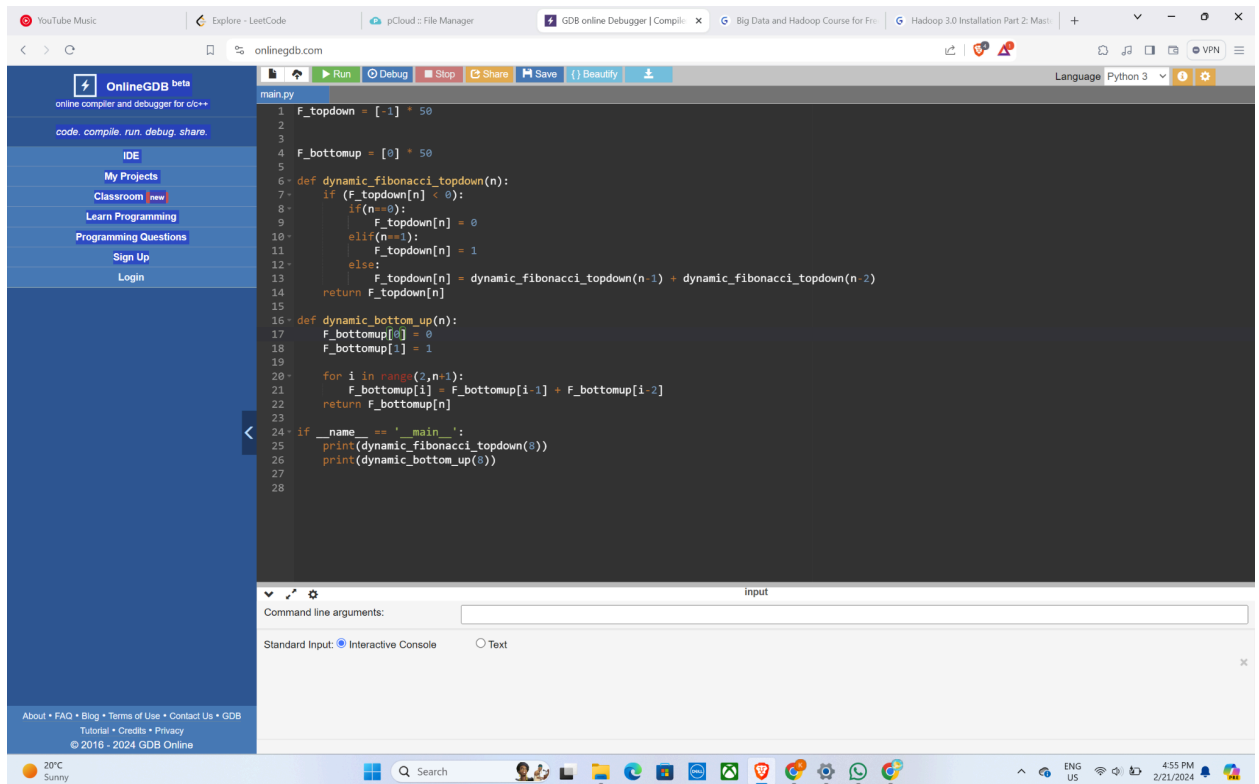


Day16 - Feb 21st 2024

1. Started my day as usual.
2. Received rejection mail from BP ..that demotivated the entire day
3. Started learning Dynamic programming and solved fibonacci using topdown and bottom up DP approaches



The screenshot shows the OnlineGDB website interface. The left sidebar contains navigation links: "OnlineGDB beta", "code, compile, run, debug, share.", "IDE", "My Projects", "Classroom new", "Learn Programming", "Programming Questions", "Sign Up", and "Login". The main editor area displays a Python file named "main.py" with the following code:

```
1 F_topdown = [-1] * 50
2
3
4 F_bottomup = [0] * 50
5
6 def dynamic_fibonacci_topdown(n):
7     if (F_topdown[n] < 0):
8         if (n==0):
9             F_topdown[n] = 0
10        elif (n==1):
11            F_topdown[n] = 1
12        else:
13            F_topdown[n] = dynamic_fibonacci_topdown(n-1) + dynamic_fibonacci_topdown(n-2)
14        return F_topdown[n]
15
16 def dynamic_bottom_up(n):
17     F_bottomup[0] = 0
18     F_bottomup[1] = 1
19
20     for i in range(2,n+1):
21         F_bottomup[i] = F_bottomup[i-1] + F_bottomup[i-2]
22     return F_bottomup[n]
23
24 if __name__ == '__main__':
25     print(dynamic_fibonacci_topdown(5))
26     print(dynamic_bottom_up(5))
27
28
```

Below the code editor, there is an "input" field and a "Standard Input" section with radio buttons for "Interactive Console" (selected) and "Text". The bottom of the image shows a Windows taskbar with the date and time "4:55 PM 2/21/2024".

4. And solved one leetcode problem

The screenshot shows a LeetCode submission for the problem "Intersection of Two Linked Lists". The submission is accepted, with a runtime of 77 ms and memory usage of 27.16 MB. The code is in Python3 and uses a set to find the intersection of two linked lists. The test case shows two lists intersecting at the value 8.

Runtime: 77 ms, Beats 82.45% of users with Python3

Memory: 27.16 MB, Beats 66.48% of users with Python3

Code (Python3):

```
class Solution:
    def getIntersectionNode(self, headA: ListNode, headB: ListNode) -> ListNode:
        # Definition for singly-linked list.
        # class ListNode:
        #     def __init__(self, x):
        #         self.val = x
        #         self.next = None

        # Create sets to store the nodes of both lists
        setA = set()
        setB = set()

        # Traverse list A and add nodes to setA
        curr = headA
        while curr:
            setA.add(curr)
            curr = curr.next

        # Traverse list B and check for intersection
        curr = headB
        while curr:
            if curr in setA:
                return curr
            curr = curr.next

        # If no intersection found, return None
        return None

# Testcase
A = headA
B = headB

while A != B:
    A = headB if A is None else A.next
```

Testcase:

Input: 8, [4,1,8,4,5], [5,6,1,8,4,5], 2, 3

Output: Intersected at '8'

Expected: Intersected at '8'