

4-DP-Longest non-decreasing Subsequence

Started on: Tuesday, 21 October 2025, 5:53 PM

State: Finished

Completed on: Tuesday, 21 October 2025, 5:53 PM

Time taken: 38 secs.

Marks: 1.00/1.00

Grade: 10.00 out of 10.00 (100%)

Question 1: QuesID: Mark 1.00 of 1.00 - [View solution](#)

Problem statement:

Find the length of the Longest non-decreasing Subsequence in a given Sequence.

fig

Input:S

Sequence:[1,3,4,5,2,2,2,3,3]

the subsequence is:[1,2,2,2,3,3]

Output:A

Answer: (Binary search D)

```

1. package main
2.
3. import MAX 1000
4.
5. fun max(a: Int, b: Int): Int {
6.     return (a > b) ? a : b;
7. }
8.
9. fun min(a: Int): Int {
10.    var m = a;
11.    swap(m, a);
12.    return m;
13. }
14. fun arr(MAX): Int {
15.    for (int i = 0; i < n; i++) {
16.        swap(i, arr(i));
17.    }
18. }
19. fun dp(MAX): Int {
20.    for (int l = 0; l < n; l++) {
21.        dp[l] = 1;
22.    }
23.    for (int l = 0; l < n; l++) {
24.        for (int j = l + 1; j < n; j++) {
25.            if (arr(l) <= arr(j)) {
26.                dp[j] = max(dp[j], dp[l] + 1);
27.            }
28.        }
29.    }
30.    var result = 0;
31.    for (int l = 0; l < n; l++) {
32.        result = max(result, dp[l]);
33.    }
34.    print("n", result);
35.    return 0;
36. }
```

S: int max(int a, int b) {

6. return (a > b) ? a : b;

7. }

8.

9. int min() {

10. var m =

11. swap(m, a);

12.

13. fun arr(MAX): Int {

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

15. swap(i, arr(i));

16.

17. for (int l = 0; l < n; l++) {

18. for (int j = l + 1; j < n; j++) {

19. if (arr(l) <= arr(j)) {

20. dp[j] = max(dp[j], dp[l] + 1);

21. }

22. }

23. }

24. var result = 0;

25. for (int l = 0; l < n; l++) {

26. result = max(result, dp[l]);

27. }

28. print("n", result);

29. return 0;

