

Day – 8 Greedy Algorithm

Problem Statement: There is **one** meeting room in a firm. You are given two arrays, start and end each of size N. For an index 'i', start[i] denotes the starting time of the ith meeting while end[i] will denote the ending time of the ith meeting. Find the maximum number of meetings that can be accommodated if only one meeting can happen in the room at a particular time. Print the order in which these meetings will be performed.

from typing import List

class meeting:

def __init__(self, start, end, pos):

self.start = start

self.end = end

self.pos = pos

def maxMeetings(s, e, n) :

meet = [meeting(s[i], e[i], i + 1) for i in range(n)]

sorted(meet, key=lambda x: (x.end, x.pos))

answer = []

limit = meet[0].end

answer.append(meet[0].pos)

for i in range(1, n):

if meet[i].start > limit:

limit = meet[i].end

```
    answer.append(meet[i].pos)
```

```
print("The order in which the meetings will be performed is ")
```

```
for i in answer:
```

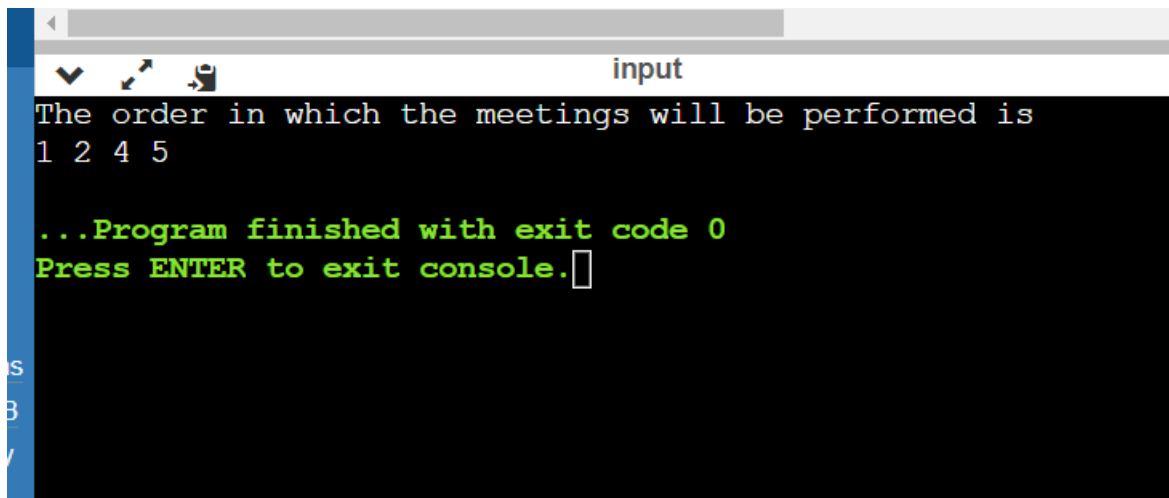
```
    print(i, end=" ")
```

```
n = 6
```

```
start = [1, 3, 0, 5, 8, 5]
```

```
end = [2, 4, 5, 7, 9, 9]
```

```
maxMeetings(start, end, n)
```



```
input
The order in which the meetings will be performed is
1 2 4 5
...Program finished with exit code 0
Press ENTER to exit console.
```

Problem Statement: We are given two arrays that represent the arrival and departure times of trains that stop at the platform. We need to find the minimum number of platforms needed at the railway station so that no train has to wait.

```
def minimum_platforms(arr, dep):
```

```
    arr.sort()
```

```
dep.sort()
```

```
platforms = 1
```

```
max_platforms = 1
```

```
arr_idx, dep_idx = 1, 0
```

```
n = len(arr)
```

```
while arr_idx < n and dep_idx < n:
```

```
    if arr[arr_idx] <= dep[dep_idx]:
```

```
        platforms += 1
```

```
        arr_idx += 1
```

```
    else:
```

```
        platforms -= 1
```

```
        dep_idx += 1
```

```
max_platforms = max(max_platforms, platforms)
```

```
return max_platforms
```

```
arrival_times = ["9:00", "9:45", "9:55", "11:00", "15:00", "18:00"]
```

