100

Kibru Menore
Other

Score

100% • 80 / 80
scored in TIP102: Unit 1 Version B (Standard) - Spring 2025 in 66 min 1 sec on 23 Feb 2025 17:11:56 PST

Candidate Information

Email	Kibrussmenore@gmail.com
Test	TIP102: Unit 1 Version B (Standard) - Spring 2025
Candidate Packet	View ⌤
Taken on	23 Feb 2025 17:11:56 PST
Time taken	66 min 1 sec/ 90 min
Personal Member ID	121498
Email Address with CodePath	Kibrussmenore@gmail.com
Github username with CodePath	kibrus
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
✓	1	Check For X Coding	13 min 48 sec	-	20/20
✓	2	First Repeating Substring Coding	27 min 21 sec	-	20/20

Status	No.	Question	Time Taken	Skill	Score
✓	3	Special Array Coding	19 min 2 sec	-	20/20

Multiple Choice + Debugging • 20 / 20

Status	No.	Question	Time Taken	Skill	Score
✓	4	What is the output of the following code snippet? Multiple Choice	1 min 16 sec	-	5/5
✓	5	What is the output of the following code snippet? Multiple Choice	54 sec	-	5/5
✓	6	What is the output? Multiple Choice	1 min 11 sec	-	5/5
✓	7	Find the bug! Coding	2 min 9 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**

```python
#!/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):
    count = 0
    length = len(nums)

    for num in nums:
        if num >= length:
            count += 1
    if count == length:
        return True
    return False
```

```
29
30  if __name__ == '__main__':
31      outfile = open(os.environ['OUTPUT_PATH'], 'w')
32      input_lines = sys.stdin.read().strip().splitlines()
33
34      for input_str in input_lines:
35          if input_str.strip() == "":
36              continue
37
38          try:
39              nums = ast.literal_eval(input_str)
40
41              result = check_list(nums)
42
43              outfile.write(str(result) + '\n')
44          except (ValueError, SyntaxError):
45              print("Error: Invalid input")
46      outfile.close()
```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Basic Case	Easy	Hidden	Success	0	0.0281 sec	11.3 KB
All Elements Equal to Length	Easy	Hidden	Success	0	0.0279 sec	11.3 KB
Numbers Greater Than or Equal to Length	Easy	Hidden	Success	0	0.0279 sec	11.4 KB
No Elements Greater Than or Equal to Length	Easy	Hidden	Success	0	0.029 sec	11.3 KB
Testcase 5	Easy	Hidden	Success	0	0.0291 sec	11.3 KB

Empty List	Easy	Hidden	Success	0	0.0272 sec	11.2 KB
Mixed Values with Insufficient Count	Easy	Hidden	Success	0	0.0282 sec	11.3 KB
Testcase 7	Easy	Hidden	Success	0	0.0281 sec	11.4 KB
Pass/Fail Case	Easy	Hidden	Success	20	0.029 sec	11.4 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**

```python
#!/bin/python3

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):
    l, r = 0, k
    for i in range(len(s)):
        if s[l:r] == s[r: r+k]:
            return l
        l += 1
        r +=1
    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	0.0287 sec	11.4 KB
Test Case with Overlapping Substrings	Easy	Hidden	Success	0	0.0256 sec	11.4 KB
Test Case with No Repeating Substring	Easy	Hidden	Success	0	0.0277 sec	11.4 KB
Test Case where k is Greater than String Length	Easy	Hidden	Success	0	0.0285 sec	11.3 KB
Test Case with a Single Occurrence of Substring	Easy	Hidden	Success	0	0.0292 sec	11.4 KB
Test case with empty string	Easy	Hidden	Success	0	0.0279 sec	11.2 KB
Test case with a single character string	Easy	Hidden	Success	0	0.028 sec	11.4 KB
Pass/Fail Case	Easy	Hidden	Success	20	0.0283 sec	11.4 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
```

```python
 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      nums.sort()
21      n = len(nums)
22
23      for i in range(n):
24          x = n-i
25          if nums[i] >= x and (i == 0 or nums[i-1]<x):
26              return x
27      return -1
28
29  if __name__ == '__main__':
30      outfile = open(os.environ['OUTPUT_PATH'], 'w')
31      input_lines = sys.stdin.read().strip().splitlines()
32
33      for input_str in input_lines:
34          if input_str.strip() == "":
35              continue
36
37          try:
38              nums = ast.literal_eval(input_str)
39
40              result = special_array(nums)
41
42              outfile.write(str(result) + '\n')
43          except (ValueError, SyntaxError):
44              print("Error: Invalid input")
45      outfile.close()
```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Basic Case with Distinct Values	Easy	Hidden	Success	0	0.0305 sec	11.4 KB
All Zeros	Easy	Hidden	Success	0	0.0386 sec	11.3 KB
Mixed Values with Multiple Valid x	Easy	Hidden	Success	0	0.0311 sec	11.4 KB
Case with Values in Descending Order	Easy	Hidden	Success	0	0.0311 sec	11.4 KB
Single Value Greater than Length	Easy	Hidden	Success	0	0.0289 sec	11.4 KB
Single Value Less than Length	Easy	Hidden	Success	0	0.0352 sec	11.3 KB
All Same Values Greater than Length	Easy	Hidden	Success	0	0.0279 sec	11.3 KB
Large List with No Valid x	Easy	Hidden	Success	0	0.0375 sec	11.4 KB
Decreasing Values with No Valid x	Easy	Hidden	Success	0	0.0325 sec	11.2 KB

Empty List	Easy	Hidden	Success	0	0.0388 sec	11.4 KB
Pass/Fail Case	Easy	Hidden	Success	20	0.0285 sec	11.4 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
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 # Complete the 'sort_by_parity' function below.
12 #
13 # The function is expected to return an INTEGER.
14 # The function accepts INTEGER_ARRAY nums as parameter.
15 #
16
17 def sort_by_parity(nums):
18     evens = []
19     odds = []
20     for num in nums:
21         if num % 2 == 0:
22             evens.append(num)
23         if num % 2 == 1:
24             odds.append(num)
25     return evens + odds
26 if __name__ == '__main__':
27     input_str = sys.stdin.read().strip()
28     input_list = ast.literal_eval(input_str)
29     result = sort_by_parity(input_list)
30     print(result)
```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED

Pass/Fail Case	Easy	Hidden	Success	5	0.0271 sec	11.3 KB

⊘ **No comments.**