74

Muhammad Tanveer

Software Engineer

PDF generated at: 13 Aug 2024 02:42:40 UTC

View this report on HackerRank

Score

74% • 88 / 120

scored in CodePath TIP103: Unit 9 Assessment - Summer 2024 in 81 min 50 sec on 12 Aug 2024 18:19:14 PDT

Candidate Information

Email	tanveerm176@gmail.com
Test	CodePath TIP103: Unit 9 Assessment - Summer 2024
Candidate Packet	View
Taken on	12 Aug 2024 18:19:14 PDT
Time taken	81 min 50 sec/ 90 min
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

Status	No.	Question	Time Taken	Skill	Score
✓	1	Recursion, Memoization, and Iteration Multiple Choice	11 min 5 sec	-	5/5
✓	2	First Element Retrieval Multiple Choice	2 min 21 sec	-	5/5
⊗	3	Hash Tables Multiple Choice	48 sec	-	0/5
✓	4	Trees Multiple Choice	4 min 59 sec	-	3.75/5

⊗	5	Graph Traversal Multiple Choice	5 min 23 sec	-	0/5
✓	6	Big O Notation Multiple Choice	3 min 1 sec	-	5/5
✓	7	Validate Binary Search Tree Coding	27 min 40 sec	-	40/40
✓	8	Zigzag Conversion Coding	26 min 7 sec	-	30/50

1. Recursion, Memoization, and Iteration

✓ Correct

Multiple Choice

Question description

Consider the following Python implementations of the fibonacci function:

```
def fib1(self, n: int) -> int:

  if n == 0 or n == 1:

    return n

  return fib1(n - 1) + fib1(n - 2)



memo = {0: 0, 1: 1}
```

```
def fib2(self, n: int) -> int:
 if n in memo:
  return memo[n]
 memo[n] = fib2(n - 1) + fib2(n - 2)
 return memo[n]


def fib3(self, n: int) -> int:
 if n == 0 or n == 1:
  return n
 a, b = 0, 1
 for i in range(n - 1):
  a, b = b, a+b
 return b
```

Which of the following statements are true? Pick one or more options.

Candidate's Solution

Options: (Expected answer indicated with a tick)

☑ fib1 has higher time complexity than fib2 ✓

□ fib1 has higher space complexity than fib2

□ fib3 has lower time complexity than both fib1 and fib2

☑ fib3 has lower space complexity than both fib1 and fib2 ✓

ⓘ No comments.

2. First Element Retrieval

✓ Correct

Multiple Choice

Question description

Assume you add n items, one at a time, to a data structure. From which data structure can you then access the first item in a single constant-time operation? Pick one or more options.

Candidate's Solution

Options: (Expected answer indicated with a tick)

○ A stack

● A queue ✓

○ A heap

ⓘ No comments.

3. Hash Tables

⊗ Incorrect

Multiple Choice

Question description

Consider the following Java code that uses a hash table:

```
Map<String, Character> map = new HashMap<>();
map.put("hello", 'h');

map.put("goodbye", 'g');

System.out.println(map.get("hello"));
```

What will happen if "hello" and "goodbye" have the same hash values (i.e., collide)? Pick one option.

Candidate's Solution

Options: (Expected answer indicated with a tick)

○ The final line will always print: h ✓

○ The final line will always print: g

○ The final line will always print: null

● The final line could print either g or h.

○ There will be an error/exception on the final line.

No comments.

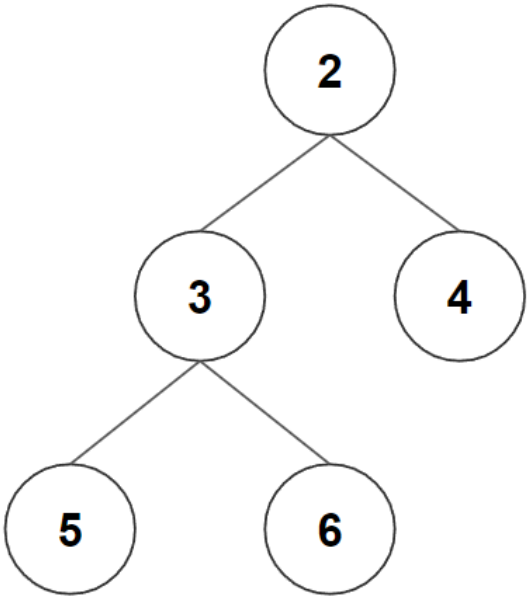
4. Trees

✓ Partially correct

Multiple Choice

Question description

Consider this tree:



Which of the following terms accurately describe the tree? Pick one or more options.

Candidate's Solution

Options: (Expected answer indicated with a tick)

- ☑ A binary tree ✓
- □ A binary search tree
- ☑ A balanced binary tree ✓
- ☑ A min-heap ✓
- □ A max-heap
- □ A graph ✓

ⓘ No comments.

5. Graph Traversal

ⓧ Incorrect

Multiple Choice

Question description

Consider an implementation of DFS that does not keep track of which nodes have already been visited:

```
def dfs(node):
  for neighbor in node.neighbors:
     dfs(neighbor)
```

When is it guaranteed to terminate? Pick one or more options.

