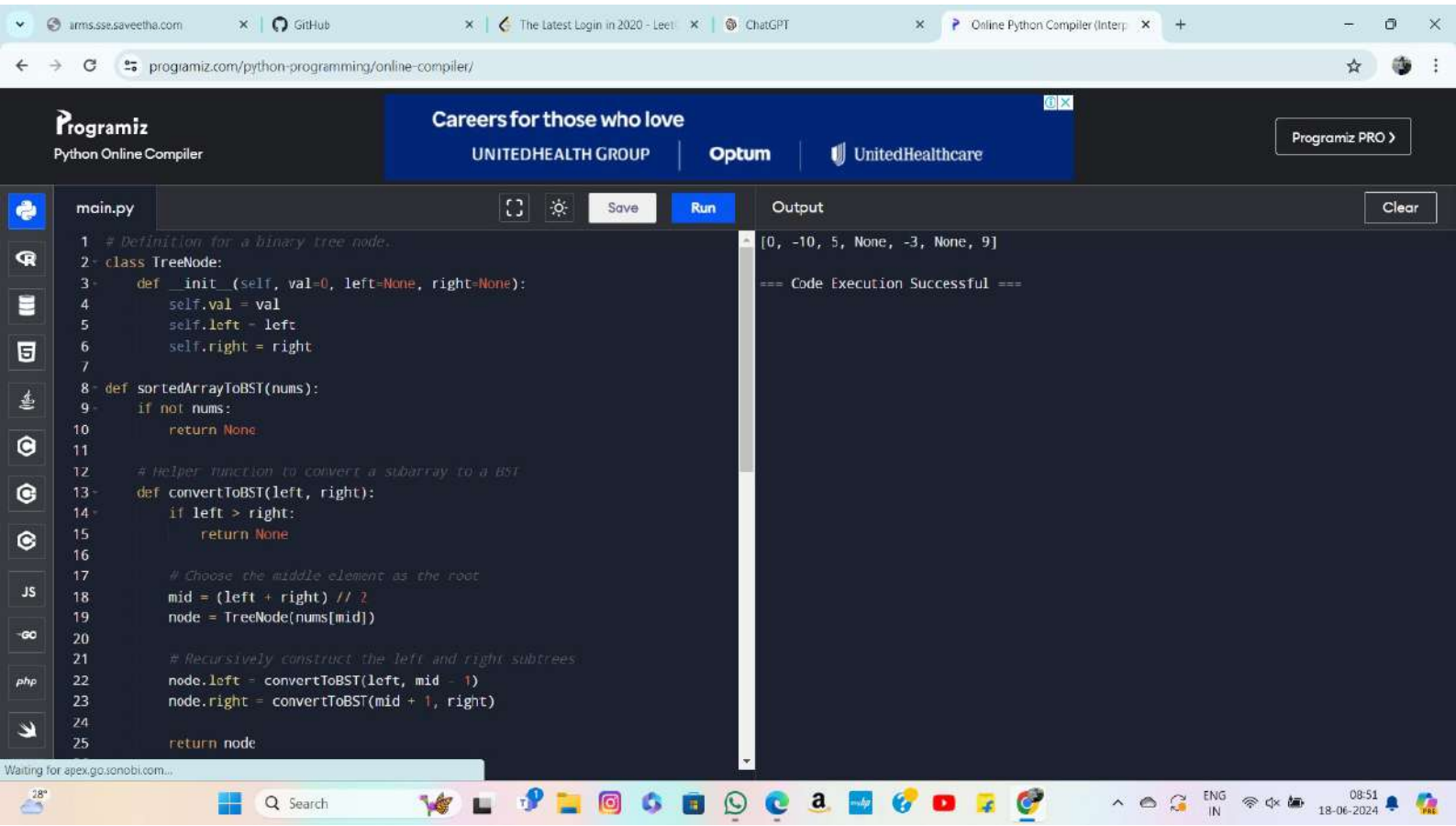
arms.sse.saveetha.com
GitHub
The Latest Login in 2020 - Leet
ChatGPT
Online Python Compiler (Interp
programiz.com/python-programming/online-compiler/
Programiz
Python Online Compiler
Careers for those who love
UNITEDHEALTH GROUP
Optum
UnitedHealthcare
Programiz PRO
main.py
Save
Run
Output
Clear
# Definition for a binary tree node.
class TreeNode:
def __init__(self, val=0, left=None, right=None):
self.val = val
self.left = left
self.right = right
def sortedArrayToBST(nums):
if not nums:
return None
# Helper function to convert a subarray to a BST
def convertToBST(left, right):
if left > right:
return None
# Choose the middle element as the root
mid = (left + right) // 2
node = TreeNode(nums[mid])
# Recursively construct the left and right subtrees
node.left = convertToBST(left, mid - 1)
node.right = convertToBST(mid + 1, right)
return node
[0, -10, 5, None, -3, None, 9]
=== Code Execution Successful ===
Waiting for apex.go.sonobi.com...
Search
ENG
IN
08:51
18-06-2024

main.py

Save Run

```python
def findSubstrings(words):
    result = []
    n = len(words)

    for i in range(n):
        for j in range(n):
            if i != j and words[i] in words[j]:
                result.append(words[i])
                break  # No need to check further for this word

    return result

# Example usage
words = ["mass", "as", "hero", "superhero"]
print(findSubstrings(words))  # Output: ["as", "hero"]
```

Output Clear

```
['as', 'hero']

=== Code Execution Successful ===
```

main.py

Save | Run

```python
from collections import deque

def updateMatrix(mat):
    rows, cols = len(mat), len(mat[0])
    distances = [[float('inf')] * cols for _ in range(rows)]
    queue = deque()

    # Initialize the queue with all positions of 0s and set their distance to 0
    for r in range(rows):
        for c in range(cols):
            if mat[r][c] == 0:
                distances[r][c] = 0
                queue.append((r, c))

    # Directions for moving up, down, left, right
    directions = [(0, 1), (1, 0), (0, -1), (-1, 0)]

    # Perform BFS
    while queue:
        r, c = queue.popleft()

        for dr, dc in directions:
            new_r, new_c = r + dr, c + dc

            if 0 <= new_r < rows and 0 <= new_c < cols:
                if distances[new_r][new_c] > distances[r][c] + 1:
```

Output | Clear

```
[[0, 0, 0], [0, 1, 0], [0, 0, 0]]
[[0, 0, 0], [0, 1, 0], [1, 2, 1]]

=== Code Execution Successful ===
```

main.py

Save Run

```python
def min_length_after_removals(nums):
    n = len(nums)
    left, right = 0, n - 1
    pairs_removed = 0

    while left < right:
        # Check if we can form a pair
        if nums[left] < nums[right]:
            # Remove the pair
            pairs_removed += 1
            left += 1
            right -= 1
        else:
            # If we cannot form a pair, move the right pointer to the left
            right -= 1

    # The remaining length is the original length minus twice the number of pairs removed
    return n - 2 * pairs_removed

# Example usage
nums = [1, 2, 3, 4]
print(min_length_after_removals(nums))  # Output: 0
```

Output

Clear

```
0

=== Code Execution Successful ===
```