55.2

Muhammad Tanveer

Software Engineer

PDF generated at: 18 Jun 2024 09:16:26 UTC

View this report on HackerRank

Score

55.2% • 80 / 145

scored in CodePath TIP103: Unit 2 Assessment - Summer 2024 in 57 min 8 sec on 17 Jun 2024 20:03:04 PDT

Candidate Information

Email	tanveerm176@gmail.com
Test	CodePath TIP103: Unit 2 Assessment - Summer 2024
Candidate Packet	View
Taken on	17 Jun 2024 20:03:04 PDT
Time taken	57 min 8 sec/ 90 min
Invited by	CodePath

Skill Distribution

No.	Skill	Score
1	Problem Solving Basic	47%

Tags Distribution

Arrays	90%	Hard	0%

Data Structures	0%	Algorithms	50%
Problem Solving	50%	Easy	50%
Strings	0%	Interviewer Guidelines	0%

Questions

Status	No.	Question	Time Taken	Skill	Score
✓	1	True or False: Arrays Multiple Choice	29 sec	-	5/5
✗	2	Output of this snippet Multiple Choice	26 sec	-	0/5
✓	3	Why use an array? Multiple Choice	12 sec	-	5/5
✓	4	Time complexity of this snippet Multiple Choice	3 min 28 sec	-	5/5
✗	5	Time complexity of the binary search algorithm Multiple Choice	53 sec	-	0/5
✗	6	Sorted Array Multiple Choice	10 min 11 sec	Problem Solving (Basic)	0/5

✓	7	River Records Coding	21 min 26 sec	Problem Solving (Basic)	50/50
⊗	8	No Pairs Allowed Coding	10 min 49 sec	Problem Solving (Basic)	0/50
✓	9	Output of a List Multiple Choice	6 min 4 sec	-	5/5
✓	10	Find the middle element of the linked list Multiple Choice	1 min 30 sec	-	5/5
✓	11	True or False: Linked Lists Multiple Choice	1 min 33 sec	-	5/5

1. True or False: Arrays

✓ Correct

Multiple Choice

Question description

Which one of the following statements is not true about an array?

Candidate's Solution

Options: (Expected answer indicated with a tick)

● When an an array is full, it expands automatically ✓

○ Duplicate values are allowed in an array

○ Arrays understand the concept of ordered elements

○ A zero index is used to refer to the first element of an array

No comments.

2. Output of this snippet

⊗ Incorrect

Multiple Choice

Question description

What will be the output of the following code snippet?

```
def solve():
  a = [1, 2, 3, 4, 5]
  sum = 0
  for i in a:
    if i % 2 == 0:
      sum += a[i]
  print(sum)
```

Candidate's Solution

Options: (Expected answer indicated with a tick)

○ 5

○ 15

● 6

○ 8 ✓

ⓘ No comments.

3. Why use an array?

✓ Correct

Multiple Choice

Question description

Which of the following is the advantage of the array data structure?

Candidate's Solution

Options: (Expected answer indicated with a tick)

● Easier to access elements ✓

○ The index of the first element starts from 1

○ Elements of the array can’t be stored

○ Elements of mixed data type can be stored

ⓘ No comments.

4. Time complexity of this snippet

✓ Correct

Multiple Choice

Question description

What is the time complexity of the following code snippet?

```
def solve():
    s = "scaler"
    n = len(s)
    for i in n:
       s = s + s[i]
    print(s)
```

Candidate's Solution

Options: (Expected answer indicated with a tick)

○ <p>O(n)</p>

● <p>O(n^2)</p> ✓

○ <p>O(n log n)</p>

○ <p>O(1)</p>

ⓘ No comments.

5. Time complexity of the binary search algorithm

⊗ Incorrect

Multiple Choice

Question description

What is the time complexity of the binary search algorithm?

Candidate's Solution

Options: (Expected answer indicated with a tick)

○ O(n)

○ O(1)

○ O(log2n) ✓

○ O(n^2)

○ Runtime error

● None of the above

ⓘ No comments.

6. Sorted Array

ⓧ Incorrect

Multiple Choice Arrays Hard Data Structures

Question description

Given an array, arr[0, 2, 3, 5, 4], and an integer x = 1, sort the array using the method below.

Each operation is: Choose a number i such that arr[i] > x. Swap the values of a[i] and x.

What is the minimum number of operations required to sort the array in ascending order?

