Code : 1  
package Topic\_20\_DynamicProgramming;

import java.util.Scanner;

public class A\_Fibonaccidp {

public static void main(String[] args) throws Exception {

Scanner scn = new Scanner(System.in);

int n = scn.nextInt();

int fib = Fib(n);

System.out.println(fib);

scn.close();

}

public static int Fib(int n) {

if (n == 0 || n == 1) {

return n;

}

int fibn1 = Fib(n - 1);

int fibn2 = Fib(n - 2);

int fibn = fibn1 + fibn2;

return fibn;

}

}

Code : 2  
package Topic\_20\_DynamicProgramming;

import java.util.Scanner;

public class B\_ClimbStairs {

public static void main(String[] args) throws Exception {

Scanner scn = new Scanner(System.in);

int n = scn.nextInt();

int paths = countPathsTab(n);

System.out.println(paths);

scn.close();

}

public static int countPathsTab(int n) {

if (n == 0) {

return 1;

} else if (n < 0) {

return 0;

}

int[] dp = new int[n + 1];

dp[0] = 1;

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

dp[i] = dp[i - 1];

if (i >= 2)

dp[i] += dp[i - 2];

if (i >= 3)

dp[i] += dp[i - 3];

}

return dp[n];

}

}

Code : 3  
package Topic\_20\_DynamicProgramming;

import java.util.Scanner;

public class C\_ClimbStairsWithVariableJumps {

public static void main(String[] args) throws Exception {

Scanner scn = new Scanner(System.in);

int n = scn.nextInt();

int[] arr = new int[n];

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

arr[i] = scn.nextInt();

}

int[] dp = new int[n + 1];

dp[n] = 1;

for (int i = n - 1; i >= 0; i--) {

for (int reach = i + 1; reach <= Math.min(n, i + arr[i]); reach++) {

dp[i] += dp[reach];

}

}

System.out.println(dp[0]);

scn.close();

}

}

Code : 4  
package Topic\_20\_DynamicProgramming;

import java.util.Scanner;

public class D\_ClimbStairsWithMinimumMoves {

public static void main(String[] args) throws Exception {

Scanner scn = new Scanner(System.in);

int n = scn.nextInt();

int[] arr = new int[n];

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

arr[i] = scn.nextInt();

}

Integer[] dp = new Integer[n + 1];

dp[n] = 0;

for (int i = n - 1; i >= 0; i--) {

if (arr[i] == 0)

continue;

int min = Integer.MAX\_VALUE;

for (int j = 1; j <= arr[i] && i + j < dp.length; j++) {

if (dp[i + j] != null) {

min = Math.min(min, dp[i + j]);

}

}

if (min != Integer.MAX\_VALUE)

dp[i] = min + 1;

}

System.out.println(dp[0]);

scn.close();

}

}

Code : 5  
package Topic\_20\_DynamicProgramming;

import java.util.Scanner;

public class E\_MinCostInMazeTraversal {

public static void main(String[] args) throws Exception {

Scanner scn = new Scanner(System.in);

int n = scn.nextInt();

int m = scn.nextInt();

int[][] arr = new int[n][m];

for (int i = 0; i < n; i++) { //1

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

arr[i][j] = scn.nextInt();

}

}

int[][] dp = new int[n][m]; //2

for (int i = n - 1; i >= 0; i--) { //3

for (int j = m - 1; j >= 0; j--) { //4

if (i == n - 1 && j == m - 1) { //5

dp[i][j] = arr[i][j];

} else if (i == n - 1) { //6

dp[i][j] = arr[i][j] + dp[i][j + 1];

} else if (j == m - 1) { //7

dp[i][j] = arr[i][j] + dp[i + 1][j];

} else { //8

int min = Math.min(dp[i + 1][j], dp[i][j + 1]);

dp[i][j] = arr[i][j] + min;

}

}

}

System.out.println(dp[0][0]); //9

}

}

Code : 6  
package Topic\_20\_DynamicProgramming;

import java.util.Scanner;

public class F\_Goldmine {

public static void main(String[] args) throws Exception {

Scanner scn = new Scanner(System.in);

int n = scn.nextInt();

int m = scn.nextInt();

int[][] arr = new int[n][m];

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

for (int j = 0; j < m; j++) { //1

arr[i][j] = scn.nextInt();

}

}

int[][] dp = new int[n][m]; //2

for (int j = m - 1; j >= 0; j--) { //3

for (int i = n - 1; i >= 0; i--) { //4

if (j == m - 1) {

dp[i][j] = arr[i][j]; //5

} else if (i == 0) { //6

dp[i][j] = arr[i][j] + Math.max(dp[i][j + 1], dp[i + 1][j + 1]);

} else if (i == n - 1) { //7

dp[i][j] = arr[i][j] + Math.max(dp[i][j + 1], dp[i - 1][j + 1]);

