



Venkata Naga Sri Sai Pranavi Kolipaka
Other

PDF generated at: 20 Jun 2025 18:47:39 UTC
[View this report on HackerRank](#)

Score

100% • 80 / 80
scored in TIP102: Unit 3 Version A (Standard) - Summer 2025 in 73 min 52 sec on 20 Jun 2025 10:31:50 PDT

Candidate Information

Email	kolipakavnssaipranavi@gmail.com
Test	TIP102: Unit 3 Version A (Standard) - Summer 2025
Candidate Packet	View
Taken on	20 Jun 2025 10:31:50 PDT
Time taken	73 min 52 sec/ 90 min
Personal Member ID	129054
Email Address with CodePath	kolipakavnssaipranavi@gmail.com
Github username with CodePath	Pranavi2002
Invited by	CodePath

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	9 min 19 sec	-	20/20	-
	2	First Non-Repeating Character Coding	22 min 21 sec	-	20/20	-

	3	Two Sum Sorted Coding	22 min 41 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	7 min 5 sec	-	5/5	-
	5	What will be the output of the following code snippet? Multiple Choice	3 min 42 sec	-	5/5	-
	6	What will be the output of the following code snippet? Multiple Choice	3 min 8 sec	-	5/5	-
	7	Debug this code! Coding	4 min 54 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

```
1  #!/bin/python3
2
3  import math
4  import os
5  import random
6  import re
7  import sys
8  import ast
9
10
11 #
12 # Complete the 'is_valid' function below.
13 #
14 # The function is expected to return a BOOLEAN.
15 # The function accepts STRING s as parameter.
16 #
17
18 def is_valid(s):
19     # Write your code here
```

```

20     d = {'(': ')': '(',
21           ')': '}',
22           '[': ']',
23           ']' : '['}
24     stack = []
25     for c in s:
26         if c in '({[':
27             stack.append(c)
28         else:
29             if not stack or stack[-1] != d.get(c):
30                 return False
31             stack.pop()
32 if __name__ == '__main__':
33     outfile = open(os.environ['OUTPUT_PATH'], 'w')
34     input_lines = sys.stdin.read().strip().splitlines()
35
36     for input_str in input_lines:
37         if input_str.strip() == "":
38             continue
39
40         try:
41             s = ast.literal_eval(input_str)
42
43             result = is_valid(s)
44
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	0.0343 sec	11 KB
Testcase 1	Easy	Hidden	Success	0	0.0299 sec	10.9 KB
Testcase 2	Easy	Hidden	Success	0	0.0278 sec	10.8 KB

Empty String	Easy	Hidden	Success	0	0.0287 sec	10.9 KB
Single Bracket (Unmatched)	Easy	Hidden	Success	0	0.0314 sec	10.8 KB
Single Bracket (Unmatched)	Easy	Hidden	Success	0	0.0283 sec	11 KB
Nested Brackets	Easy	Hidden	Success	0	0.0272 sec	11 KB
Multiple Nested Brackets (True)	Easy	Hidden	Success	0	0.0308 sec	10.9 KB
Incorrectly Nested Brackets (False)	Easy	Hidden	Success	0	0.0286 sec	11 KB
Multiple Nested Brackets (False)	Easy	Hidden	Success	0	0.0347 sec	10.9 KB
Incorrectly Nested Brackets (True)	Easy	Hidden	Success	0	0.0318 sec	10.9 KB
Mismatched Pairs (False)	Easy	Hidden	Success	0	0.0312 sec	10.8 KB
Mismatched Pairs (True)	Easy	Hidden	Success	0	0.0317 sec	10.9 KB

Pass/Fail Case

Easy

Hidden

Success

20

0.0311
sec

10.8 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: $s = \text{"leetcode"}$
Output: 0

Example 2:
Input: $s = \text{"loveleetcode"}$
Output: 2

Example 3:
Input: $s = \text{"aabb"}$
Output: -1

Candidate's Solution

Language used: Python 3

