100

Rajasekhar Reddy Kolagotla

Other

PDF generated at: 23 Jun 2025 08:00:33 UTC

View this report on HackerRank

Score

100% • 80 / 80

scored in TIP102: Unit 3 Version A (Standard) - Summer 2025 in 29 min 33 sec on 23 Jun 2025 00:28:47 PDT

Candidate Information

Email	rajasekhar1131997@gmail.com
Test	TIP102: Unit 3 Version A (Standard) - Summer 2025
Candidate Packet	View
Taken on	23 Jun 2025 00:28:47 PDT
Time taken	29 min 33 sec/ 90 min
Personal Member ID	126663
Email Address with CodePath	rajasekhar1131997@gmail.com
Github username with CodePath	Rajasekhar1131997
Invited by	CodePath

Suspicious Activity detected

Code similarity

Code similarity • 1 question

Skill Distribution

There is no associated skills data that can be shown for this assessment

Tags Distribution

There is no associated tags data that can be shown for this assessment

Questions

Coding Questions • 60 / 60

Status	No.	Question	Time Taken	Skill	Score	Code Quality
✓	1	Valid Parenthesis (Stack question) Coding	5 min 36 sec	-	20/20	-

✓	2	First Non-Repeating Character Coding	9 min 40 sec	-	20/20	-
✓	3	Two Sum Sorted Coding	3 min 5 sec	-	20/20	-

Multiple Choice + Debugging • 20 / 20

Status	No.	Question	Time Taken	Skill	Score	Code Quality
✓	4	What will be the output of the following code snippet? Multiple Choice	2 min 31 sec	-	5/5	-
✓	5	What will be the output of the following code snippet? Multiple Choice	2 min 25 sec	-	5/5	-
✓	6	What will be the output of the following code snippet? Multiple Choice	2 min 33 sec	-	5/5	-
✓	7	Debug this code! Coding	3 min 19 sec	-	5/5	-

1. Valid Parenthesis (Stack question)

Correct

Coding

Question description

Given a string `s` containing just the characters `'('`, `')'`, `'{'`, `'}'`, `'['` and `']'`, determine if the input string is valid.
An input string is valid if:

1. Open brackets must be closed by the same type of brackets.
2. Open brackets must be closed in the correct order.
3. Every close bracket has a corresponding open bracket of the same type.

Example 1:
Input: s = "()"
Output: true

Example 2:
Input: s = "()[]{}"
Output: true

Example 3:
Input: s = "(]"
Output: false

Candidate's Solution

Language used: Python 3

```python
#!/bin/python3

import math
import os
import random
import re
import sys
import ast


#
# Complete the 'is_valid' function below.
#
# The function is expected to return a BOOLEAN.
# The function accepts STRING s as parameter.
#
```

```
18
19  def is_valid(s):
20      dictionary = {
21          ')': '(',
22          '}' : '{',
23          ']' : '['
24      }
25      stack = []
26      for ch in s:
27          if ch in dictionary.values():
28              stack.append(ch)
29          elif ch in dictionary.keys():
30              if not stack or stack[-1] != dictionary[ch]:
31                  return False
32              stack.pop()
33      if len(stack) == 0:
34          return True
35      else:
36          return False
37
38  if __name__ == '__main__':
39      outfile = open(os.environ['OUTPUT_PATH'], 'w')
40      input_lines = sys.stdin.read().strip().splitlines()
41
42      for input_str in input_lines:
43          if input_str.strip() == "":
44              continue
45
46          try:
47              s = ast.literal_eval(input_str)
48
49              result = is_valid(s)
50
51              outfile.write(str(result) + '\n')
52          except (ValueError, SyntaxError):
53              print("Error: Invalid input")
54      outfile.close()
```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Testcase 0	Easy	Hidden	Success	0	0.0298 sec	10.9 KB

Testcase 1	Easy	Hidden	Success	0	0.0298 sec	10.9 KB
Testcase 2	Easy	Hidden	Success	0	0.0271 sec	10.9 KB
Empty String	Easy	Hidden	Success	0	0.0277 sec	11 KB
Single Bracket (Unmatched)	Easy	Hidden	Success	0	0.0307 sec	10.9 KB
Single Bracket (Unmatched)	Easy	Hidden	Success	0	0.0312 sec	11 KB
Nested Brackets	Easy	Hidden	Success	0	0.0293 sec	10.9 KB
Multiple Nested Brackets (True)	Easy	Hidden	Success	0	0.0336 sec	11 KB
Incorrectly Nested Brackets (False)	Easy	Hidden	Success	0	0.0335 sec	10.9 KB
Multiple Nested Brackets (False)	Easy	Hidden	Success	0	0.0291 sec	10.9 KB
Incorrectly Nested Brackets (True)	Easy	Hidden	Success	0	0.0288 sec	10.8 KB

Mismatched Pairs (False)	Easy	Hidden	Success	0	0.0272 sec	10.9 KB
Mismatched Pairs (True)	Easy	Hidden	Success	0	0.0284 sec	11 KB
Pass/Fail Case	Easy	Hidden	Success	20	0.0283 sec	10.9 KB

No comments.