} else { //8

dp[i][j] = arr[i][j] + Math.max(dp[i][j + 1], Math.max(dp[i + 1][j + 1],

dp[i - 1][j + 1]));

}

}

}

int max = dp[0][0]; //9

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

if (dp[i][0] > max) //10

max = dp[i][0];

}

System.out.println(max); //11

}

}

Code : 7  
package Topic\_20\_DynamicProgramming;

import java.io.BufferedReader;

import java.io.InputStreamReader;

public class G\_TargetSumSubsetsDp {

public static void main(String[] args) throws Exception {

BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

int n = Integer.parseInt(br.readLine());

int[] arr = new int[n];

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

arr[i] = Integer.parseInt(br.readLine());

}

int tar = Integer.parseInt(br.readLine());

boolean[][] dp = new boolean[arr.length + 1][tar + 1];

for (int i = 0; i < dp.length; i++) {

for (int j = 0; j < dp[0].length; j++) {

if (i == 0 && j == 0) {

dp[i][j] = true;

} else if (i == 0) {

dp[i][j] = false;

} else if (j == 0) {

dp[i][j] = true;

} else {

if (dp[i - 1][j] == true) {

dp[i][j] = true;

} else {

int val = arr[i - 1];

if (j >= val

&& dp[i - 1][j - val] == true) {

dp[i][j] = true;

}

}

}

}

}

System.out.println(dp[dp.length - 1][tar]);

}

}

Code : 8  
package Topic\_20\_DynamicProgramming;

import java.io.BufferedReader;

import java.io.InputStreamReader;

public class H\_CoinChangeCombination {

public static void main(String[] args) throws Exception {

BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

int n = Integer.parseInt(br.readLine());

int[] coins = new int[n];

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

coins[i] = Integer.parseInt(br.readLine());

}

int amt = Integer.parseInt(br.readLine());

int[] dp = new int[amt + 1];

dp[0] = 1;

for (int coin : coins) {

for (int i = 1; i < dp.length; i++) {

if (i >= coin) {

dp[i] += dp[i - coin];

}

}

}

System.out.println(dp[amt]);

}

}

Code : 9  
package Topic\_20\_DynamicProgramming;

import java.io.BufferedReader;

import java.io.InputStreamReader;

public class I\_CoinChangePermutations {

public static void main(String[] args) throws Exception {

BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

int n = Integer.parseInt(br.readLine());

int[] coins = new int[n];

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

coins[i] = Integer.parseInt(br.readLine());

}

int amt = Integer.parseInt(br.readLine());

int[] dp = new int[amt + 1];

dp[0] = 1;

for (int i = 1; i < dp.length; i++) {

for (int coin : coins) {

if (i >= coin) {

dp[i] += dp[i - coin];

}

}

}

System.out.println(dp[amt]);

}

}

Code : 10  
package Topic\_20\_DynamicProgramming;

import java.io.BufferedReader;

import java.io.InputStreamReader;

public class J\_ZeroOneKnapsack {

public static void main(String[] args) throws Exception {

BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

int n = Integer.parseInt(br.readLine());

int[] price = new int[n];

String str1 = br.readLine();

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

price[i] = Integer.parseInt(str1.split(" ")[i]);

}

int[] weight = new int[n];

String str2 = br.readLine();

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

weight[i] = Integer.parseInt(str2.split(" ")[i]);

}

int cap = Integer.parseInt(br.readLine());

int[][] dp = new int[n + 1][cap + 1];

for (int i = 1; i < dp.length; i++) {

for (int j = 1; j < dp[0].length; j++) {

int val = price[i - 1];

int wt = weight[i - 1];

if (j >= wt) { //If the current capacity is greater than the weight of the current item

dp[i][j] = Math.max(dp[i - 1][j], dp[i - 1][j - wt] + val); // max cost will be max of cost before putting the item and after putting it

} else {

dp[i][j] = dp[i - 1][j]; //If current capacity is less than weight do not add item to the bag

}

}

}

System.out.println(dp[n][cap]);

}

}

Code : 11  
package Topic\_20\_DynamicProgramming;

import java.util.Arrays;

import java.util.Scanner;

public class K\_FractionalKnapsackOfficial {

static class Item implements Comparable<Item> {

int val;

int wt;

double vwRatio;

public int compareTo(Item o) {

if (this.vwRatio < o.vwRatio) {

return -1;

} else if (this.vwRatio > o.vwRatio) {

return +1;

} else {

return 0;

}

}

}

public static void main(String[] args) {

Scanner scn = new Scanner(System.in);

int n = scn.nextInt();

Item[] items = new Item[n];

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

items[i] = new Item();

items[i].val = scn.nextInt();

}

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

items[i].wt = scn.nextInt();

items[i].vwRatio = items[i].val \* 1.0 / items[i].wt;

}

Arrays.sort(items);

double vib = 0;

int space = scn.nextInt();

int i = n - 1;

while (space > 0 && i >= 0) {

if (space >= items[i].wt) {

vib = vib + items[i].val;

space = space - items[i].wt;