Candidate's Solution

Options: (Expected answer indicated with a tick)

- ☑ If passed the root of a directed tree, this is guaranteed to terminate. ✓
- ☑ If passed a node in a directed acyclic graph, this is guaranteed to terminate. ✓
- ☑ If passed a node in a directed cyclic graph, this is guaranteed to terminate.
- ☐ None of the above.

ⓘ No comments.

6. Big O Notation

✓ Correct

Multiple Choice

Question description

Which of the following take O(log n) time? Pick one or more options.

Candidate's Solution

Options: (Expected answer indicated with a tick)

□ Reading the nth value in an array of size n.

□ Sorting an array of n integers.

☑ Adding a single item to a heap of size n. ✓

☑ Checking if an integer is present in a balanced binary search tree of size n. ✓

No comments.

7. Validate Binary Search Tree

✓ Correct

Coding

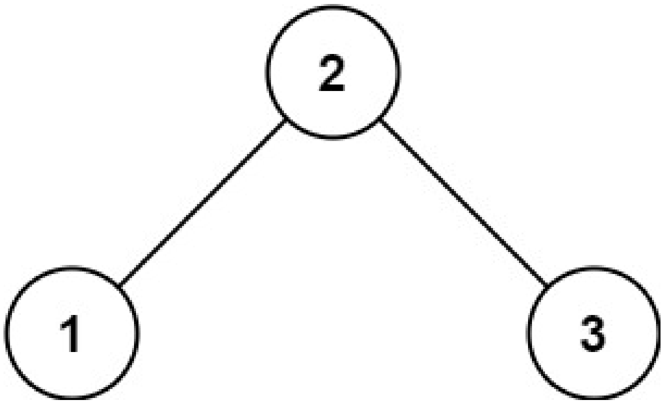
Question description

Given the `root` of a binary tree, *determine if it is a valid binary search tree (BST).*

A **valid BST** is defined as follows:

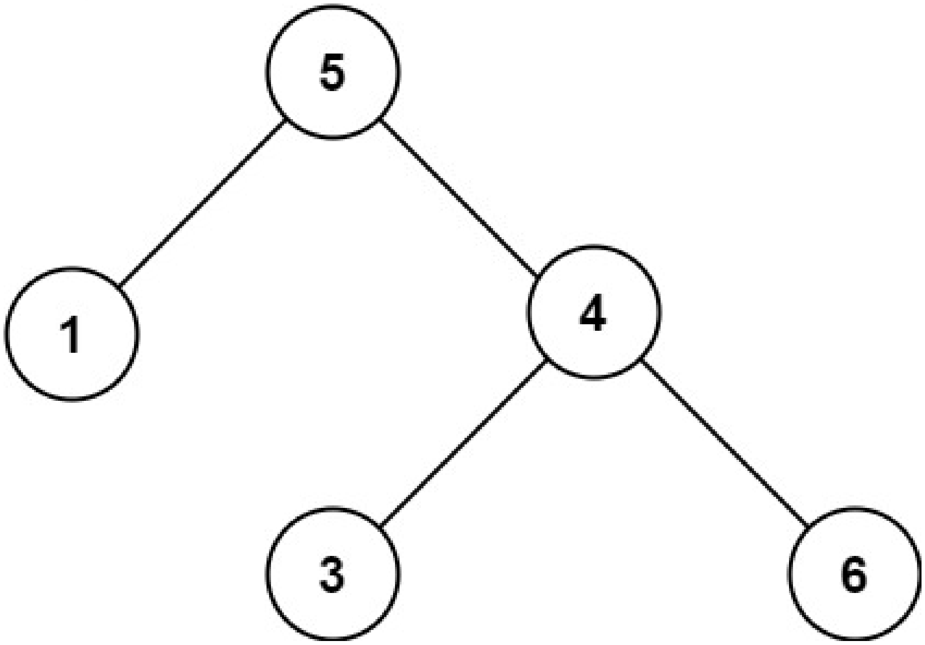
- The left
  subtree
  of a node contains only nodes with keys **less than** the node's key.
- The right subtree of a node contains only nodes with keys **greater than** the node's key.
- Both the left and right subtrees must also be binary search trees.

Example 1:



Input: root = [2,1,3]
Output: true

Example 2:



Input: root = [5,1,4,null,null,3,6]
Output: false

Explanation: The root node's value is 5 but its right child's value is 4.

