillSecond = !fillSecond;
       long answer;
22 🗸
      if (x == 1) {
        answer = (ways[fillSecond][1] * (k - 1)) % MOD;
24
       }
25 🗸
      else {
26
         answer = (ways[fillSecond][1] * (k - 2) + ways[fillSecond][0]) % MOD;
27
28
       return answer;
```

```
int main()
33 🗸 {
34
         ofstream fout(getenv("OUTPUT_PATH"));
         string first_multiple_input_temp;
         getline(cin, first_multiple_input_temp);
         vector<string> first_multiple_input = split(rtrim(first_multiple_input_temp));
40
41
         int n = stoi(first_multiple_input[0]);
42
43
         int k = stoi(first_multiple_input[1]);
44
45
         int x = stoi(first_multiple_input[2]);
46
47
         long answer = countArray(n, k, x);
48
         fout << answer << "\n";</pre>
49
         fout.close();
         return 0;
56 > string ltrim(const string &str) {...
67 > string rtrim(const string &str) { ···
```

Discover, Learn, Empower,



```
78 ∨ vector<string> split(const string &str) {
        vector<string> tokens;
81
         string::size_type start = 0;
82
         string::size_type end = 0;
         while ((end = str.find(" ", start)) != string::npos) {
84 🗸
85
             tokens.push_back(str.substr(start, end - start));
86
87
             start = end + 1;
88
89
         tokens.push_back(str.substr(start));
         return tokens;
```

<u>2.</u>

```
20
                                Change Theme Language C++14
                                                                                 1
    #include <bits/stdc++.h>
3
   using namespace std;
5
   string ltrim(const string &);
6
   string rtrim(const string &);
7
    vector<string> split(const string &);
8
9
    int equal(vector<int> arr) {
10 \vee int ans = INT_MAX;
      int min_elem = *min_element(arr.begin(),arr.end());
      for(int base=0; base<3; base++){</pre>
12 🗸
13
         int temp = 0;
14 V
         for(int i=0;i<arr.size();i++){</pre>
           int dist = arr[i] - (min_elem - base);
            int steps = dist/5 + (dist%5)/2 + (dist%5)%2;
            temp+=steps;
19
        ans = min(ans, temp);
20
      return ans;
    int main()
24
25 🗸 {
         ofstream fout(getenv("OUTPUT_PATH"));
27
28
         string t_temp;
         getline(cin, t_temp);
```

Discover, Learn, Empower.



```
int t = stoi(ltrim(rtrim(t_temp)));
         for (int t_itr = 0; t_itr < t; t_itr++) {...</pre>
         fout.close();
         return 0;
62 ∨ string ltrim(const string &str) {
63
         string s(str);
64
65 V
         s.erase(
66
             s.begin(),
67
             find_if(s.begin(), s.end(), not1(ptr_fun<int, int>(isspace)))
         );
70
         return s;
73 ∨ string rtrim(const string &str) {
74
         string s(str);
76 V
         s.erase(
             find_if(s.rbegin(), s.rend(), not1(ptr_fun<int, int>(isspace))).base(),
         );
81
         return s;
```

```
71
    }
73 ∨ string rtrim(const string &str) {
74
         string s(str);
76 ∨
         s.erase(
77
             find_if(s.rbegin(), s.rend(), not1(ptr_fun<int, int>(isspace))).base(),
78
             s.end()
79
         );
81
         return s;
84 ∨ vector<string> split(const string &str) {
85
         vector<string> tokens;
87
         string::size_type start = 0;
         string::size_type end = 0;
90 🗸
         while ((end = str.find(" ", start)) != string::npos) {
             tokens.push_back(str.substr(start, end - start));
             start = end + 1;
         }
         tokens.push_back(str.substr(start));
         return tokens;
```

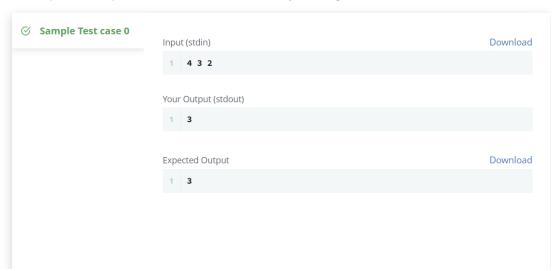


Outcome-

Problem 1 outcome

Congratulations!

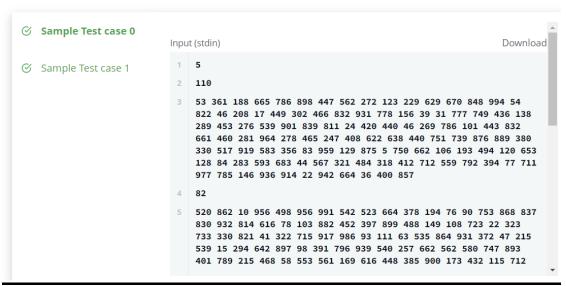
You have passed the sample test cases. Click the submit button to run your code against all the test cases.



Problem 2 Outcome

Congratulations!

You have passed the sample test cases. Click the submit button to run your code against all the test cases.



Learning Outcomes-

- 1. Learnt about the dynamic programming.
- 2. Learnt about the some dynamic programming problem.