Name : Khune Radha Rahul

Class : TE Comp-2

Roll no : 07(F22112010)

**Input:**

def print\_gantt\_chart(gantt\_chart):

print("Gantt Chart:")

for time, pid in gantt\_chart:

print(f"| {pid} ", end="")

print("|")

for time, \_ in gantt\_chart:

print(f"{time:>3}", end="")

print()

def fcfs(processes):

time = 0

gantt\_chart = []

processes.sort(key=lambda p: p['arrival\_time'])

for process in processes:

if time < process['arrival\_time']:

time = process['arrival\_time']

gantt\_chart.append((time, process['pid']))

time += process['burst\_time']

gantt\_chart.append((time, "END"))

print\_gantt\_chart(gantt\_chart)

def sjf\_preemptive(processes):

time = 0

gantt\_chart = []

while processes:

available\_processes = [p for p in processes if p['arrival\_time'] <= time]

if not available\_processes:

time += 1

continue

current\_process = min(available\_processes, key=lambda p: p['remaining\_time'])

gantt\_chart.append((time, current\_process['pid']))

current\_process['remaining\_time'] -= 1

if current\_process['remaining\_time'] == 0:

processes.remove(current\_process)

time += 1

gantt\_chart.append((time, "END"))

print\_gantt\_chart(gantt\_chart)

def priority\_non\_preemptive(processes):

time = 0

gantt\_chart = []

while processes:

available\_processes = [p for p in processes if p['arrival\_time'] <= time]

if not available\_processes:

time += 1

continue

current\_process = min(available\_processes, key=lambda p: p['priority'])

gantt\_chart.append((time, current\_process['pid']))

time += current\_process['burst\_time']

processes.remove(current\_process)

gantt\_chart.append((time, "END"))

print\_gantt\_chart(gantt\_chart)

def round\_robin(processes, quantum):

time = 0

gantt\_chart = []

queue = processes[:]

while queue:

current\_process = queue.pop(0)

gantt\_chart.append((time, current\_process['pid']))

if current\_process['remaining\_time'] > quantum:

time += quantum

current\_process['remaining\_time'] -= quantum

queue.append(current\_process)

else:

time += current\_process['remaining\_time']

current\_process['remaining\_time'] = 0

gantt\_chart.append((time, "END"))

print\_gantt\_chart(gantt\_chart)

# Sample Processes

processes = [

{'pid': 'P1', 'burst\_time': 5, 'arrival\_time': 0, 'remaining\_time': 5, 'priority': 2},

{'pid': 'P2', 'burst\_time': 3, 'arrival\_time': 2, 'remaining\_time': 3, 'priority': 1},

{'pid': 'P3', 'burst\_time': 8, 'arrival\_time': 3, 'remaining\_time': 8, 'priority': 3},

{'pid': 'P4', 'burst\_time': 6, 'arrival\_time': 5, 'remaining\_time': 6, 'priority': 2},

]

# Simulate Algorithms

print("FCFS:")

fcfs(processes[:])

print("\nSJF (Preemptive):")

sjf\_preemptive(processes[:])

print("\nPriority (Non-Preemptive):")

priority\_non\_preemptive(processes[:])

print("\nRound Robin (Quantum = 2):")

round\_robin(processes[:], quantum=2)

**Output:**

FCFS:

Gantt Chart:

| P1 | P2 | P3 | P4 | END |

0 5 8 16 22

SJF (Preemptive):

Gantt Chart:

| P1 | P1 | P1 | P1 | P1 | P2 | P2 | P2 | P4 | P4 | P4 | P4 | P4 | P4 | P3 | P3 | P3 | P3 | P3 | P3 | P3 | P3 | END |

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22

Priority (Non-Preemptive):

Gantt Chart:

| P1 | P2 | P4 | P3 | END |

0 5 8 14 22

Round Robin (Quantum = 2):

Gantt Chart:

| P1 | P2 | P3 | P4 | END |

0 0 0 0 0