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.

Example 1:
Input: startTime = [1,2,3,3], endTime = [3,4,5,6], profit = [50,10,40,70]
Output: 120
Explanation: The subset chosen is the first and fourth job.
Time range [1-3]+[3-6] , we get profit of 120 = 50 + 70.
Example 2:
Input: startTime = [1,2,3,4,6], endTime = [3,5,10,6,9], profit = [20,20,100,70,60]
Output: 150
Explanation: The subset chosen is the first, fourth and fifth job. Profit obtained 150 = 20 + 70 + 60.

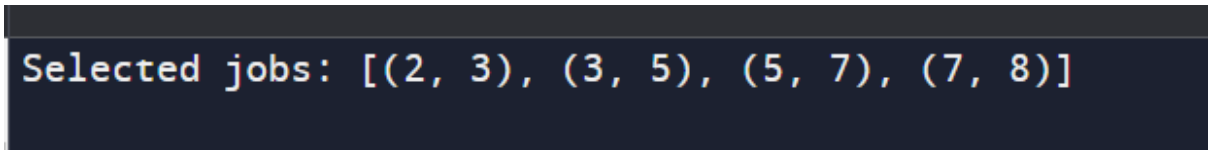
AIM: To write a python program for the you can take such that there are no two jobs in the subset with overlapping time range. If you choose a job that ends at time X you will be able to start another job that starts at time X.

PROGRAM :

```
def interval_scheduling(jobs):


    # Sort jobs based on their end time

    jobs.sort(key=lambda x: x[1])


    # Initialize variables

    selected_jobs = []

    current_end_time = 0


    # Iterate through the sorted jobs

    for job in jobs:

        start, end = job
```

```
        if start >= current_end_time:
            # If the job does not overlap with the previous job, select it
            selected_jobs.append(job)
            current_end_time = end

    return selected_jobs

# Example usage
jobs = [(1, 4), (2, 3), (3, 5), (7, 8), (5, 7), (6, 9)]
selected_jobs = interval_scheduling(jobs)
print("Selected jobs:", selected_jobs)
```

OUTPUT:



TIME COMPLEXITY : O(n)