Constraints:

- The number of nodes in the tree is in the range [1, 10^4].
- -2^{31} <= Node.val <= 2^{31} - 1

Candidate's Solution

Language used: **Python 3**

```python
#!/usr/bin/env python

class TreeNode:
    def __init__(self, x):
        self.val = x
        self.left = self.right = None

def input_binary_tree():
    input_values = input().split()
    index = 0
    num_nodes = int(input_values[index])
    index += 1
    if (num_nodes == 0):
        return None

    nodes = []
    current_parent_index = 0

    root = TreeNode(int(input_values[index]))
    index += 1
    nodes.append(root)

    for i in range(1, num_nodes, 2):
        left_val = int(input_values[index])
        index += 1
        if (left_val != -1):
            left = TreeNode(left_val)
            nodes.append(left)
            nodes[current_parent_index].left = left

        right_val = int(input_values[index])
        index += 1
        if (right_val != -1):
            right = TreeNode(right_val)
            nodes.append(right)
            nodes[current_parent_index].right = right

        current_parent_index += 1

    return root

def description(root):
    if root is None:
        return " "

    queue = []

    output = str(root.val)
    queue.append(root)
    cursor = 0

    while cursor < len(queue):
        node = queue[cursor]
        cursor += 1

        if node.left is not None:
            output += " " + str(node.left.val)
            queue.append(node.left)

        if node.right is not None:
            output += " " + str(node.right.val)
            queue.append(node.right)

    return output
"""
class TreeNode:
    def __init__(self, x):
        self.val = x
        self.left = self.right = None
"""

def isValidBST(root) :
    """
    Write your code here
    :type root: TreeNode
    :rtype: TreeNode
    """
    def isValidBST_helper(currNode, minVal, maxVal):
        if currNode is None:
            return True

        # check if curr node val is between minVal and maxVal
        if (currNode.val <= minVal) or (currNode.val >= maxVal):
            return False

        #recursively check if the right and left subtrees meet the criteria of a bst
        return isValidBST_helper(currNode.left, minVal, currNode.val) and isValidBST_helper(currNode.right, currNode.val, maxVal)

    # call helper function using root and the initial infinity bounds
    return isValidBST_helper(root, float('-inf'), float('inf'))


root = input_binary_tree()
result = isValidBST(root)
print(str(result).lower())

```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Testcase 0	Easy	Sample	Success	0	0.0245 sec	9.45 KB
Testcase 1	Easy	Sample	Success	0	0.0248 sec	9.4 KB
Testcase 2	Easy	Hidden	Success	10	0.0192 sec	9.3 KB
Testcase 3	Easy	Hidden	Success	10	0.0374 sec	9.14 KB
Testcase 4	Easy	Hidden	Success	10	0.0258 sec	9.2 KB
Testcase	Easy	Hidden	Success	10	0.0208 sec	9.08 KB

```
5
```

ⓘ No comments.

8. Zigzag Conversion

✓ Partially correct

Coding

Question description

The string "PAYPALISHIRING" is written in a zigzag pattern on a given number of rows like this: (you may want to display this pattern in a fixed font for better legibility)

```
P   A   H   N
A P L S I I G
Y   I   R
```

And then read line by line: "PAHNAPLSIIGYIR"
Write the code that will take a string and make this conversion given a number of rows:

```
string convert(string s, int numRows);
```

Example 1:

```
Input: s = "PAYPALISHIRING", numRows = 3
Output: "PAHNAPLSIIGYIR"
```

Example 2:

```
Input: s = "PAYPALISHIRING", numRows = 4
Output: "PINALSIGYAHRPI"
Explanation:
P     I    N
A   L S  I G
```

```
Y A   H R
P     I
```

Example 3:

Input: s = "A", numRows = 1
Output: "A"

Constraints:

- `1 <= s.length <= 1000`
- `s` consists of English letters (lower-case and upper-case), `','` and `'.'`.
- `1 <= numRows <= 1000`

Candidate's Solution

Language used: **Python 3**

```python
#!/bin/python3

import math
import os
import random
import re
import sys


#
# Complete the 'convert' function below.
#
# The function is expected to return a STRING.
# The function accepts following parameters:
#  1. STRING s
#  2. INTEGER numRows
#

def convert(s, numRows):
    # Write your code here
    if (numRows == 1) or (numRows >= len(s)):
        return s

    # create a list of rows
    rows = [''] *numRows
    # init current row
    curr_row = 0

    # last row
    last_row = numRows -1

    # process each char in string
    for char in s:
        rows[curr_row] = rows[curr_row] + char

        #if at first row or last row move currRow up or down
        if curr_row == 0:
            curr_row = curr_row + 1
        elif curr_row == last_row:
            curr_row = curr_row -1

    print(rows)

    result = ''
    for string_row in rows:
        result = string_row + result
    return result
if __name__ == '__main__':
    fptr = open(os.environ['OUTPUT_PATH'], 'w')

    s = input()

    numRows = int(input().strip())

    result = convert(s, numRows)

    fptr.write(result + '\n')

    fptr.close()

```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Testcase 0	Easy	Sample	Wrong Answer	0	0.0345 sec	10.5 KB
Testcase 1	Easy	Hidden	Wrong Answer	0	0.0353 sec	10.5 KB

Testcase 2	Easy	Hidden	Success	10	0.0446 sec	10.6 KB
Testcase 3	Easy	Hidden	Success	10	0.0347 sec	10.6 KB
Testcase 4	Easy	Hidden	Success	10	0.0455 sec	10.4 KB
Testcase 5	Easy	Hidden	Wrong Answer	0	0.0416 sec	10.7 KB

ⓘ **No comments.**