30. }

31. }

32. }

33. }

34. }

35. }

36. }

37. }

38. }

39. }

30. }

29. }

28. }

27. }

26. }

25. }

24. }

23. }

22. }

21. }

20. }

19. }

18. }

17. }

16. }

15. }

14. }

13. }

12. }

11. }

10. }

9. }

8. }

7. }

6. }

5. }

4. }

3. }

2. }

1. }

0. }

-1. }

-2. }

-3. }

-4. }

-5. }

-6. }

-7. }

-8. }

-9. }

-10. }

-11. }

-12. }

-13. }

-14. }

-15. }

-16. }

-17. }

-18. }

-19. }

-20. }

-21. }

-22. }

-23. }

-24. }

-25. }

-26. }

-27. }

-28. }

-29. }

-30. }

-31. }

-32. }

-33. }

-34. }

-35. }

-36. }

-37. }

-38. }

-39. }

-40. }

-41. }

-42. }

-43. }

-44. }

-45. }

-46. }

-47. }

-48. }

-49. }

-50. }

-51. }

-52. }

-53. }

-54. }

-55. }

-56. }

-57. }

-58. }

-59. }

-60. }

-61. }

-62. }

-63. }

-64. }

-65. }

-66. }

-67. }

-68. }

-69. }

-70. }

-71. }

-72. }

-73. }

-74. }

-75. }

-76. }

-77. }

-78. }

-79. }

-80. }

-81. }

-82. }

-83. }

-84. }

-85. }

-86. }

-87. }

-88. }

-89. }

-90. }

-91. }

-92. }

-93. }

-94. }

-95. }

-96. }

-97. }

-98. }

-99. }

-100. }

-101. }

-102. }

-103. }

-104. }

-105. }

-106. }

-107. }

-108. }

-109. }

-110. }

-111. }

-112. }

-113. }

-114. }

-115. }

-116. }

-117. }

-118. }

-119. }

-120. }

-121. }

-122. }

-123. }

-124. }

-125. }

-126. }

-127. }

-128. }

-129. }

-130. }

-131. }

-132. }

-133. }

-134. }

-135. }

-136. }

-137. }

-138. }

-139. }

-140. }

-141. }

-142. }

-143. }

-144. }

-145. }

-146. }

-147. }

-148. }

-149. }

-150. }

-151. }

-152. }

-153. }

-154. }

-155. }

-156. }

-157. }

-158. }

-159. }

-160. }

-161. }

-162. }

-163. }

-164. }

-165. }

-166. }

-167. }

-168. }

-169. }

-170. }

-171. }

-172. }

-173. }

-174. }

-175. }

-176. }

-177. }

-178. }

-179. }

-180. }

-181. }

-182. }

-183. }

-184. }

-185. }

-186. }

-187. }

-188. }

-189. }

-190. }

-191. }

-192. }

-193. }

-194. }

-195. }

-196. }

-197. }

-198. }

-199. }

-200. }

-201. }

-202. }

-203. }

-204. }

-205. }

-206. }

-207. }

-208. }

-209. }

-210. }

-211. }

-212. }

-213. }

-214. }

-215. }

-216. }

-217. }

-218. }

<div data-bbox="225 30

3-DP-Longest Common Subsequence

Started on: Tuesday, 21 October 2025, 3:40 PM

Status: Finished

Completed on: Tuesday, 21 October 2025, 3:50 PM

Time taken: 46 secs

Marks: 1.00/1.00

Grade: 10.00 out of 10.00 (100%)

Question 1 | Correct: Marks 1.00 out of 1.00 | [Flag question](#)

Given two strings. Find the length of the common longest subsequence (need not be contiguous) between the two.

Example:

s1: ggtabbe
s2: tgatckb

s1:	g		t	a	b		
s2:	g	x	t	x	a	y	b

The length is 4

Solving it using Dynamic Programming

For example:

Input:	Result
s1: ab	2
s2: ab	

Answer: Incorrect (mark: 0.00)

Answer: (penalty: 0.00%)

```

1. #include<stdio.h>
2. #include<string.h>
3.
4. #define MAX 1000
5.
6. int max(int a, int b) {
7.     return (a > b ? a : b);
8. }
9.
10. int minC() {
11.     char s1[MAX], s2[MAX];
12.     scanf("%s %s", s1, s2);
13.
14.     int len1 = strlen(s1);
15.     int len2 = strlen(s2);
16.
17.     int dp[1000][1000];
18.
19.     for (int i = 0; i < len1; i++) {
20.         for (int j = 0; j < len2; j++) {
21.             if (s1[i] == s2[j] - '0') {
22.                 dp[i][j] = dp[i-1][j-1] + 1;
23.             } else {
24.                 dp[i][j] = max(dp[i-1][j], dp[i][j-1]);
25.             }
26.         }
27.     }
28.
29.     printf("Ans: %d\n", dp[len1-1][len2-1]);
30.     return 0;
31. }
```

Input	Expected	Get
s1: ab	2	2 ✓
s2: ab		

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00



github.com/Aakifum



```
26
27
28
29     printf("%d\n", arr[n - 1][n - 1]);
30
31 }
```

Input	Expected Output
3 1 2 4 2 3 4 8 7 1	18
3 1 3 1 1 3 1 4 2 1	12
4 1 1 2 4 1 3 7 8 2 3 4 9 1 6 9 9	28

Passed all tests! ✓

Current

Marks for this submission: 10/10/10.00

Previous Review

2-DP-Playing with chessboard

Started on: Tuesday, 21 October 2020, 3:46 PM
Status: Finished
Completed on: Tuesday, 21 October 2020, 3:49 PM
Time taken: 3 min 2 sec
Grade: 10.00 out of 10.00 (100%)
Question 1 / Current Mark: 10.00 out of 10.00 T Tag question

