100

Rajasekhar Reddy Kolagotla

Other

PDF generated at: 9 Jun 2025 00:24:02 UTC

View this report on HackerRank

Score

100% • 80 / 80

scored in TIP102: Unit 1 Version A (Standard) - Summer 2025 in 20 min 18 sec on 8 Jun 2025 17:02:04 PDT

Candidate Information

Email	rajasekhar1131997@gmail.com
Test	TIP102: Unit 1 Version A (Standard) - Summer 2025
Candidate Packet	View
Taken on	8 Jun 2025 17:02:04 PDT
Time taken	20 min 18 sec/ 90 min
Personal Member ID	126663
Email Address with CodePath	rajasekhar1131997@gmail.com
Github username with CodePath	Rajasekhar1131997
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	Unique Coding	2 min 58 sec	-	20/20	-
✓	2	Needle in Haystack Coding	4 min 41 sec	-	20/20	-

✓	3	Flowerbed Coding	7 min 27 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 1 sec	-	5/5	-
✓	5	What is the output of the following code snippet? Multiple Choice	1 min 6 sec	-	5/5	-
✓	6	What is the output of the following code snippet? Multiple Choice	26 sec	-	5/5	-
✓	7	Find the bug! Coding	2 min 16 sec	-	5/5	-

1. Unique

✓ Correct

Coding

Question description

Given a string `s`, return `True` if every character in the string is unique. Return `False` if any characters in `s` are repeated.

Example 1
Input: s = "abcdef"
Expected Output: True

Example 2
Input: s = "aabbcc"
Output: False

Example 3
Example Input: s = ""
Expected Output: True

Candidate's Solution

Language used: **Python 3**

```python
#!/bin/python3

import math
import os
import random
import re
import sys



def has_all_unique_characters(s):
    unique = set(s)
    return len(unique) == len(s)
if __name__ == "__main__":
    # Read the entire input
    input_data = sys.stdin.read().strip().split("\n")

    results = []
    for line in input_data:
        # Handle input with quotes (e.g., "abcdef" or "")
        s = line.strip()
        if s == '""':  # Interpret "" as an actual empty string
            s = ""

        # Redirect debugging output to stderr to suppress student print statements
        original_stdout = sys.stdout
        try:
            sys.stdout = sys.stderr  # Redirect stdout to stderr for debugging prints
            # Call the function here
            result = has_all_unique_characters(s)
        finally:
            sys.stdout = original_stdout  # Restore stdout

        # Collect the result for this test case
        results.append(result)

    # Print all results (one per line)
    for res in results:
        print(res)
```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Is Unique	Easy	Hidden	Success	0	0.023 sec	10.1 KB
Is Not Unique	Easy	Hidden	Success	0	0.0233 sec	10 KB
Upper/lower	Easy	Hidden	Success	0	0.0204 sec	10.1 KB
Empty String	Easy	Hidden	Success	0	0.0225 sec	10.1 KB
Single Char	Easy	Hidden	Success	0	0.0226 sec	10 KB

Pass/Fail Test Case	Easy	Hidden	Success	20	0.0273 sec	10 KB

ⓘ No comments.

2. Needle in Haystack

✓ Correct

Coding

Question description

Given two strings `needle` and `haystack`, return the index of the first occurrence of needle in haystack, or `-1` if `needle` is not part of `haystack`.

```
Example 1:
Input: haystack = "sadbutsad", needle = "sad"
Output: 0
Explanation: "sad" occurs twice, starting at indices 0 and 6.
The first occurrence is at index 0, so we return 0.

Example 2:
Input: haystack = "leetcode", needle = "leeto"
Output: -1
Explanation: "leeto" did not occur in "leetcode", so we return -1.

Example 3:
Input: haystack = "mad" needle = "madden"
Needle is longer than haystack, so we return -1.
```

Candidate's Solution

Language used: **Python 3**

```
#!/bin/python3

```

```python
import math
import os
import random
import re
import sys



#
# Complete the 'find_needle' function below.
#
# The function is expected to return an INTEGER.
# The function accepts following parameters:
#  1. STRING haystack
#  2. STRING needle
#

def find_needle(haystack, needle):
    return haystack.find(needle)
if __name__ == "__main__":
    # Read the entire input
    input_data = sys.stdin.read().strip().split("\n")

    results = []
    for i in range(0, len(input_data), 2):
        # Each test case contains two lines: haystack and needle
        haystack = input_data[i].strip()
        needle = input_data[i + 1].strip()

        # Redirect debugging output to stderr to suppress student print statements
        original_stdout = sys.stdout
        try:
            sys.stdout = sys.stderr  # Redirect stdout to stderr for debugging prints
            # Call the function here
            result = find_needle(haystack, needle)
        finally:
            sys.stdout = original_stdout  # Restore stdout

        # Collect the result for this test case
        results.append(result)

    # Print all results (one per line)
    for res in results:
        print(res)
```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Needle in Haystack at 0	Easy	Hidden	Success	0	0.0238 sec	10.1 KB
Needle not in Haystack	Easy	Hidden	Success	0	0.0258 sec	10.3 KB
Haystack smaller than needle	Easy	Hidden	Success	0	0.0285 sec	10.3 KB
Empty haystack	Easy	Hidden	Success	0	0.0244 sec	10.1 KB
Empty Strings	Easy	Hidden	Success	0	0.0303 sec	10.3 KB
First occurence not at 0	Easy	Hidden	Success	0	0.0235 sec	10.1 KB
Pass/Fail Testcases	Easy	Hidden	Success	20	0.0255 sec	10.1 KB

ⓘ No comments.

3. Flowerbed

✓ Correct

Coding

Question description

You have a single long flowerbed in which some of the plots are planted, and some are not. However, flowers cannot be planted **directly adjacent** to another flower.