} else {

vib = vib + (space \* items[i].val \* 1.0) / items[i].wt;

space = 0;

break;

}

i--;

}

System.out.println(vib);

}

}

Code : 12  
package Topic\_20\_DynamicProgramming;

import java.util.Scanner;

public class L\_UnboundedKnapsack {

public static void main(String[] args) throws Exception {

Scanner scn = new Scanner(System.in);

int n = scn.nextInt();

int[] values = new int[n];

int[] wts = new int[n];

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

values[i] = scn.nextInt();

}

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

wts[i] = scn.nextInt();

}

int cap = scn.nextInt();

int[] dp = new int[cap + 1];

dp[0] = 0;

for (int bagc = 1; bagc <= cap; bagc++) {

int max = 0;

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

if (wts[i] <= bagc) {

int rbagc = bagc - wts[i];

int rbagv = dp[rbagc];

int tbagv = rbagv + values[i];

if (tbagv > max) {

max = tbagv;

}

}

}

dp[bagc] = max;

}

System.out.println(dp[cap]);

}

}

Code : 13  
package Topic\_20\_DynamicProgramming;

import java.util.Scanner;

public class M\_CountBinaryStrings {

public static void main(String[] args) throws Exception {

Scanner scn = new Scanner(System.in);

int n = scn.nextInt();

int zeroes = 1;

int ones = 1;

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

int nzeroes = ones;

int nones = ones + zeroes;

zeroes = nzeroes;

ones = nones;

}

System.out.println(zeroes + ones);

}

}

Code : 14  
package Topic\_20\_DynamicProgramming;

import java.util.Scanner;

public class N\_CountEncodings {

public static void main(String[] args) throws Exception {

Scanner scn = new Scanner(System.in);

String str = scn.next();

int[] dp = new int[str.length()];

dp[0] = 1;

for (int i = 1; i < str.length(); i++) {

if (str.charAt(i - 1) == '0' && str.charAt(i) == '0') {

dp[i] = 0;

} else if (str.charAt(i - 1) == '0' && str.charAt(i) != '0') {

dp[i] = dp[i - 1];

} else if (str.charAt(i - 1) != '0' && str.charAt(i) == '0') {

if (str.charAt(i - 1) == '1' || str.charAt(i - 1) == '2') {

dp[i] = (i >= 2 ? dp[i - 2] : 1);

}

} else {

dp[i] = dp[i - 1];

if (Integer.parseInt(str.substring(i - 1, i + 1)) <= 26) {

dp[i] += (i >= 2 ? dp[i - 2] : 1);

}

}

}

System.out.println(dp[str.length() - 1]);

scn.close();

}

}

Code : 15  
package Topic\_20\_DynamicProgramming;

import java.util.Scanner;

public class O\_Count\_APlus\_BPlus\_CPlusSubsequences {

public static void main(String[] args) throws Exception {

Scanner scn = new Scanner(System.in);

String str = scn.next();

int a = 0;

int ab = 0;

int abc = 0;

for (int i = 0; i < str.length(); i++) {

char ch = str.charAt(i);

if (ch == 'a') {

a = 2 \* a + 1;

} else if (ch == 'b') {

ab = 2 \* ab + a;

} else if (ch == 'c') {

abc = 2 \* abc + ab;

}

}

System.out.println(abc);

scn.close();

}

}

Code : 16  
package Topic\_20\_DynamicProgramming;

import java.util.Scanner;

// Java code to Count Palindromic Subsequence

// in a given String

public class P\_CountPalindromicSubsequences {

// Function return the total palindromic

// subsequence

static int countPS(String str) {

int N = str.length();

// create a 2D array to store the count

// of palindromic subsequence

int[][] cps = new int[N][N];

// palindromic subsequence of length 1

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

cps[i][i] = 1;

// check subsequence of length L is

// palindrome or not

for (int L = 2; L <= N; L++) {

for (int i = 0; i <= N - L; i++) {

int k = L + i - 1;

if (str.charAt(i) == str.charAt(k)) {

cps[i][k] = cps[i][k - 1]

+ cps[i + 1][k] + 1;

} else {

cps[i][k] = cps[i][k - 1]

+ cps[i + 1][k]

- cps[i + 1][k - 1];

}

}

}

// return total palindromic subsequence

return cps[0][N - 1];

}

// Driver program

public static void main(String args[]) {

Scanner sc = new Scanner(System.in);

String str = sc.nextLine();

System.out.println(countPS(str));

}

}

// This code is contributed by Sumit Ghosh

Code : 17  
import java.util.Scanner;

public class Q\_CountPalindromicSubstrings {

public static void main(String[] args) throws Exception {

Scanner scn = new Scanner(System.in);

String s = scn.nextLine();

boolean[][] dp = new boolean[s.length()][s.length()];

int count = 0;

for (int g = 0; g < s.length(); g++) {

for (int i = 0, j = g; j < dp.length; i++, j++) {

if (g == 0) {

dp[i][j] = true;