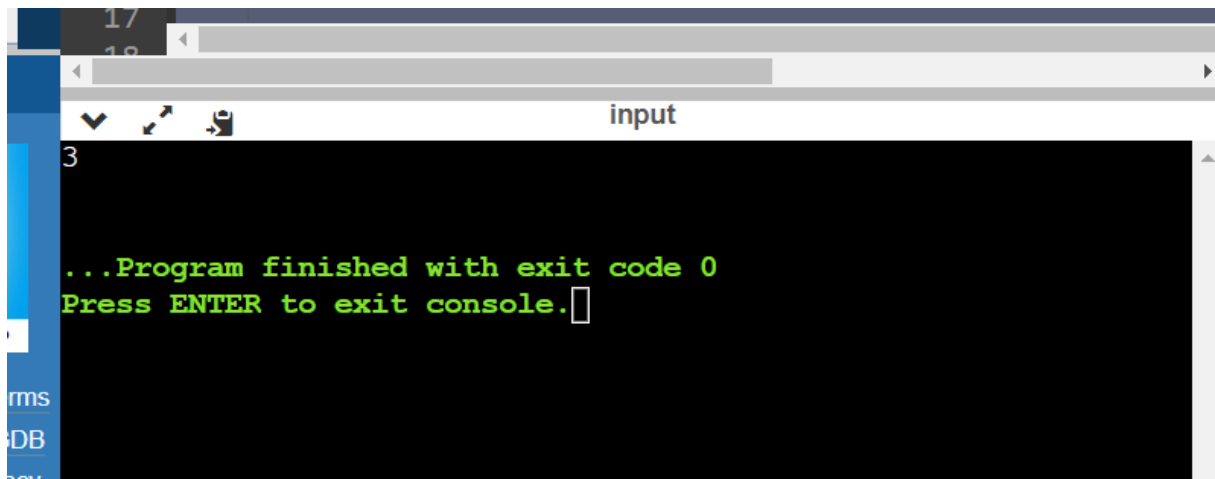
```
departure_times = ["9:20", "12:00", "11:30", "11:50", "19:00", "20:00"]
```

```
arrival_minutes = [int(time.split(':')[0]) * 60 + int(time.split(':')[1]) for time in arrival_times]
```

```
departure_minutes = [int(time.split(':')[0]) * 60 + int(time.split(':')[1]) for time in departure_times]
```

```
result = minimum_platforms(arrival_minutes, departure_minutes)
```

```
print(result)
```

A screenshot of a terminal window with a dark background and green text. The text reads: "...Program finished with exit code 0" followed by "Press ENTER to exit console." with a cursor. The terminal has a title bar with the word "input" and standard window controls. On the left, a portion of a file explorer sidebar is visible, showing folders like "rms" and "DB".

```
...Program finished with exit code 0
Press ENTER to exit console.
```

Problem Statement: You are given a set of N jobs where each job comes with a **deadline** and **profit**. The profit can only be earned upon completing the job within its deadline. Find the **number of jobs** done and the **maximum profit** that can be obtained. Each job takes a **single unit** of time and only **one job** can be performed at a time.

```
def findMaxProfit(N, jobs):
    jobs.sort(key=lambda x: x[2], reverse=True)
    maxDeadline = max(jobs, key=lambda x: x[1])[1]
    slot = [-1] * maxDeadline
    count = 0
    maxProfit = 0

    for job in jobs:
        for i in range(job[1] - 1, -1, -1):
            if slot[i] == -1:
                slot[i] = job[0]
                count += 1
                maxProfit += job[2]
                break
```

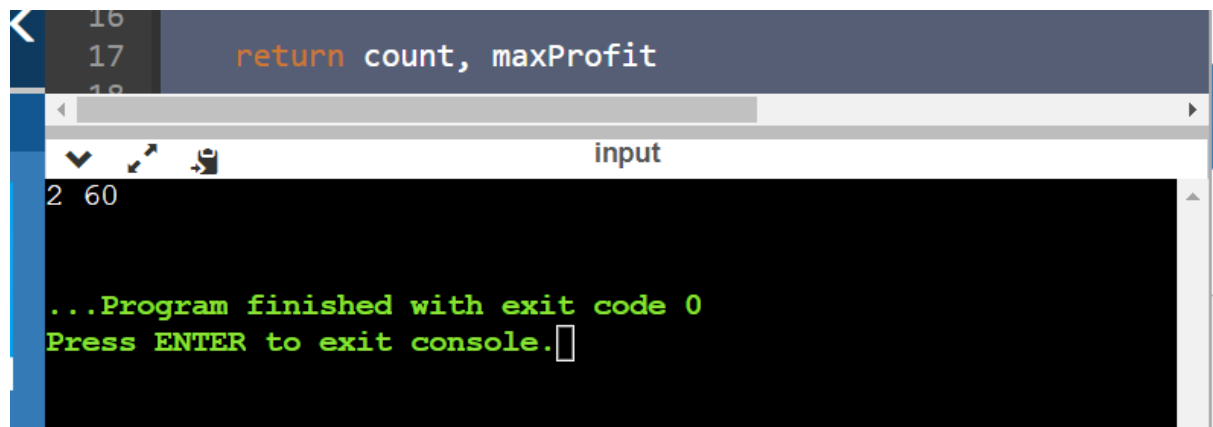
```
return count, maxProfit
```

```
jobs = [(1, 4, 20), (2, 1, 10), (3, 1, 40), (4, 1, 30)]
```

```
N = 4
```

```
count, maxProfit = findMaxProfit(N, jobs)
```

```
print(count, maxProfit)
```

A screenshot of a code editor and terminal window. The code editor at the top shows a Python function with the line `return count, maxProfit` highlighted. Below the editor is a terminal window titled 'input' which displays the output '2 60' and a green message: '...Program finished with exit code 0' followed by 'Press ENTER to exit console.' with a cursor.

