

197. We have  $n$  jobs, where every job is scheduled to be done from  $startTime[i]$  to  $endTime[i]$ , obtaining a profit of  $profit[i]$ . You're given the  $startTime$ ,  $endTime$  and  $profit$  arrays, return the maximum profit you can take such that there are no two jobs in the subset with overlapping time range. If you choose a job that ends at time  $X$  you will be able to start another job that starts at time  $X$ .

Program.

```
def jobScheduling(startTime, endTime, profit):

    jobs = sorted(zip(startTime, endTime, profit), key=lambda x: x[1])

    n = len(jobs)

    dp = [0] * n

    dp[0] = jobs[0][2] # Initialize dp for the first job

    for i in range(1, n):

        current_profit = jobs[i][2]

        prev_non_overlap = -1

        # Find the most recent non-overlapping job
        for j in range(i - 1, -1, -1):

            if jobs[j][1] <= jobs[i][0]:

                prev_non_overlap = j

                break

        if prev_non_overlap != -1:

            current_profit += dp[prev_non_overlap]

        dp[i] = max(dp[i - 1], current_profit)

    return dp[-1]

# Example 1

startTime1 = [1, 2, 3, 3]

endTime1 = [3, 4, 5, 6]

profit1 = [50, 10, 40, 70]

print(jobScheduling(startTime1, endTime1, profit1)) # Output: 120

# Example 2
```

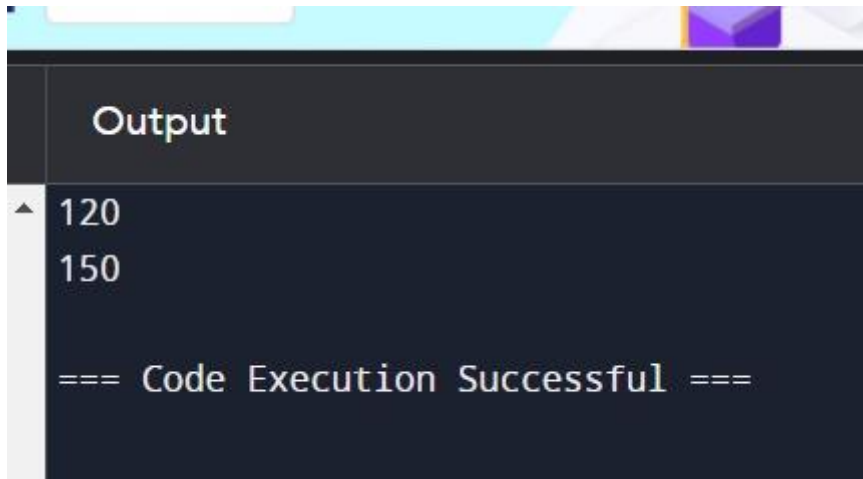
```
startTime2 = [1, 2, 3, 4, 6]
```

```
endTime2 = [3, 5, 10, 6, 9]
```

```
profit2 = [20, 20, 100, 70, 60]
```

```
print(jobScheduling(startTime2, endTime2, profit2))
```

Output:

A screenshot of a code execution environment's output window. The window has a dark background with a light-colored title bar at the top. The title bar contains the word "Output" in a light blue font. Below the title bar, the output is displayed in a light blue monospaced font. The output consists of three lines: "120", "150", and "=== Code Execution Successful ===". A vertical scrollbar is visible on the left side of the output area.

```
Output
120
150

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

Time complexity:  $O(n \log n)$