Interviewer guidelines

array a[5] = {0,2,3,5,4} and X = 1 (initial values)

- Choose i = 2, as a_i > X , swap a_i and X updated array a[5] = {0,1,3,5,4} X = 2
- Choose i = 3, as a_i > X , swap a_i and X updated array a[5] = {0,1,2,5,4} X = 3
- Choose i = 4, as a_i > X , swap a_i and X updated array a[5] = {0,1,2,3,4} X = 5

After 3 steps the initial array is sorted.

Candidate's Solution

Options: (Expected answer indicated with a tick)

○ 5

○ 3 ✓

● 4

○ 6

ⓘ **No comments.**

7. River Records

✓ Correct

Coding Algorithms Arrays Problem Solving Easy

Question description

Given an array of integers, without reordering, determine the maximum difference between any element and any prior smaller element. If there is never a lower prior element, return -1.

Example
arr = [5, 3, 6, 7, 4]

There are no earlier elements than *arr[0]*.
There is no earlier reading with a value lower than *arr[1]*.
There are two lower earlier readings with a value lower than *arr[2] = 6:*

- *arr[2] - arr[1] = 6 - 3 = 3*
- *arr[2] - arr[0] = 6 - 5 = 1*

There are three lower earlier readings with a lower value than $arr[3] = 7$:

- $arr[3] - arr[2] = 7 - 6 = 1$
- $arr[3] - arr[1] = 7 - 3 = 4$
- $arr[3] - arr[0] = 7 - 5 = 2$

There is one lower earlier reading with a lower value than $arr[4] = 4$:

- $arr[4] - arr[1] = 4 - 3 = 1$

The maximum trailing record is $arr[3] - arr[1] = 4$.

Example

$arr = [4, 3, 2, 1]$

No item in *arr* has a lower earlier reading, therefore return *-1*

Function Description

Complete the function *maximumTrailing* in the editor below.

maximumTrailing has the following parameter(s):

int arr[n]: an array of integers

Returns:

int: the maximum trailing difference, or *-1* if no element in *arr* has a lower earlier value

Constraints

- $1 \le n \le 2 \times 10^5$
- $-10^6 \le arr[i] \le 10^6$ and $0 \le i < n$

▼ INPUT FORMAT FOR CUSTOM TESTING

Input from stdin will be processed as follows and passed to the function:

The first line contains a single integer, *n*, the number of elements in the array *arr*.
Each of the *n* subsequent lines contains a single integer, each an element *arr[i]* where $0 \le i < n$.

▼ SAMPLE CASE 0

Sample Input 0

```
STDIN       Function
-----       --------
7     →    arr[] size n = 7
2     →    arr = [2, 3, 10, 2, 4, 8, 1]
3
10
2
4
8
1
```

Sample Output

```
8
```

Explanation

Differences are calculated as:

- *3 - [2] = [1]*
- *10 - [3, 2] = [7, 8]*
- *4 - [2, 3, 2] = [2, 1, 2]*
- *8 - [4, 2, 3, 2] = [4, 6, 5, 6]*

The maximum trailing difference is *10 - 2 = 8.*

▼ SAMPLE CASE 1

Sample Input 1

```
STDIN       Function
-----       --------
6     →    arr[] size n = 6
7     →    arr = [7, 9, 5, 6, 3, 2]
9
5
6
3
2
```

Sample Output

```
2
```

Explanation

Differences are calculated as:

- *9 - [7] = 2*
- *6 - [5] = 1*

The maximum trailing difference is *2.*

Candidate's Solution

Language used: **Python 3**

```python
#!/bin/python3

import math
import os
import random
import re
import sys


#
# Complete the 'maxTrailing' function below.
#
# The function is expected to return an INTEGER.
# The function accepts INTEGER_ARRAY arr as parameter.
#

def maxTrailing(arr):
    # Write your code here

    # if array has only one element return -1
    if len(arr) < 2:
        return -1

    # init min_val at first element
    min_val = arr[0]
    # set max_diff to base case
    max_diff = -1

    # iterate over the array starting from second element
    for i in range(1, len(arr)):
        # if element is less than min val then set min val to the element
        if arr[i] < min_val:
            min_val = arr[i]
        # otherwise find the difference between the current element and the min val
        # then find the max between that and the maximum difference and set that to the max difference
        elif arr[i]>min_val:
            max_diff = max(max_diff, arr[i]-min_val)

        # if the current element and the first element are equal continue to next element
        else:
            continue

    return max_diff
if __name__ == '__main__':
    fptr = open(os.environ['OUTPUT_PATH'], 'w')

    arr_count = int(input().strip())

    arr = []

    for _ in range(arr_count):
        arr_item = int(input().strip())
        arr.append(arr_item)

    result = maxTrailing(arr)

    fptr.write(str(result) + '\n')

    fptr.close()