2. First Non-Repeating Character

Correct

Coding

Question description

Given a string `s`, find the first non-repeating character in it and return its index. If it does not exist, return -1.

Example 1:
Input: s = "leetcode"
Output: 0

Example 2:
Input: s = "loveleetcode"
Output: 2

Example 3:
Input: s = "aabb"
Output: -1

Candidate's Solution

Language used: Python 3

```python
#!/bin/python

import math
import os
import random
import re
import sys
import ast



#
# Complete the 'first_non_repeating_character' function below.
#
# The function is expected to return an INTEGER.
# The function accepts STRING s as parameter.
#

from collections import deque

def first_non_repeating_character(s):
    frequency = {}
    q = deque()
    for i, ch in enumerate(s):
        frequency[ch] = frequency.get(ch,0)+1
        q.append(i)
        while q and frequency[s[q[0]]] > 1:
            q.popleft()
    return q[0] if q else -1


if __name__ == '__main__':
    outfile = open(os.environ['OUTPUT_PATH'], 'w')
    input_lines = sys.stdin.read().strip().splitlines()

    for input_str in input_lines:
        if input_str.strip() == "":
            continue

        try:
            s = ast.literal_eval(input_str)

            result = first_non_repeating_character(s)

```

```
45              outfile.write(str(result) + '\n')
46          except (ValueError, SyntaxError):
47              print("Error: Invalid input")
48      outfile.close()
```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Testcase 0	Easy	Hidden	Success	0	0.0432 sec	11 KB
Testcase 1	Easy	Hidden	Success	0	0.0292 sec	11 KB
All Characters Repeating	Easy	Hidden	Success	0	0.0278 sec	11 KB
Empty String	Easy	Hidden	Success	0	0.0406 sec	10.9 KB
Single Character	Easy	Hidden	Success	0	0.0282 sec	11 KB
Non-Repeating Character at the End	Easy	Hidden	Success	0	0.0296 sec	11 KB
Non-Repeating Character in the Middle	Easy	Hidden	Success	0	0.0254 sec	10.9 KB
Case Sensitivity	Easy	Hidden	Success	0	0.0285 sec	11 KB

Pass/Fail Case	Easy	Hidden	Success	20	0.0287 sec	11 KB

No comments.

3. Two Sum Sorted

Correct

Coding

Question description

You are given a 1-indexed array of integers `numbers`, sorted in non-decreasing order, and an integer `target`.

Your task is to find **two distinct elements** in the array such that they **add up to the target**. Return the **1-based indices** of the two numbers in the form `[index1, index2]`, where `index1 < index2`.

Requirements

- You **may not use the same element twice**.
- Your solution must use **constant extra space**.

Example:
Input: numbers = [2, 7, 11, 15], target = 9
Output: [1, 2]

Explanation: numbers[1] + numbers[2] = 2 + 7 = 9

Candidate's Solution

Language used: **Python 3**

```
#!/bin/python3

import math
import os
import random
import re
import sys
import ast
```

```
 9
10
11
12  #
13  # Complete the 'two_sum' function below.
14  #
15  # The function is expected to return an INTEGER_ARRAY.
16  # The function accepts following parameters:
17  #  1. INTEGER_ARRAY numbers
18  #  2. INTEGER target
19  #
20
21  def two_sum(numbers, target):
22      left = 0
23      right = len(numbers) - 1
24      while left < right:
25          s = numbers[left] + numbers[right]
26
27          if s == target:
28              return [left+1, right+1]
29          elif s<target:
30              left += 1
31          else:
32              right -= 1
33
34      return []
35
36  if __name__ == '__main__':
37      outfile = open(os.environ['OUTPUT_PATH'], 'w')
38      input_data = sys.stdin.read().strip().splitlines()
39
40      for line in input_data:
41          input_list = ast.literal_eval(line)
42          nums = input_list[0]
43          target = input_list[1]
44
45          result = two_sum(nums, target)
46          outfile.write(str(result) + '\n')
47      outfile.close()
```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED

