100

Venkata Naga Sri Sai Pranavi Kolipaka

Other

PDF generated at: 6 Jun 2025 23:09:08 UTC

View this report on HackerRank

Score

100% • 80 / 80

scored in TIP102: Unit 1 Version B (Standard) - Summer 2025 in 25 min 45 sec on 6 Jun 2025 15:41:30 PDT

Candidate Information

Email	kolipakavnssaipranavi@gmail.com
Test	TIP102: Unit 1 Version B (Standard) - Summer 2025
Candidate Packet	View
Taken on	6 Jun 2025 15:41:30 PDT
Time taken	25 min 45 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	Check For X Coding	2 min 8 sec	-	20/20	-
✓	2	First Repeating Substring Coding	9 min 21 sec	-	20/20	-
✓	3	Special Array Coding	6 min 32 sec	-	20/20	-

Multiple Choice + Debugging • 20 / 20

Status	No.	Question	Time Taken	Skill	Score	Code Quality
✓	4	What is the output of the following code snippet? Multiple Choice	1 min 11 sec	-	5/5	-
✓	5	What is the output of the following code snippet? Multiple Choice	2 min 45 sec	-	5/5	-
✓	6	What is the output? Multiple Choice	2 min 35 sec	-	5/5	-
✓	7	Find the bug! Coding	56 sec	-	5/5	-

1. Check For X

✓ Correct

Coding

Question description

Given a list of integers `nums`, return `True` if there are at least `x` numbers in `nums` that are greater than or equal to the length of the `nums`. The value of `x` is the length of the list.

Return `False` otherwise.

Example 1:
Input: lst = [1, 2, 3, 4]

Expected Output: False

Example 2:
Input: [4, 4, 4, 4]
Expected Output: True

Example 3:
Input: [5, 6, 7, 8]
Expected Output: True

Candidate's Solution

Language used: Python 3

```
#!/bin/python3

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



#
# Complete the 'check_list' function below.
#
# The function is expected to return a BOOLEAN.
# The function accepts INTEGER_ARRAY nums as parameter.
#

def check_list(nums):
    # Write your code here
    n = len(nums)
    for num in nums:
        if(num < n):
            return False
    return True

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:
            nums = ast.literal_eval(input_str)

            result = check_list(nums)

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

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Basic Case	Easy	Hidden	Success	0	0.027 sec	10.9 KB
All Elements Equal to Length	Easy	Hidden	Success	0	0.0316 sec	11 KB
Numbers Greater Than or Equal to Length	Easy	Hidden	Success	0	0.0311 sec	11 KB
No Elements Greater Than or Equal to Length	Easy	Hidden	Success	0	0.0298 sec	10.8 KB
Testcase 5	Easy	Hidden	Success	0	0.0279 sec	10.9 KB
Empty List	Easy	Hidden	Success	0	0.0284 sec	10.8 KB

Mixed Values with Insufficient Count	Easy	Hidden	Success	0	0.0274 sec	10.9 KB
Testcase 7	Easy	Hidden	Success	0	0.0283 sec	11 KB
Pass/Fail Case	Easy	Hidden	Success	20	0.0277 sec	10.9 KB

No comments.

2. First Repeating Substring

Correct

Coding

Question description

Given a string s, find the first substring of length k that appears exactly twice in the string and return its starting index. If no such substring exists, return -1

Example usage:
Input: s = "abcabcabc" k = 3
Expected Output: 0

Example Usage 2:
Input: s = "banana" k = 2
Expected Output: 1

Candidate's Solution

Language used: Python 3

```
1 #!/bin/python3
2
```

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


#
# Complete the 'first_repeating_substring' function below.
#
# The function is expected to return an INTEGER.
# The function accepts following parameters:
#  1. STRING s
#  2. INTEGER k
#

def first_repeating_substring(s, k):
    # Write your code here
    n = len(s)
    res = []
    for i in range(n-k+1):
        sub = s[i:i+k]
        for ind, str in res:
            if str == sub:
                return ind
        res.append((i, sub))
    return -1


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

    for line in input_str:
        s, k = line.split(', ')
        s = s.strip('"')
        k = int(k)
        result = first_repeating_substring(s, k)
        outfile.write(str(result) + '\n')
    outfile.close()
```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Basic Case with No Repetition	Easy	Hidden	Success	0	0.027 sec	10.9 KB
Test Case with Overlapping Substrings	Easy	Hidden	Success	0	0.0319 sec	10.6 KB
Test Case with No Repeating Substring	Easy	Hidden	Success	0	0.0277 sec	10.9 KB
Test Case where k is Greater than String Length	Easy	Hidden	Success	0	0.0314 sec	10.6 KB
Test Case with a Single Occurrence of Substring	Easy	Hidden	Success	0	0.0359 sec	10.9 KB
Test case with empty string	Easy	Hidden	Success	0	0.0284 sec	10.9 KB
Test case with a single character string	Easy	Hidden	Success	0	0.0306 sec	10.8 KB
Pass/Fail Case	Easy	Hidden	Success	20	0.0283 sec	10.8 KB

No comments.

3. Special Array

Correct

Coding

Question description

You are given a list `nums` of non-negative integers. `nums` is considered **special** if there exists a number `x` such that there are **exactly** `x` numbers in `nums` that are **greater than or equal to** `x`.