Problem Statement: The weight of **N** items and their corresponding values are given. We have to put these items in a knapsack of weight **W** such that the **total value** obtained is **maximized**.

```
class Item:
```

```
    def __init__(self, value, weight):
```

```
        self.value = value
```

```
        self.weight = weight
```

```
def fractionalKnapsack( W, arr, n):
```

```
    arr.sort(key=lambda x: x.value / x.weight, reverse=True)
```

```
    curWeight = 0
```

```
finalvalue = 0.0
for i in range(n):
    if curWeight + arr[i].weight <= W:
        curWeight += arr[i].weight
        finalvalue += arr[i].value
    else:
        remain = W - curWeight
        finalvalue += arr[i].value / arr[i].weight * remain
        break
return finalvalue
```

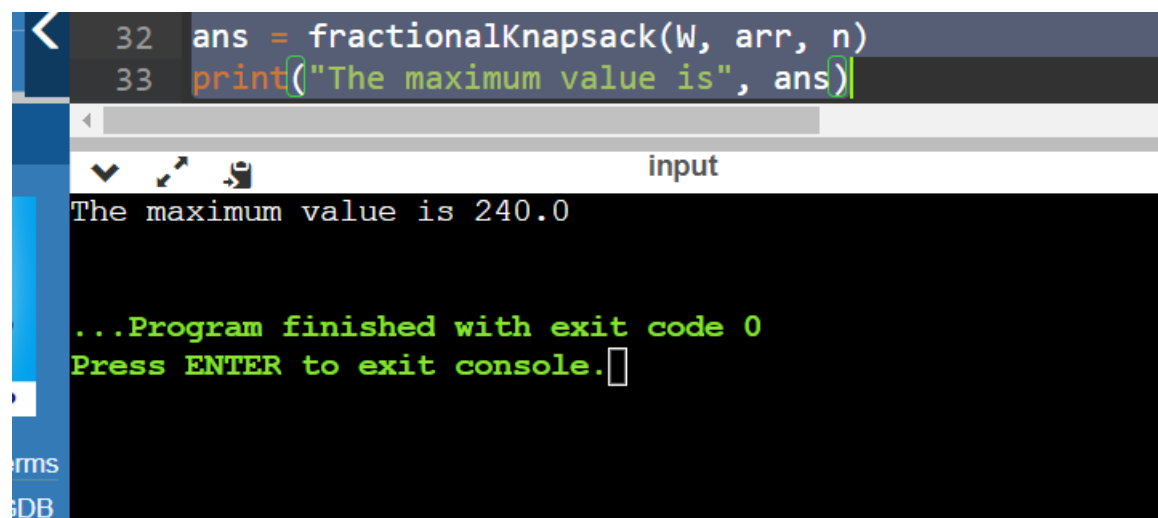
n = 3

W = 50

arr = [Item(60, 10), Item(100, 20), Item(120, 30)]

ans = fractionalKnapsack(W, arr, n)

print("The maximum value is", ans)

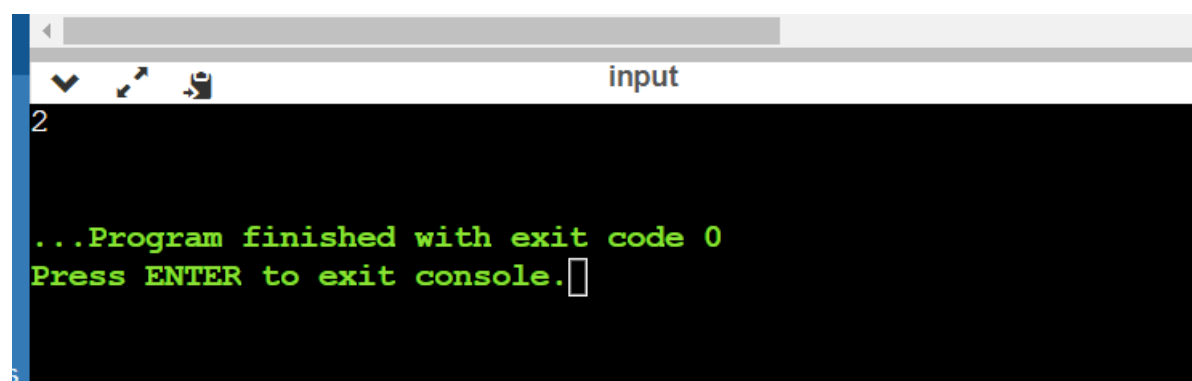


The screenshot shows a code editor with two lines of Python code: `ans = fractionalKnapsack(W, arr, n)` on line 32 and `print("The maximum value is", ans)` on line 33. Below the editor is a terminal window titled "input" which displays the output "The maximum value is 240.0". At the bottom of the terminal, a green message states "...Program finished with exit code 0" and "Press ENTER to exit console." with a cursor. On the left side of the terminal, there is a vertical sidebar with the text "rms" and "DB".