} else if (g == 1) {

if (s.charAt(i) == s.charAt(j)) {

dp[i][j] = true;

} else {

dp[i][j] = false;

}

} else {

if (s.charAt(i) == s.charAt(j) && dp[i + 1][j - 1] == true) {

dp[i][j] = true;

} else {

dp[i][j] = false;

}

}

if (dp[i][j]) {

count++;

}

}

}

System.out.println(count);

}

}

Code : 18  
package Topic\_20\_DynamicProgramming;

import java.io.BufferedReader;

import java.io.InputStreamReader;

public class R\_CountOfValleysAndMountains {

public static void main(String[] args) throws Exception {

BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

int n = Integer.parseInt(br.readLine());

long[] dp = new long[n + 1];

dp[0] = 1;

for (int i = 1; i < dp.length; i++) {

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

dp[i] += dp[j] \* dp[i - 1 - j];

}

}

System.out.println(dp[n]);

}

}

Code : 19  
package Topic\_20\_DynamicProgramming;

import java.io.BufferedReader;

import java.io.InputStreamReader;

public class S\_CountBrackets {

public static void main(String[] args) throws Exception {

BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

int n = Integer.parseInt(br.readLine());

long[] dp = new long[n + 1];

dp[0] = 1;

for (int i = 1; i < dp.length; i++) {

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

dp[i] += dp[j] \* dp[i - 1 - j];

}

}

System.out.println(dp[n]);

}

}

Code : 20  
package Topic\_20\_DynamicProgramming;

import java.util.Scanner;

public class T\_ArrangeBuildings {

public static void main(String[] args) throws Exception {

Scanner scn = new Scanner(System.in);

long n = scn.nextInt();

long ob = 1;

long os = 1;

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

long nb = os;

long ns = os + ob;

ob = nb;

os = ns;

}

long total = ob + os;

System.out.println(total \* total);

}

}

Code : 21  
package Topic\_20\_DynamicProgramming;

import java.util.Scanner;

public class U\_MaximumSumNonAdjacentElements {

public static void main(String[] args) throws Exception {

Scanner scn = new Scanner(System.in);

int n = scn.nextInt();

int[] arr = new int[n];

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

arr[i] = scn.nextInt();

}

int inc = arr[0]; //1

int exc = 0;

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

int ninc = exc + arr[i]; //2

int nexc = Math.max(inc, exc);

inc = ninc; //3

exc = nexc;

}

System.out.println(Math.max(inc, exc)); //4

}

}

Code : 22  
package Topic\_20\_DynamicProgramming;

import java.io.BufferedReader;

import java.io.InputStreamReader;

public class V\_MaximumSumIncreasingSubsequence {

public static void main(String[] args) throws Exception {

BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

int n = Integer.parseInt(br.readLine());

int[] arr = new int[n];

for (int i = 0; i < arr.length; i++) {

arr[i] = Integer.parseInt(br.readLine());

}

int omax = Integer.MIN\_VALUE;

int[] dp = new int[arr.length];

for (int i = 0; i < arr.length; i++) {

Integer max = null;

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

if (arr[j] <= arr[i]) {

if (max == null || dp[j] > max) {

max = dp[j];

}

}

}

if (max != null) {

dp[i] = max + arr[i];

} else {

dp[i] = arr[i];

}

if (dp[i] > omax) {

omax = dp[i];

}

}

System.out.println(omax);

}

}

Code : 23  
package Topic\_20\_DynamicProgramming;

import java.io.BufferedReader;

import java.io.InputStreamReader;

import java.util.Arrays;

public class W\_MaximumNonoverlappingBridges {

public static class Bridge implements Comparable<Bridge> {

int n;

int s;

@Override

public int compareTo(Bridge o) {

if (this.n != o.n) {

return this.n - o.n;

} else {

return this.s - o.s;

}

}

}

public static void main(String[] args) throws Exception {

BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

int n = Integer.parseInt(br.readLine());

Bridge[] brdgs = new Bridge[n];

for (int i = 0; i < brdgs.length; i++) {

String str = br.readLine();

brdgs[i] = new Bridge();

brdgs[i].n = Integer.parseInt(str.split(" ")[0]);

brdgs[i].s = Integer.parseInt(str.split(" ")[1]);

}

Arrays.sort(brdgs);

int[] lis = new int[brdgs.length];

for (int i = 0; i < brdgs.length; i++) {

Integer max = null;

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

if (brdgs[j].s <= brdgs[i].s) {

if (max == null || lis[j] > max) {

max = lis[j];

}

}

}

if (max != null) {

lis[i] = max + 1;

} else {

lis[i] = 1;

}

}

int omax = 0;

for (int i = 0; i < brdgs.length; i++) {

if (lis[i] > omax) {

omax = lis[i];

}

}

System.out.println(omax);