Testcase 0	Easy	Hidden	Success	0	0.0273 sec	10.8 KB
Testcase 1	Easy	Hidden	Success	0	0.0279 sec	11 KB
Testcase 2	Easy	Hidden	Success	0	0.0318 sec	10.9 KB
Target is the Sum of First and Last Elements	Easy	Hidden	Success	0	0.0254 sec	10.9 KB
Negative Numbers in the List	Easy	Hidden	Success	0	0.0272 sec	10.9 KB
Zeros in the List	Easy	Hidden	Success	0	0.0293 sec	10.9 KB
Target is Zero and Elements Include Negative and Positive Numbers	Easy	Hidden	Success	0	0.0297 sec	10.9 KB
Multiple Pairs Sum to Target, Return the First Pair	Easy	Hidden	Success	0	0.0294 sec	11 KB
Duplicates in the List with Valid Pair	Easy	Hidden	Success	0	0.0295 sec	10.9 KB
Pass/Fail Case	Easy	Hidden	Success	20	0.033 sec	10.9 KB

No comments.

4. What will be the output of the following code snippet?

Correct

Multiple Choice

Question description

```
def mystery_function(nums):
    left = len(nums) - 1
    right = len(nums) - 1

    while right >= 0:
        if nums[right] != 0:
            temp = nums[right]
            nums[right] = nums[left]
            nums[left] = temp
            left -= 1
        right -= 1
    return nums

nums = [0, 0, 1, 2, 0, 3]
print(mystery_function(nums))
```

Candidate's Solution

Options: (Expected answer indicated with a tick)

- [1, 2, 0, 0, 0, 3]
- [0, 1, 0, 2, 0, 3]

[1, 2, 3, 0, 0 ,0]

[0, 0, 0, 1, 2, 3]

No comments.

5. What will be the output of the following code snippet?

Correct

Multiple Choice

Question description

```
def process_strings(chars):
    stack = ["start"]
    for char in chars:
        if char.isupper():
            stack.append(char)
        elif stack and char.islower():
            stack.pop()
    return stack

chars = ['A', 'b', 'c', 'D', 'E', 'f']
print(process_strings(chars))
```

Candidate's Solution

Options: (Expected answer indicated with a tick)

['start', 'D', 'E']

['start', 'A', 'D', 'E']

['start', 'A', 'D']

['D']

No comments.

6. What will be the output of the following code snippet?

Correct

Multiple Choice

Question description

```
from collections import deque

def process_numbers(nums):
    queue = deque([10, 20, 30])
    for num in nums:
        if num % 2 == 0:
            queue.append(num)
        elif queue and num % 2 != 0:
            queue.popleft()
    return list(queue)

nums = [2, 3, 4, 5, 6, 7]
print(process_numbers(nums))
```

Candidate's Solution

Options: (Expected answer indicated with a tick)

- [x] [2, 4, 6]
- [ ] [20, 30, 2, 4, 6]
- [ ] [30, 2, 4, 6]
- [ ] [10, 2, 4, 6]

No comments.

7. Debug this code!

Correct

Coding

Question description

The code provided below incorrectly implements `is_palindrome()`. Correctly implemented, `is_palindrome()` accepts a string `s` and returns `True` if, after converting all uppercase letters into lowercase letters, `s` reads the same forward and backward. Otherwise, the function returns `False`. `s` is guaranteed to contain only alphanumeric characters. Alphanumeric characters include letters and numbers.

Identify any bug(s) within the given implementation and correct the code so that it successfully passes the provided test cases.

```
def is_palindrome(s):
    left, right = 0, len(s) - 1
```

```
    while left < right:

        left += 1
        right -= 1

        if s[left].lower() != s[right].lower():
            return False

    return True

# Test Cases
s1 = "amanaplanacanalPanama"
s2 = "abbd"
print(is_palindrome(s1))
print(is_palindrome(s2))
```

Candidate's Solution

Language used: **Python 3**

```
#!/bin/python3

import math
import os
import random
import re
import sys
import ast


#
# Complete the 'is_palindrome' function below.
#
# The function is expected to return a BOOLEAN.
# The function accepts STRING s as parameter.
#

def is_palindrome(s):
    left, right = 0, len(s) - 1

    while left < right:

        if s[left].lower() != s[right].lower():
            return False

```

```
26          left += 1
27          right -= 1
28      return True
29
30  if __name__ == '__main__':
31      input_data = sys.stdin.read().strip()
32      arr = ast.literal_eval(input_data)
33
34      result = is_palindrome(arr)
35      print(result)
```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Pass/Fail Case	Easy	Hidden	Success	5	0.0317 sec	10.8 KB

No comments.