```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
TestCase 0	Easy	Sample	Success	1	0.0399 sec	10.2 KB
TestCase 1	Easy	Sample	Success	1	0.0303 sec	10.3 KB
TestCase 2	Easy	Sample	Success	1	0.0359 sec	10.4 KB
TestCase 3	Easy	Sample	Success	1	0.0315 sec	10.2 KB

TestCase 4	Easy	Hidden	Success	10	0.0308 sec	10.3 KB
TestCase 5	Easy	Sample	Success	1	0.0315 sec	10.3 KB
TestCase 6	Easy	Hidden	Success	11	0.0311 sec	10.3 KB
TestCase 7	Easy	Hidden	Success	5	0.1383 sec	14.1 KB
TestCase 8	Easy	Hidden	Success	5	0.1142 sec	13.5 KB
TestCase 9	Easy	Hidden	Success	6	0.1131 sec	13.6 KB
TestCase 10	Easy	Hidden	Success	2	0.2326 sec	18 KB
TestCase 11	Easy	Hidden	Success	2	0.2877 sec	17.8 KB
TestCase 12	Easy	Hidden	Success	2	0.2612 sec	18 KB
TestCase 13	Easy	Hidden	Success	2	0.2134 sec	18 KB

ⓘ No comments.

8. No Pairs Allowed

⊗ Incorrect

Coding · Strings · Easy · Algorithms · Problem Solving · Interviewer Guidelines

Question description

For each word in a list of words, if any two adjacent characters are equal, change one of them. Determine the minimum number of substitutions so the final string contains no adjacent equal characters.

Example

words = ['add', 'boook', 'break']

1. *'add':* change one *d* (1 change)
2. *'boook':* change the middle *o* (1 change)
3. *'break':* no changes are necessary (0 changes)

The return array is *[1,1,0]*.

Function Description

Complete the function *minimalOperations* in the editor below.

minimalOperations has the following parameter(s):
string words[n]: an array of strings

Returns:
int[n]: each element *i* is the minimum substitutions for *words[i]*

Constraints

- $1 \le n \le 100$
- $2 \le$ *length of words[i]* $\le 10^5$
- Each character of *words[i]* is in the range *ascii[a-z]*.

▼ INPUT FORMAT FOR CUSTOM TESTING

Input from stdin will be processed as follows and passed to the function.

The first line contains an integer *n*, the size of the array *words*.
Each of the next *n* lines contains a string *words[i]*.

▼ SAMPLE CASE 0

Sample Input 0

```
STDIN        Function Parameters
-----        -------------------
5        →  words[] Size = 5
ab        →  words[] = [ 'ab','aab','abb', 'abab','abaaaba' ]
aab
abb
abab
abaaaba
```

Sample Output 0

```
0
1
1
0
1
```

Explanation 0

- *words = 'ab'* is already acceptable, so *0* replacements are needed.
- *words = 'aab'* Replace an *'a'* with an appropriate character so *1* replacement.
- *words = 'abb'* is not acceptable. Replace a *'b'* with an appropriate character, again *1* replacement.
- *words = 'abab'* is already acceptable so *0* replacements are needed.
- *words = 'abaaaba'* is not acceptable. Replace the middle *'a'* in *'aaa'*, *1* replacement.

The return array is *[0, 1, 1, 0, 1]*.

Interviewer guidelines

▼ HINT 1

As you iterate through the string, which character(s) need to be tested for equivalence? For each character check only characters adjacent to it on the left.

▼ HINT 2

If you replace a character, can you always assume the replacement differs from the character to its right as well?
Why, and how can you use this fact?

The characters left and right can either be the same or different. There are 25 or 24 letters available in all cases.
This allows you to skip over the next character after a replacement.

▼ SOLUTION

Concepts covered: This problem covers the concepts of strings and arrays.