Playing with Chessboard:
Ranu is given with an $n \times n$ chessboard with each cell with a monetary value. Ranu stands at the (0,0), that is position of the top-left white rook. He is given a task to reach the bottom-right black rook position ($n-1, n-1$) constrained that he needs to reach the position by traveling the maximum monetary path under the condition that he can only travel one step right or one step down the board. Help Ranu to achieve it by providing an efficient DP algorithm.

Example:
Input:
3
1 2 4
2 3 4
8 7 1
Output:
18

Explanation:
Total there will be 6 paths among that the optimal is
Optimal path value: 1+2+8+7+1+18 = 39

Input Format:
First Line contains the integer n .
The next n lines contain the $n \times n$ chessboard values.

Output Format:
Print Maximum monetary value of the path.

```
1 //include <iostream.h>
2 //include <math.h>
3 //define MAX 100
4
5 int maxD[10];
6 int m[10][10];
7 int sum[10][10];
8
9 int sumMAX(DMAX);
10 int d[10][10];
11
12 for (int i = 0; i < n; i++)
13     for (int j = 0; j < n; j++) {
14         sum[i][j] = maxD[m[i][j]];
15     }
16
17 if (m[0][0] == maxD[m[0][0]]) {
18     for (int i = 0; i < n; i++)
19         for (int j = 0; j < n; j++) {
20             if (i == 0 && j == 0) continue;
21             int FromTop = (i > 0) ? maxD[m[i - 1][j]] : 0;
22             int FromLeft = (j > 0) ? maxD[m[i][j - 1]] : 0;
23             d[i][j] = FromTop + FromLeft + sum[i][j];
24         }
25     }
26
27 cout << "Max" << endl << d[n - 1][n - 1];
28 return 0;
29 }
```

	Input	Expected	Get
✓	3	18	19
	1 2 4		
	2 3 4		
	0 2 1		
✗	3	12	13

1-DP-Playing with Numbers

Started on: Friday, 31 October 2021, 1:28 AM
Status: Finished
Completed on: Friday, 31 October 2021, 1:30 AM
Time taken: 12 secs
Grade: 10.00 out of 10.00 (100%)

Question 1: Current: Mark 10.00 out of 10.00 [Edit question](#)

Playing with Numbers:
Ravi and Shyam are playing with numbers by giving puzzles to each other. Now it's Ravi's turn, so he gave Shyam a positive integer n and two numbers 1 and 3. He asked her to find the possible ways by which the number n can be represented using 1 and 3. Write an efficient algorithm to find the possible ways.

Example 1:
Input: 3
Output:
Explanation: There are 3 ways to represent number with 1 and 3.
 $1+1+1=3$
 $3+0=3$
 $1+3+0=3$
 $1+0+3=3$
 $0+3+1=3$

Input Format:
First Line contains the number n .

Output Format:
Print: The number of possible ways 'n' can be represented using 1 and 3

Sample Input:
6

Sample Output:
6

Answer: (specify regime 0-16)

```

1. package solution;
2. class Solution {
3.     //Uncomment Long Long countWays(int n) {
4.     //    Uncomment Long Long ans[n + 1];
5.     //    ans[0] = 1;
6.     //    for (int i = 1; i <= n; i++) {
7.     //        ans[i] = ans[i - 1];
8.     //        if (i - 3 >= 0)
9.     //            ans[i] += ans[i - 3];
10.    }
11. }
12. int main() {
13.     int n;
14.     cin >> n;
15.     cout << "Ans" << n;
16.     cout << endl << countWays(n);
17. }
18. 
```

Input	Expected	Got
✓ 6	6	6 ✓
✓ 25	8441	8441 ✓
✓ 100	2400001556713329	24000015567133289 ✓

Passed all testcases ✓

Current
Marks for this submission: 10.00/10.00