}

}

Code : 24  
package Topic\_20\_DynamicProgramming;

import java.util.Scanner;

public class X\_PaintHouse {

public static void main(String[] args) throws Exception {

Scanner scn = new Scanner(System.in);

int n = scn.nextInt();

int[][] arr = new int[n][3];

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

for (int j = 0; j < 3; j++) { //1

arr[i][j] = scn.nextInt();

}

}

int[][] dp = new int[n][3]; //2

dp[0][0] = arr[0][0];

dp[0][1] = arr[0][1]; //3

dp[0][2] = arr[0][2];

for (int i = 1; i < n; i++) { //4

dp[i][0] = Math.min(dp[i - 1][1], dp[i - 1][2]) + arr[i][0];

dp[i][1] = Math.min(dp[i - 1][0], dp[i - 1][2]) + arr[i][1];

dp[i][2] = Math.min(dp[i - 1][1], dp[i - 1][0]) + arr[i][2];

}

System.out.println(Math.min(dp[n - 1][0], (Math.min(dp[n - 1][1], dp[n - 1][2]))));

//5

}

}

Code : 25  
package Topic\_20\_DynamicProgramming;

import java.util.Scanner;

public class YA\_PaintHouseManyColors {

public static void main(String[] args) throws Exception {

Scanner scn = new Scanner(System.in);

int n = scn.nextInt();

int c = scn.nextInt();

int[][] arr = new int[n][c];

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

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

arr[i][j] = scn.nextInt();

}

}

int[][] dp = new int[arr.length][arr[0].length];

for (int j = 0; j < arr[0].length; j++) {

dp[0][j] = arr[0][j];

}

int least = Integer.MAX\_VALUE;

int sleast = Integer.MAX\_VALUE;

for (int j = 0; j < arr[0].length; j++) {

dp[0][j] = arr[0][j];

if (arr[0][j] <= least) {

sleast = least;

least = arr[0][j];

} else if (arr[0][j] <= sleast) {

sleast = arr[0][j];

}

}

for (int i = 1; i < dp.length; i++) {

int nleast = Integer.MAX\_VALUE;

int nsleast = Integer.MAX\_VALUE;

for (int j = 0; j < dp[0].length; j++) {

if (least == dp[i - 1][j]) {

dp[i][j] = sleast + arr[i][j];

} else {

dp[i][j] = least + arr[i][j];

}

if (dp[i][j] <= nleast) {

nsleast = nleast;

nleast = dp[i][j];

} else if (dp[i][j] <= nsleast) {

nsleast = dp[i][j];

}

}

least = nleast;

sleast = nsleast;

}

System.out.println(least);

}

}

Code : 26  
package Topic\_20\_DynamicProgramming;

import java.util.Scanner;

public class Y\_PaintHouseManyColors {

public static void main(String[] args) throws Exception {

Scanner scn = new Scanner(System.in);

int n = scn.nextInt();

int c = scn.nextInt();

int[][] arr = new int[n][c];

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

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

arr[i][j] = scn.nextInt();

}

}

int[][] dp = new int[arr.length][arr[0].length];

for (int j = 0; j < arr[0].length; j++) {

dp[0][j] = arr[0][j];

}

for (int i = 1; i < dp.length; i++) {

for (int j = 0; j < dp[0].length; j++) {

int min = Integer.MAX\_VALUE;

for (int k = 0; k < dp[0].length; k++) {

if (k != j) {

if (dp[i - 1][k] < min) {

min = dp[i - 1][k];

}

}

}

dp[i][j] = arr[i][j] + min;

}

}

int min = Integer.MAX\_VALUE;

for (int k = 0; k < dp[0].length; k++) {

if (dp[n - 1][k] < min) {

min = dp[n - 1][k];

}

}

System.out.println(min);

}

}

Code : 27  
package Topic\_20\_DynamicProgramming;

import java.util.Scanner;

public class ZA\_TilingWith21Tiles {

public static void main(String[] args) throws Exception {

Scanner scn = new Scanner(System.in);

int n = scn.nextInt();

int[] dp = new int[n + 1];

dp[0] = 0;

dp[1] = 1;

dp[2] = 2;

for (int i = 3; i < dp.length; i++) {

dp[i] = dp[i - 1] + dp[i - 2];

}

System.out.println(dp[n]);

}

}

Code : 28  
package Topic\_20\_DynamicProgramming;

import java.io.BufferedReader;

import java.io.InputStreamReader;

public class ZB\_TilingWithM1Tiles {

public static void main(String[] args) throws Exception {

BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

int n = Integer.parseInt(br.readLine());

int m = Integer.parseInt(br.readLine());

int[] dp = new int[n + 1];

dp[1] = 1;

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

if (i < m) {

dp[i] = 1;

} else if (i == m) {

dp[i] = 2;

} else {

dp[i] = dp[i - 1] + dp[i - m];

}

}

System.out.println(dp[n]);