Problem Statement: Given a value V, if we want to make a change for V Rs, and we have an infinite supply of each of the denominations in Indian currency, i.e., we have an infinite supply of { 1, 2, 5, 10, 20, 50, 100, 500, 1000} valued coins/notes, what is the minimum number of coins and/or notes needed to make the change.

```
def find_minimum_coins(V):  
    total = V  
    count = 0  
    denominations = [1000, 500, 100, 50, 20, 10, 5, 2, 1]  
  
    for d in denominations:  
        if total >= d:  
            count += total // d  
            total -= (total // d) * d  
  
    return count
```

```
V = 70  
minimum_coins = find_minimum_coins(V)  
print(minimum_coins)
```

A screenshot of a terminal window with a dark background. The title bar at the top says 'input'. The terminal shows the number '2' on the first line, followed by the green text '...Program finished with exit code 0' and 'Press ENTER to exit console.' on the next line. A cursor is visible at the end of the second line. The terminal window has standard OS window controls (minimize, maximize, close) on the left side of the title bar.

```
2  
...Program finished with exit code 0  
Press ENTER to exit console.
```

Problem Statement: There is **one** meeting room in a firm. You are given two arrays, start and end each of size N. For an index 'i', start[i] denotes the starting time of the ith meeting while end[i] will denote the ending time of the ith

meeting. Find the maximum number of meetings that can be accommodated if only one meeting can happen in the room at a particular time. Print the order in which these meetings will be performed.

```
def max_meetings(N, start, end):  
    meetings = [(start[i], end[i], i+1) for i in range(N)]  
    meetings.sort(key=lambda x: (x[1], x[0]))
```

```
    selected_meetings = []  
    previous_end = 0  
    for meeting in meetings:  
        start_time, end_time, index = meeting  
        if start_time > previous_end:  
            selected_meetings.append(index)  
            previous_end = end_time
```

```
    return selected_meetings
```

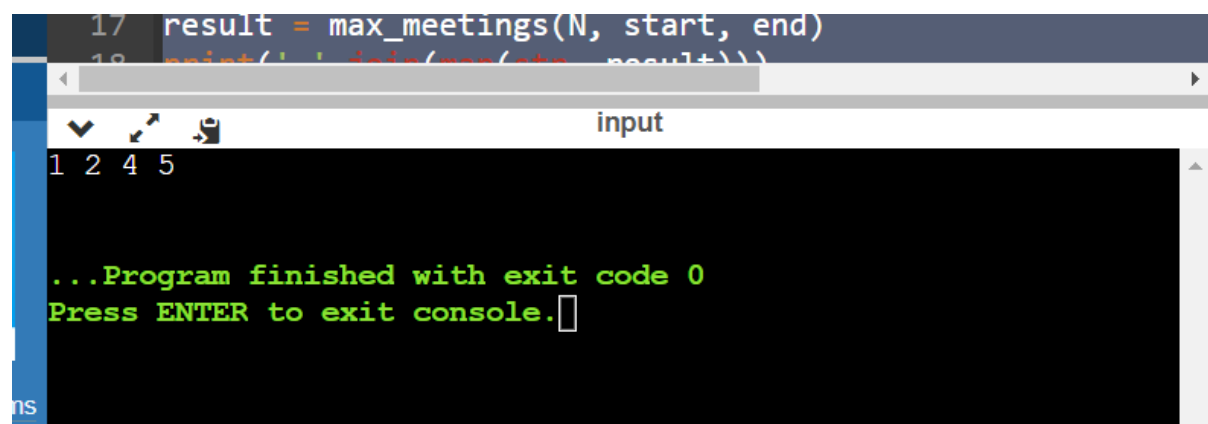
```
N = 6
```

```
start = [1, 3, 0, 5, 8, 5]
```

```
end = [2, 4, 5, 7, 9, 9]
```

```
result = max_meetings(N, start, end)
```

```
print(' '.join(map(str, result)))
```



The screenshot shows a code editor with the following Python code:

```
17 result = max_meetings(N, start, end)  
18 print(' '.join(map(str, result)))
```

Below the code editor is a terminal window titled "input". It displays the output of the program:

```
1 2 4 5  
  
...Program finished with exit code 0  
Press ENTER to exit console.
```