Notice that `x` **does not** have to be an element in `nums`.

Return `x` *if the list is **special**, otherwise, return* `-1`. It can be proven that if `nums` is special, the value for `x` is **unique**.
Note: Students will need to use the built-in function sort().

Example 1:
Input: nums = [3,5]
Output: 2
Explanation: There are 2 values (3 and 5) that are greater than or equal to 2.

Example 2:
Input: nums = [0,0]
Output: -1
Explanation: No numbers fit the criteria for x.
If x = 0, there should be 0 numbers >= x, but there are 2.
If x = 1, there should be 1 number >= x, but there are 0.
If x = 2, there should be 2 numbers >= x, but there are 0.
x cannot be greater since there are only 2 numbers in nums.

Example 3:
Input: nums = [0,4,3,0,4]
Output: 3
Explanation: There are 3 values that are greater than or equal to 3.

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 #
13 # Complete the 'special_array' function below.
14 #
15 # The function is expected to return an INTEGER.
16 # The function accepts INTEGER_ARRAY nums as parameter.
17 #
18
19 def special_array(nums):
20     # Write your code here
21     nums.sort(reverse=True)
22     for i in range(1, len(nums) + 1):
23         if nums[i-1] >= i and (i == len(nums) or nums[i] < i):
24             return i
25     return -1
26
27 if __name__ == '__main__':
28     outfile = open(os.environ['OUTPUT_PATH'], 'w')
29     input_lines = sys.stdin.read().strip().splitlines()
30
31     for input_str in input_lines:
32         if input_str.strip() == "":
33             continue
34
35         try:
36             nums = ast.literal_eval(input_str)
37
38             result = special_array(nums)
39
40             outfile.write(str(result) + '\n')
41         except (ValueError, SyntaxError):
42             print("Error: Invalid input")
43     outfile.close()
```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Basic Case with Distinct Values	Easy	Hidden	Success	0	0.0273 sec	11 KB
All Zeros	Easy	Hidden	Success	0	0.0282 sec	10.6 KB
Mixed Values with Multiple Valid x	Easy	Hidden	Success	0	0.0272 sec	11 KB
Case with Values in Descending Order	Easy	Hidden	Success	0	0.0278 sec	11 KB
Single Value Greater than Length	Easy	Hidden	Success	0	0.0296 sec	11 KB
Single Value Less than Length	Easy	Hidden	Success	0	0.0282 sec	10.9 KB
All Same Values Greater than Length	Easy	Hidden	Success	0	0.0271 sec	10.8 KB
Large List with No Valid x	Easy	Hidden	Success	0	0.0271 sec	11 KB
Decreasing Values with No Valid x	Easy	Hidden	Success	0	0.0298 sec	10.9 KB

Empty List	Easy	Hidden	Success	0	0.0378 sec	11 KB
Pass/Fail Case	Easy	Hidden	Success	20	0.0303 sec	10.9 KB

No comments.

4. What is the output of the following code snippet?

Correct

Multiple Choice

Question description

```
lst = [1, 2, 3, 4]
lst[3] = 'banana'
print(lst)
```

Candidate's Solution

Options: (Expected answer indicated with a tick)

- [1, 2, 'banana', 4]
- ✓ [1, 2, 3, 'banana']

Throws an error because all elements of a list must have the same data type.

Throws an error because index 3 does not exist within lst.

No comments.

5. What is the output of the following code snippet?

Correct

Multiple Choice

Question description

```
def mystery_function(s, specific_digits):
    count = 0
    for char in s:
        if char in specific_digits:
            count += 1
    return count

result = mystery_function("There are 2 apples, 3 bananas, 5 cherries, and 7 dates.", "2378")
print(result)
```

Candidate's Solution

Options: (Expected answer indicated with a tick)

1

- 2
- 3 ✓
- 4

No comments.

6. What is the output?

Correct

Multiple Choice

Question description

```
def mystery_function(n):
    count = 0
    while count < n:
        count += 1
    return count

result = mystery_function(5)
print(result)
```

Candidate's Solution

Options: (Expected answer indicated with a tick)

- 4
- 5 ✓
- 6
- 10

No comments.

7. Find the bug!

Correct

Coding

Question description

The provided code incorrectly implements the function `sort_by_parity`. Given a list of integers `nums`, `sort_by_parity` should return a new list that moves all of the even integers to the beginning of the list followed by all of the odd integers. Relative order of odd integers and even integers does not need to be maintained. Identify any bug(s) within the given implementation and correct the code so that it successfully passes the provided test cases.

```
def sort_by_parity(nums):
    evens = []
    odds = []
    for num in nums:
        if num % 2 == 0:
            evens.append(num)
        if num % 2 == 1:
            odds.append(num)
```

return odds + evens

Candidate's Solution

Language used: Python 3

```python
#!/bin/python3

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

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

def sort_by_parity(nums):
    evens = []
    odds = []
    for num in nums:
        if num % 2 == 0:
            evens.append(num)
        if num % 2 == 1:
            odds.append(num)
    return evens + odds
if __name__ == '__main__':
    input_str = sys.stdin.read().strip()
    input_list = ast.literal_eval(input_str)
    result = sort_by_parity(input_list)
    print(result)
```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED

Pass/Fail Case	Easy	Hidden	Success	5	0.0276 sec	10.9 KB

No comments.