}

}

Code : 29  
package Topic\_20\_DynamicProgramming;

import java.util.Scanner;

public class ZC\_FriendsPairing {

public static void main(String[] args) throws Exception {

Scanner scn = new Scanner(System.in);

int n = scn.nextInt();

int[] dp = new int[n + 1];

dp[0] = 1;

dp[1] = 1;

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

dp[i] = dp[i - 1] + (i - 1) \* dp[i - 2];

}

System.out.println(dp[n]);

scn.close();

}

}

Code : 30  
package Topic\_20\_DynamicProgramming;

import java.util.Scanner;

public class ZD\_PartitionIntoSubsets {

public static void main(String[] args) throws Exception {

Scanner scn = new Scanner(System.in);

int n = scn.nextInt();

int k = scn.nextInt();

if (n == 0 || k == 0 || n < k) {

System.out.println(0);

scn.close();

return;

}

long[][] dp = new long[k + 1][n + 1];

for (int t = 1; t <= k; t++) {

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

if (p == t)

dp[t][p] = 1;

else if (p > t)

dp[t][p] = t \* dp[t][p - 1] + dp[t - 1][p - 1];

}

}

System.out.println(dp[k][n]);

scn.close();

}

}

Code : 31  
package Topic\_20\_DynamicProgramming;

import java.io.BufferedReader;

import java.io.InputStreamReader;

public class ZE\_BuyAndSellStocksOneTransactionAllowed {

public static void main(String[] args) throws Exception {

BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

int n = Integer.parseInt(br.readLine());

int[] arr = new int[n];

for (int i = 0; i < arr.length; i++) {

arr[i] = Integer.parseInt(br.readLine());

}

int msf = arr[0];

int op = 0;

for (int i = 1; i < arr.length; i++) {

if (arr[i] < msf) {

msf = arr[i];

}

int cp = arr[i] - msf;

if (cp > op) {

op = cp;

}

}

System.out.println(op);

}

}

Code : 32  
package Topic\_20\_DynamicProgramming;

import java.io.BufferedReader;

import java.io.InputStreamReader;

public class ZF\_BuyAndSellStocksInfiniteTransactionsAllowed {

public static void main(String[] args) throws Exception {

BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

int n = Integer.parseInt(br.readLine());

int[] arr = new int[n];

for (int i = 0; i < arr.length; i++) {

arr[i] = Integer.parseInt(br.readLine());

}

int bon = 0;

int son = 0;

int op = 0;

for (int i = 1; i < arr.length; i++) {

if (arr[i] < arr[i - 1]) {

op += arr[son] - arr[bon];

bon = son = i;

} else {

son++;

}

}

op += arr[son] - arr[bon];

System.out.println(op);

}

}

Code : 33  
package Topic\_20\_DynamicProgramming;

import java.io.BufferedReader;

import java.io.InputStreamReader;

public class ZG\_BuyAndSellStocksWithTransactionFeeInfiniteTransactionsAllowed {

public static void main(String[] args) throws Exception {

BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

int n = Integer.parseInt(br.readLine());

int[] arr = new int[n];

for (int i = 0; i < arr.length; i++) {

arr[i] = Integer.parseInt(br.readLine());

}

int fee = Integer.parseInt(br.readLine());

int bstp = -arr[0];

int sstp = 0;

for (int i = 1; i < arr.length; i++) {

int nsstp = 0;

int nbstp = 0;

if (sstp - arr[i] > bstp) {

nbstp = sstp - arr[i];

} else {

nbstp = bstp;

}

if (bstp + arr[i] - fee > sstp) {

nsstp = bstp + arr[i] - fee;

} else {

nsstp = sstp;

}

bstp = nbstp;

sstp = nsstp;

}

System.out.println(sstp);

}

}

Code : 34  
package Topic\_20\_DynamicProgramming;

import java.io.BufferedReader;

import java.io.InputStreamReader;

public class ZH\_BuyAndSellStocksWithCooldownInfiniteTransactionAllowed {

public static void main(String[] args) throws Exception {

BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

int n = Integer.parseInt(br.readLine());

int[] arr = new int[n];

for (int i = 0; i < arr.length; i++) {

arr[i] = Integer.parseInt(br.readLine());

}

int bstp = -arr[0];

int sstp = 0;

int cstp = 0;

for (int i = 1; i < arr.length; i++) {

int nbstp = 0;

int nsstp = 0;

int ncstp = 0;

if (cstp - arr[i] > bstp) {

nbstp = cstp - arr[i];

} else {

nbstp = bstp;

}

if (bstp + arr[i] > sstp) {

nsstp = bstp + arr[i];

} else {

nsstp = sstp;

}

if (sstp > cstp) {

ncstp = sstp;

} else {

ncstp = cstp;

}

bstp = nbstp;

sstp = nsstp;

cstp = ncstp;

}

System.out.println(Math.max(sstp, cstp));