Optimal Solution:
For each string, start with the character at index 1. Compare each character to the one to its left, with one exception. If the two letters are equal, assume the character to its left remains the same and the current character is replaced. It can always be replaced with a character different from both adjacent characters, left and right. The next character after a replacement can be skipped.

```
def minimalOperations(words):
    ans = []
    for w in words:
        count = 0
        i = 1
        while i < len(w):
            # test for match
            if w[i] == w[i-1]:
                # yes: increment counter and skip the next character
                count += 1
                i += 2
            else:
                # no: move to the next character
                i += 1
        ans.append(count)
    return ans
```

Sub-optimal approach: For each string, iterate its characters, checking if they are equal to the one to their left. If the characters match, replace the current character with '#'. For example: string "abbca". We check pairs one by one, 'ab', 'bb', here characters are the same, so we replace the second character with '#'. Continue checking symbols one by one, '#c', 'ca'. This finishes the process.

```
def minimalOperations(words):

    ans = []

    for i in range(len(words)):
        # replacement counter
        cur_ans = 0
        # convert the string to a list so it is mutable
        cur_word = list(words[i])

        for j in range(1, len(words[i])):
            # if characters match, replace the current character
            if cur_word[j-1] == cur_word[j]:
                # replace with a character guaranteed to be different from the next character
                cur_word[j] = "#"
                cur_ans += 1

        ans.append(cur_ans)

    return ans
```

Error Handling:

1. The case of a zero length string must be handled separately.

▼ COMPLEXITY ANALYSIS

Time Complexity - O(N) where N is the total number of characters in all words.

Accessing all characters in all words requires O(N) time

Space Complexity - O(1) - For the optimal solution only two integer variables are required.

Candidate's Solution

Language used: **Python 3**

```
#!/bin/python3

import math
import os
import random
import re
import sys


#
```

```
# Complete the 'minimalOperations' function below.
#
# The function is expected to return an INTEGER_ARRAY.
# The function accepts STRING_ARRAY words as parameter.
#

def minimalOperations(words):
    pass
    # Write your code here
if __name__ == '__main__':
    fptr = open(os.environ['OUTPUT_PATH'], 'w')

    words_count = int(input().strip())

    words = []

    for _ in range(words_count):
        words_item = input()
        words.append(words_item)

    result = minimalOperations(words)

    fptr.write('\n'.join(map(str, result)))
    fptr.write('\n')

    fptr.close()

```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
TestCase 0	Easy	Sample	Runtime Error	0	0.0324 sec	10.2 KB
TestCase 1	Easy	Sample	Runtime Error	0	0.0384 sec	10.1 KB
TestCase 2	Easy	Sample	Runtime Error	0	0.0348 sec	10.3 KB

TestCase 3	Easy	Sample	Runtime Error	0	0.0316 sec	10.3 KB
TestCase 4	Easy	Hidden	Runtime Error	0	0.0445 sec	10.4 KB
TestCase 5	Easy	Sample	Runtime Error	0	0.0287 sec	10.3 KB
TestCase 6	Easy	Hidden	Runtime Error	0	0.0374 sec	10.4 KB
TestCase 7	Easy	Hidden	Runtime Error	0	0.0533 sec	14.5 KB
TestCase 8	Easy	Hidden	Runtime Error	0	0.0472 sec	15.2 KB
TestCase 9	Easy	Hidden	Runtime Error	0	0.0499 sec	14.5 KB
TestCase 10	Easy	Hidden	Runtime Error	0	0.0399 sec	20.1 KB
TestCase 11	Easy	Hidden	Runtime Error	0	0.0379 sec	20 KB
TestCase 12	Easy	Hidden	Runtime Error	0	0.0486 sec	20.1 KB
TestCase 13	Easy	Hidden	Runtime Error	0	0.0662 sec	20 KB

ⓘ No comments.

9. Output of a List

✓ Correct

Multiple Choice

Question description

What will be the output of the following code snippet for the list *1->2->3->4->5->6*?

```
def solve(start.data):
      if start == NONE return

      print(start.data)

      if start.next != NONE:
        solve(start.next.next)

      print(start.data)
```

Candidate's Solution

Options: (Expected answer indicated with a tick)

○ 1 2 3 4 5 6

● **1 3 5 5 3 1** ✓

○ 1 3 5 1 3 5

○ 2 4 6 1 3 5

ⓘ No comments.

10. Find the middle element of the linked list

✓ Correct

Multiple Choice

Question description

Which of the following algorithm is the optimal way to find the middle element of the linked list?

Candidate's Solution

Options: (Expected answer indicated with a tick)

○ Find the length, then traverse to length/2th node

● **Fast and slow pointer method** ✓

○ Find distance of all nodes, and print the middle one

○ None of the above

ⓘ No comments.

11. True or False: Linked Lists

✓ Correct

Multiple Choice

Question description

Which of the following statements is/are true?

Candidate's Solution

Options: (Expected answer indicated with a tick)

○ Random access of elements is not possible

○ Arrays have better cache locality than linked lists

○ The size of linked list is dynamic and can be changed as needed

● All of the above ✓

ⓘ No comments.