Given an integer array `flowerbed` containing `0`'s and `1`'s, where `0` means empty and `1` means not empty, and an integer `n`, return `True` *if* `n` *new flowers can be planted in the* `flowerbed` *without violating the no-adjacent-flowers rule and* `False` *otherwise*.

```
Example 1:
Input: flowerbed = [1,0,0,0,1], n = 1
Output: True

Example 2:
Input: flowerbed = [1,0,0,0,1], n = 2
Output: False
```

Hint: When deciding where to plant a new flower, focus on each plot in the `flowerbed` and check its neighboring plots. You only need to consider the plot directly before and directly after the current plot to determine if a flower can be planted there. Remember that the flowerbed is linear, so you don't need to worry about wrapping around.

Candidate's Solution

Language used: **Python 3**

```python
#!/bin/python3

import math
import os
import random
import re
import sys



#
# Complete the 'can_place_flowers' function below.
#
# The function is expected to return a BOOLEAN.
# The function accepts following parameters:
#  1. INTEGER_ARRAY flowerbed
#  2. INTEGER n
#

def can_place_flowers(flowerbed, n):
    count = 0
    flowerbed_length = len(flowerbed)
    for i in range(flowerbed_length):
        if flowerbed[i] == 0:
            left = (i == 0 or flowerbed[i-1] == 0)
            right = (i == flowerbed_length -1 or flowerbed[i+1] == 0)
            if left and right:
                flowerbed[i] = 1
                count+=1
                if count >= n:
                    return True
    return count >= n
if __name__ == "__main__":
    input_data = sys.stdin.read().strip().split("\n")

    results = []
    for i in range(0, len(input_data), 2):
        flowerbed_line = input_data[i].strip()
        n = int(input_data[i + 1].strip())

        if flowerbed_line == "[]":
            flowerbed = []
        else:
            flowerbed = list(map(int, flowerbed_line.strip("[]").split(",")))

        # Redirect debugging output to stderr
        original_stdout = sys.stdout
        try:
            sys.stdout = sys.stderr
            result = can_place_flowers(flowerbed, n)
        finally:
            sys.stdout = original_stdout

        results.append(result)

    for res in results:
        print(res)
```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED

Can Place	Easy	Hidden	Success	0	0.0313 sec	10.1 KB
Cannot Place	Easy	Hidden	Success	0	0.0269 sec	10.1 KB
Nothing To Add	Easy	Hidden	Success	0	0.0255 sec	10 KB
Can Place Pt2	Easy	Hidden	Success	0	0.0233 sec	10.3 KB
Empty Flowerbed w/ no Addition	Easy	Hidden	Success	0	0.0227 sec	10.1 KB
Fulll Flowerbed	Easy	Hidden	Success	0	0.027 sec	9.88 KB
All Empty	Easy	Hidden	Success	0	0.0251 sec	10 KB
Pass/Fail Testcases	Easy	Hidden	Success	20	0.0239 sec	10.1 KB

No comments.

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

✓ Correct

Multiple Choice

Question description

```
name = "codepath"
name[0] = "C"
print(name)
```

Candidate's Solution

Options: (Expected answer indicated with a tick)

○ Codepath

○ Ccodepath

○ C

● d. Throws an error because strings are immutable and characters cannot be changed once the string is created. ✓

ⓘ No comments.

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

✓ Correct

Multiple Choice

Question description

```
def mystery_function(s):
    count = 0
    for i in range(1, len(s)):
        if s[i] == s[i - 1]:
            count += 1
    return count

result = mystery_function("AABBAB")
print(result)
```

Candidate's Solution

Options: (Expected answer indicated with a tick)

○ 1

● 2 ✓

○ 3

○ 4

ⓘ No comments.

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

✓ Correct

Multiple Choice

Question description

```
def mystery_function(lst, threshold):
    total = 0
    i = 0
    while i < len(lst) and total + lst[i] <= threshold:
        total += lst[i]
        i += 1
    return total

result = mystery_function([1, 2, 3, 4, 5], 7)
print(result)
```

Candidate's Solution

Options: (Expected answer indicated with a tick)

○ 3

● 6 ✓

○ 7

○ 10

ⓘ No comments.

7. Find the bug!

✓ Correct

Coding

Question description

The provided code incorrectly implements the function `reverse_lst` which should accept a list `lst` and return the original list with the elements in reverse order.

```
def reverse_lst(lst):
    left = 0
    right = len(lst) - 1

    while left < right:
        lst[left] = lst[right]
        lst[right] = lst[left]
        left -= 1
        right += 1

    return lst


lst = [1, 2, 3, 4, 5]
print(reverse_lst(lst))

lst = [10, 20, 30, 40]
print(reverse_lst(lst))
```

Identify the bug(s) within the given implementation and select the corrected code that will successfully reverse the list.

Candidate's Solution

Language used: **Python 3**

```
#!/bin/python3

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


```

```python

#
# Complete the 'reverse_lst' function below.
#
# The function is expected to return an INTEGER_ARRAY.
# The function accepts INTEGER_ARRAY lst as parameter.
#

def reverse_lst(lst):
    left = 0
    right = len(lst) - 1

    while left < right:
        lst[left], lst[right] = lst[right], lst[left]
        left += 1
        right -= 1

    return lst

if __name__ == '__main__':
    input_str = sys.stdin.read().strip()
    # Convert the input string to a list of integers
    input_list = ast.literal_eval(input_str)
    # Reverse the list
    result = reverse_lst(input_list)
    # Print the reversed list
    print(result)
```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Pass/Fail Case	Easy	Hidden	Success	5	0.0273 sec	10.9 KB

ⓘ No comments.