}

}

Code : 35  
package Topic\_20\_DynamicProgramming;

import java.io.BufferedReader;

import java.io.InputStreamReader;

public class ZI\_BuyAndSellStocksTwoTransactionsAllowed {

public static void main(String[] args) throws Exception {

BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

int n = Integer.parseInt(br.readLine());

int[] arr = new int[n];

for (int i = 0; i < arr.length; i++) {

arr[i] = Integer.parseInt(br.readLine());

}

int misf = arr[0];

int[] ps = new int[arr.length];

for (int i = 1; i < arr.length; i++) {

if (arr[i] < misf) {

misf = arr[i];

}

if (arr[i] - misf > ps[i - 1]) {

ps[i] = arr[i] - misf;

} else {

ps[i] = ps[i - 1];

}

}

int masf = arr[arr.length - 1];

int[] pb = new int[arr.length];

for (int i = arr.length - 2; i >= 0; i--) {

if (arr[i] > masf) {

masf = arr[i];

}

if (masf - arr[i] > pb[i + 1]) {

pb[i] = masf - arr[i];

} else {

pb[i] = pb[i + 1];

}

}

int mp = Integer.MIN\_VALUE;

for (int i = 0; i < arr.length; i++) {

if (ps[i] + pb[i] > mp) {

mp = ps[i] + pb[i];

}

}

System.out.println(mp);

}

}

Code : 36  
package Topic\_20\_DynamicProgramming;

import java.util.Scanner;

public class ZJ\_BuyAndSellStocksKTransactionsAllowed {

public static void main(String[] args) throws Exception {

Scanner scn = new Scanner(System.in);

int n = scn.nextInt();

int[] arr = new int[n];

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

arr[i] = scn.nextInt();

}

int k = scn.nextInt();

int[][] dp = new int[k + 1][n];

for (int t = 1; t <= k; t++) {

for (int d = 1; d < arr.length; d++) {

int max = dp[t][d - 1];

for (int pd = 0; pd < d; pd++) {

int ptilltm1 = dp[t - 1][pd];

int ptth = arr[d] - arr[pd];

if (ptilltm1 + ptth > max) {

max = ptilltm1 + ptth;

}

}

dp[t][d] = max;

}

}

System.out.println(dp[k][n - 1]);

}

}

Code : 37  
package Topic\_20\_DynamicProgramming;

import java.util.Scanner;

public class ZK\_LongestIncreasingSubsequence {

public static void main(String[] args) throws Exception {

Scanner scn = new Scanner(System.in);

int n = scn.nextInt();

int[] arr = new int[n];

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

arr[i] = scn.nextInt();

}

int omax = 0;

int[] dp = new int[n];

for (int i = 0; i < dp.length; i++) {

int max = 0;

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

if (arr[j] < arr[i]) {

if (dp[j] > max) {

max = dp[j];

}

}

}

dp[i] = max + 1;

if (dp[i] > omax) {

omax = dp[i];

}

}

System.out.println(omax);

}

}

Code : 38  
package Topic\_20\_DynamicProgramming;

import java.util.Scanner;

public class ZL\_LongestBitonicSubsequence {

public static void main(String[] args) throws Exception {

Scanner scn = new Scanner(System.in);

int n = scn.nextInt();

int[] arr = new int[n];

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

arr[i] = scn.nextInt();

}

int omax = 0;

int[] dp = new int[n];

for (int i = 0; i < dp.length; i++) {

int max = 0;

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

if (arr[j] < arr[i]) {

if (dp[j] > max) {

max = dp[j];

}

}

}

dp[i] = max + 1;

if (dp[i] > omax) {

omax = dp[i];

}

}

System.out.println(omax);

}

}

Code : 39  
package Topic\_20\_DynamicProgramming;

import java.io.BufferedReader;

import java.io.InputStreamReader;

public class ZM\_LongestCommonSubsequence {

public static void main(String[] args) throws Exception {

BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

String str1 = br.readLine();

String str2 = br.readLine();

int[][] dp = new int[str1.length() + 1][str2.length() + 1];

for (int i = dp.length - 2; i >= 0; i--) {

for (int j = dp[0].length - 2; j >= 0; j--) {

if (str1.charAt(i) == str2.charAt(j)) {

dp[i][j] = 1 + dp[i + 1][j + 1];

} else {

dp[i][j] = Math.max(dp[i + 1][j], dp[i][j + 1]);

}

}

}

System.out.println(dp[0][0]);

}

}

Code : 40  
package Topic\_20\_DynamicProgramming;

import java.util.Scanner;

public class ZN\_LongestPalindromicSubsequences {

// A utility function to get max of two integers

static int max(int x, int y) {

return (x > y) ? x : y;

}

// Returns the length of the longest

// palindromic subsequence in seq

static int lps(String seq) {

int n = seq.length();

int i, j, cl;

// Create a table to store results of subproblems

int L[][] = new int[n][n];

// Strings of length 1 are palindrome of length 1

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

L[i][i] = 1;

// Build the table. Note that the lower

// diagonal values of table are

// useless and not filled in the process.