```
1 #!/bin/python
2
3 import math
4 import os
5 import random
6 import re
```

```
7 import sys
8 import ast
9
10
11 #
12 # Complete the 'first_non_repeating_character' function below.
13 #
14 # The function is expected to return an INTEGER.
15 # The function accepts STRING s as parameter.
16 #
17
18 from collections import deque, Counter
19
20 def first_non_repeating_character(s):
21     # Write your code here
22     frequency = Counter()
23     queue = deque()
24
25     for ch in s:
26         frequency[ch] += 1
27         queue.append(ch)
28
29         while queue and frequency[queue[0]] > 1:
30             queue.popleft()
31
32     if not queue:
33         return -1
34     else:
35         return s.index(queue[0])
36
37 if __name__ == '__main__':
38     outfile = open(os.environ['OUTPUT_PATH'], 'w')
39     input_lines = sys.stdin.read().strip().splitlines()
40
41     for input_str in input_lines:
42         if input_str.strip() == "":
43             continue
44
45         try:
46             s = ast.literal_eval(input_str)
47
48             result = first_non_repeating_character(s)
49
50             outfile.write(str(result) + '\n')
51         except (ValueError, SyntaxError):
52             print("Error: Invalid input")
53     outfile.close()
```


TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Testcase 0	Easy	Hidden	Success	0	0.0412 sec	10.9 KB
Testcase 1	Easy	Hidden	Success	0	0.0275 sec	10.8 KB
All Characters Repeating	Easy	Hidden	Success	0	0.0294 sec	10.9 KB
Empty String	Easy	Hidden	Success	0	0.0282 sec	10.8 KB
Single Character	Easy	Hidden	Success	0	0.0322 sec	10.9 KB
Non-Repeating Character at the End	Easy	Hidden	Success	0	0.0288 sec	10.8 KB
Non-Repeating Character in the Middle	Easy	Hidden	Success	0	0.0439 sec	10.8 KB
Case Sensitivity	Easy	Hidden	Success	0	0.0387 sec	10.5 KB
Pass/Fail Case	Easy	Hidden	Success	20	0.0341 sec	10.8 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

```
1 #!/bin/python3
2
3 import math
4 import os
5 import random
6 import re
7 import sys
8 import ast
9
10
11 #
12 # Complete the 'two_sum' function below.
13 #
```

```

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

```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Testcase 0	Easy	Hidden	Success	0	0.0284 sec	10.9 KB
Testcase 1	Easy	Hidden	Success	0	0.0288 sec	11 KB

Testcase 2	Easy	Hidden	Success	0	0.0316 sec	10.6 KB
Target is the Sum of First and Last Elements	Easy	Hidden	Success	0	0.0337 sec	11 KB
Negative Numbers in the List	Easy	Hidden	Success	0	0.0429 sec	10.9 KB
Zeros in the List	Easy	Hidden	Success	0	0.0292 sec	10.9 KB
Target is Zero and Elements Include Negative and Positive Numbers	Easy	Hidden	Success	0	0.0297 sec	10.6 KB
Multiple Pairs Sum to Target, Return the First Pair	Easy	Hidden	Success	0	0.0309 sec	11 KB
Duplicates in the List with Valid Pair	Easy	Hidden	Success	0	0.0298 sec	10.9 KB
Pass/Fail Case	Easy	Hidden	Success	20	0.0299 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']



[&#39;D&#39;]



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)



[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

```
1  #!/bin/python3
2
3  import math
4  import os
5  import random
6  import re
7  import sys
8  import ast
9
10
11 #
12 # Complete the 'is_palindrome' function below.
13 #
14 # The function is expected to return a BOOLEAN.
15 # The function accepts STRING s as parameter.
16 #
17
18 def is_palindrome(s):
19     left, right = 0, len(s) - 1
20
21     while left < right:
22         if s[left].lower() != s[right].lower():
23             return False
24         left += 1
25         right -= 1
26     return True
27
28 if __name__ == '__main__':
29     input_data = sys.stdin.read().strip()
30     arr = ast.literal_eval(input_data)
31
32     result = is_palindrome(arr)
```

33

print(result)

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Pass/Fail Case	Easy	Hidden	Success	5	0.0268 sec	10.6 KB

 No comments.