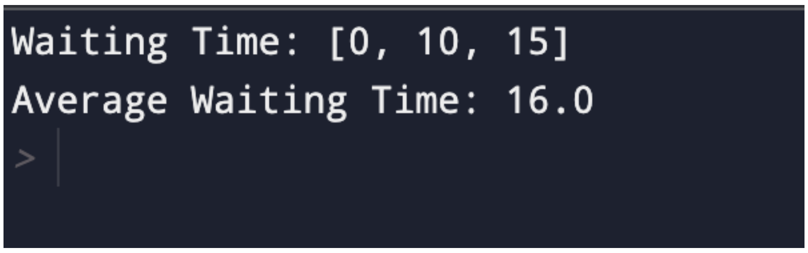
ASSIGNMENT-4
Prabhnoor Singh
102115059
3NC3

Q1)

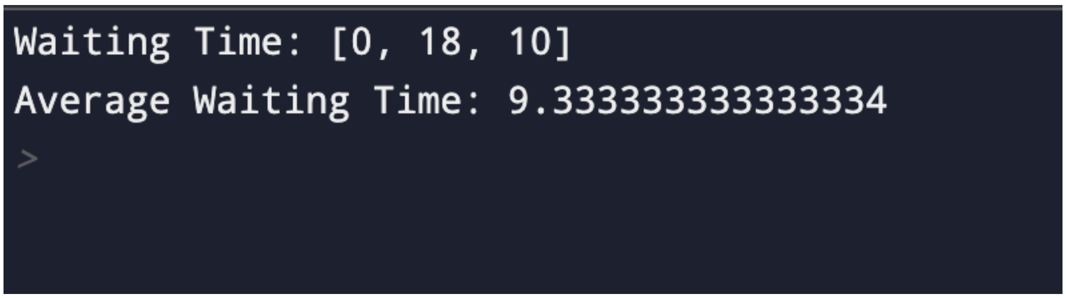
```
def hrrn_scheduling(processes, burst_times):
    n = len(processes)
    waiting_time = [0] * n
    response_ratio = [0] * n

    total_waiting_time = 0
    total_turnaround_time = 0

    for i in range(n):
        waiting_time[i] = total_waiting_time
        response_ratio[i] = (waiting_time[i] + burst_times[i]) / burst_times[i]
        total_waiting_time += burst_times[i]
        total_turnaround_time += total_waiting_time

    average_waiting_time = total_turnaround_time / n

    return waiting_time, average_waiting_time


# Example usage:
processes = ['P1', 'P2', 'P3']
burst_times = [10, 5, 8]
waiting_time, average_waiting_time = hrrn_scheduling(processes, burst_times)
print("Waiting Time:", waiting_time)
print("Average Waiting Time:", average_waiting_time)
```



```
Waiting Time: [0, 10, 15]
Average Waiting Time: 16.0
>
```

Q2)

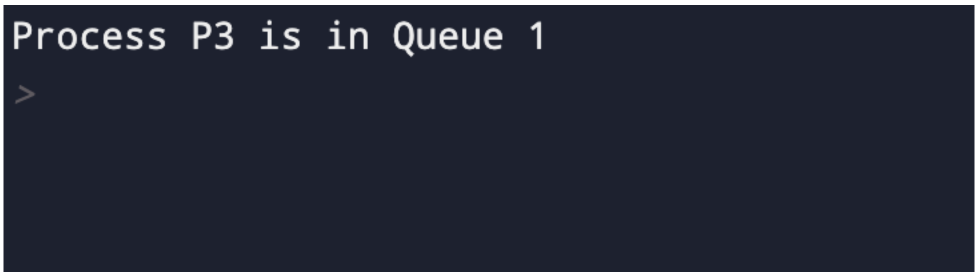
```
def ljf_scheduling(processes, burst_times):
    n = len(processes)
    waiting_time = [0] * n

    sorted_indices = sorted(range(n), key=lambda x: burst_times[x],
reverse=True)

    total_waiting_time = 0
    for i in range(n):
        waiting_time[sorted_indices[i]] = total_waiting_time
        total_waiting_time += burst_times[sorted_indices[i]]

    average_waiting_time = sum(waiting_time) / n

    return waiting_time, average_waiting_time


# Example usage:
processes = ['P1', 'P2', 'P3']
burst_times = [10, 5, 8]
waiting_time, average_waiting_time = ljf_scheduling(processes, burst_times)
print("Waiting Time:", waiting_time)
print("Average Waiting Time:", average_waiting_time)
```



```
Waiting Time: [0, 18, 10]
Average Waiting Time: 9.333333333333334
>
```

Q3)

```
class MultilevelQueue:
    def __init__(self, queues):
        self.queues = queues

    def schedule(self, process):
        for queue in self.queues:
            if process in queue:
                return queue.index(process)

# Example usage:
queues = [['P1', 'P2'], ['P3', 'P4', 'P5'], ['P6']]
multilevel_queue = MultilevelQueue(queues)
process_to_schedule = 'P3'
queue_index = multilevel_queue.schedule(process_to_schedule)
print(f"Process {process_to_schedule} is in Queue {queue_index +
1}")
```

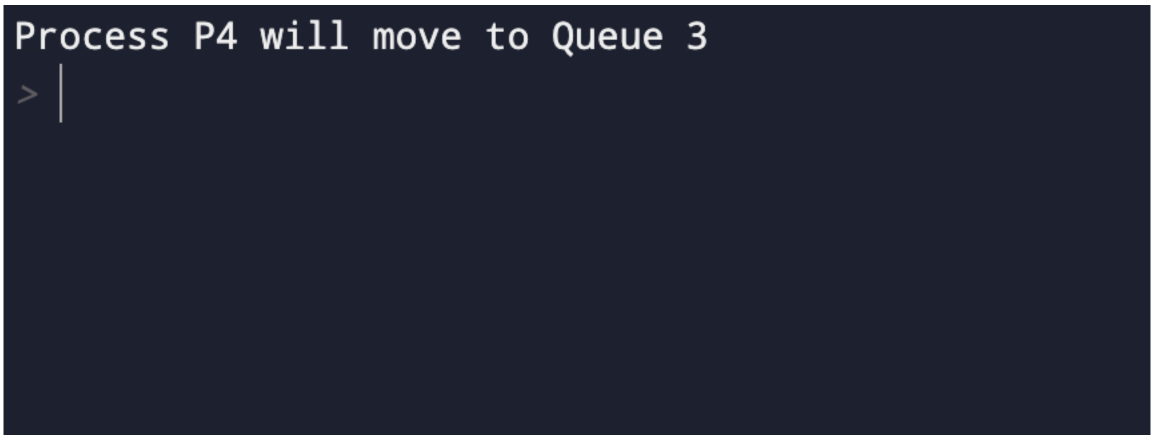


```
Process P3 is in Queue 1
>
```

Q4)

```
class MultilevelFeedbackQueue:
    def __init__(self, queues):
        self.queues = queues

    def schedule(self, process, current_queue):
        if current_queue < len(self.queues) - 1:
            return current_queue + 1
        else:
```

```
        return current_queue

# Example usage:
queues = [['P1', 'P2'], ['P3', 'P4', 'P5'], ['P6']]
multilevel_feedback_queue = MultilevelFeedbackQueue(queues)
current_queue_index = 1
process_to_schedule = 'P4'
next_queue_index =
multilevel_feedback_queue.schedule(process_to_schedule,
current_queue_index)
print(f"Process {process_to_schedule} will move to Queue {next_queue_index
+ 1}")
```



```
Process P4 will move to Queue 3
> |
```