// The values are filled in a manner similar

// to Matrix Chain Multiplication DP solution (See

// https://www.geeksforgeeks.org/matrix-chain-multiplication-dp-8/).

// cl is length of substring

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

for (i = 0; i < n - cl + 1; i++) {

j = i + cl - 1;

if (seq.charAt(i) == seq.charAt(j) && cl == 2)

L[i][j] = 2;

else if (seq.charAt(i) == seq.charAt(j))

L[i][j] = L[i + 1][j - 1] + 2;

else

L[i][j] = max(L[i][j - 1], L[i + 1][j]);

}

}

return L[0][n - 1];

}

/\* Driver program to test above functions \*/

public static void main(String args[]) {

Scanner sc = new Scanner(System.in);

String str = sc.nextLine();

String seq = str;

int n = seq.length();

System.out.println(lps(seq));

}

}

Code : 41  
package Topic\_20\_DynamicProgramming;

import java.util.Scanner;

public class ZO\_LongestPalindromicSubstring {

public static void main(String[] args) throws Exception {

Scanner scn = new Scanner(System.in);

String str = scn.nextLine();

boolean[][] dp = new boolean[str.length()][str.length()];

int len = 0;

for (int g = 0; g < str.length(); g++) {

for (int i = 0, j = g; j < str.length(); i++, j++) {

if (g == 0) {

dp[i][j] = true;

} else if (g == 1) {

if (str.charAt(i) == str.charAt(j)) {

dp[i][j] = true;

} else {

dp[i][j] = false;

}

} else {

if (str.charAt(i) == str.charAt(j) && dp[i + 1][j - 1] == true) {

dp[i][j] = true;

} else {

dp[i][j] = false;

}

}

if (dp[i][j]) {

len = g + 1;

}

}

}

System.out.println(len);

}

}

Code : 42  
package Topic\_20\_DynamicProgramming;

import java.util.Arrays;

import java.util.Scanner;

public class ZP\_RussianDollEnvelopes {

public static class Envelope implements Comparable<Envelope> { //1

int w;

int h;

Envelope(int w, int h) {

this.w = w;

this.h = h;

}

public int compareTo(Envelope o) { //1

return this.w - o.w;

}

}

public static void main(String[] args) throws Exception {

Scanner scn = new Scanner(System.in);

int n = Integer.parseInt(scn.nextLine()); //2

Envelope[] envlps = new Envelope[n];

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

String line = scn.nextLine();

String[] parts = line.split(" ");

int w = Integer.parseInt(parts[0]);

int h = Integer.parseInt(parts[1]);

envlps[i] = new Envelope(w, h);

}

Arrays.sort(envlps); //4

int[] dp = new int[n]; //5

int omax = 0; //omax=overall max

for (int i = 0; i < dp.length; i++) {

int max = 0;

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

if (envlps[j].h < envlps[i].h && envlps[j].w < envlps[i].w) { //6

if (dp[j] > max) {

max = dp[j];

}

}

}

dp[i] = max + 1;

if (dp[i] > omax) { //7

omax = dp[i];

}

}

System.out.println(omax); //8

}

}

Code : 43  
package Topic\_20\_DynamicProgramming;

import java.util.Scanner;

public class ZQ\_CatalanNumber {

public static void main(String[] args) throws Exception {

Scanner scn = new Scanner(System.in);

int n = scn.nextInt();

long[] dp = new long[n + 1];

dp[0] = 1; //since 0th Catalan number is 1

dp[1] = 1; //since 1st Catalan number is also 1

for (int i = 2; i < dp.length; i++) {

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

dp[i] += dp[j] \* dp[i - 1 - j];

}

}

System.out.println(dp[n]); //nth Catalan number

}

}

Code : 44  
package Topic\_20\_DynamicProgramming;

import java.util.Scanner;

public class ZR\_NumberOfBsts {

public static void main(String[] args) throws Exception {

Scanner scn = new Scanner(System.in);

int n = scn.nextInt();

long[] dp = new long[n + 1];

dp[0] = 1; //since 0th Catalan number is 1

dp[1] = 1; //since 1st Catalan number is also 1

for (int i = 2; i < dp.length; i++) {

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

dp[i] += dp[j] \* dp[i - 1 - j];

}

}

System.out.println(dp[n]); //nth Catalan number

}

}

Code : 45  
package Topic\_20\_DynamicProgramming;

import java.util.Scanner;

public class Z\_PaintFence {

public static void main(String[] args) throws Exception {

Scanner scn = new Scanner(System.in);

int n = scn.nextInt();

int k = scn.nextInt();

long[] dp = new long[n + 1];

long same = k \* 1;

long diff = k \* (k - 1);

long total = same + diff;

for (int i = 3; i <= n; i++) {

same = diff \* 1;

diff = total \* (k - 1);

total = same + diff;

}

System.out.println(total);

}

}