Operating System

Semester 4

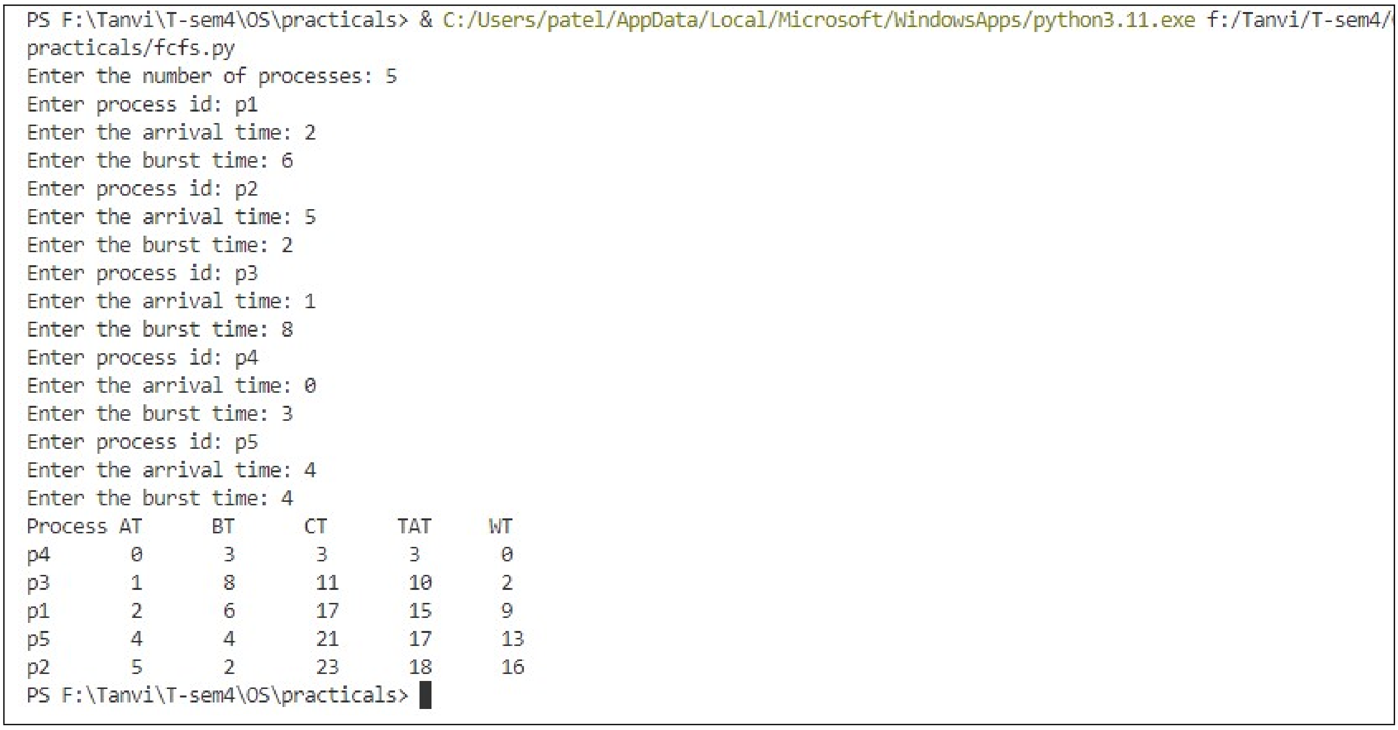
Practical-4: Write a python program to perform First Come First Serve (fcfs) Algorithm

Input:

```
#Write a python program to perform First Come First Serve
#Name: Patel Tanvi Bharatkumar

#function to arrange the process in the ascending order according to the Arrival Time
def fcfs_sort(l1,l2):
    n = len(l2)
    for i in range(n-1):
        for j in range(n-i-1):
            if(l2[j] > l2[j+1]):
                l2[j],l2[j+1] = l2[j+1],l2[j]
                l1[j],l1[j+1] = l1[j+1],l1[j]
    return l1,l2

#Function to print the output in the form of a table
def printProcessTableInSequence(d1):
    print("Process AT\tBT\tCT\tTAT\tWT")
    for i in d1:
        print(i,"\t",d1[i][0],"\t",d1[i][1],"\t",d1[i][2],"\t",d1[i][3],"\t",d1[i][4])

#Taking input from the user(number of processes)
np = int(input("Enter the number of processes: "))

#making empty lists and dictionaries
processes = dict()
listOfProcesses = []
listOfArrivalTime = []
processInProcessor = dict()

#for loop to take the input of the arrival time and burst time from the user
for i in range(np):
    p_id = input("Enter process id: ")
    arrivalTime = int(input("Enter the arrival time: "))
    burstTime = int(input("Enter the burst time: "))
    processes[p_id] = [arrivalTime,burstTime,0,0,0]
    listOfProcesses.append(p_id)
    listOfArrivalTime.append(arrivalTime)

nlistOfProcesses,nlistOfArrivalTime = fcfs_sort(listOfProcesses,listOfArrivalTime)
```

```
timer = 0

for i in nlistOfProcesses:
    if(timer < processes[i][0]):
        k = processes[i][0] - timer
        timer += k
        k = 0
    timer += processes[i][1]
    processes[i][2] = timer
    processes[i][3] = processes[i][2] - processes[i][0]
    processes[i][4] = processes[i][3] - processes[i][1]
    processInProcessor[i] = processes[i]

printProcessTableInSequence(processInProcessor)
```

Output:



```
PS F:\Tanvi\T-sem4\OS\practicals> & C:/Users/patel/AppData/Local/Microsoft/WindowsApps/python3.11.exe f:/Tanvi/T-sem4/
practicals/fcfs.py
Enter the number of processes: 5
Enter process id: p1
Enter the arrival time: 2
Enter the burst time: 6
Enter process id: p2
Enter the arrival time: 5
Enter the burst time: 2
Enter process id: p3
Enter the arrival time: 1
Enter the burst time: 8
Enter process id: p4
Enter the arrival time: 0
Enter the burst time: 3
Enter process id: p5
Enter the arrival time: 4
Enter the burst time: 4
Process AT      BT      CT      TAT     WT
p4      0       3       3       3       0
p3      1       8       11      10      2
p1      2       6       17      15      9
p5      4       4       21      17      13
p2      5       2       23      18      16
PS F:\Tanvi\T-sem4\OS\practicals>
```

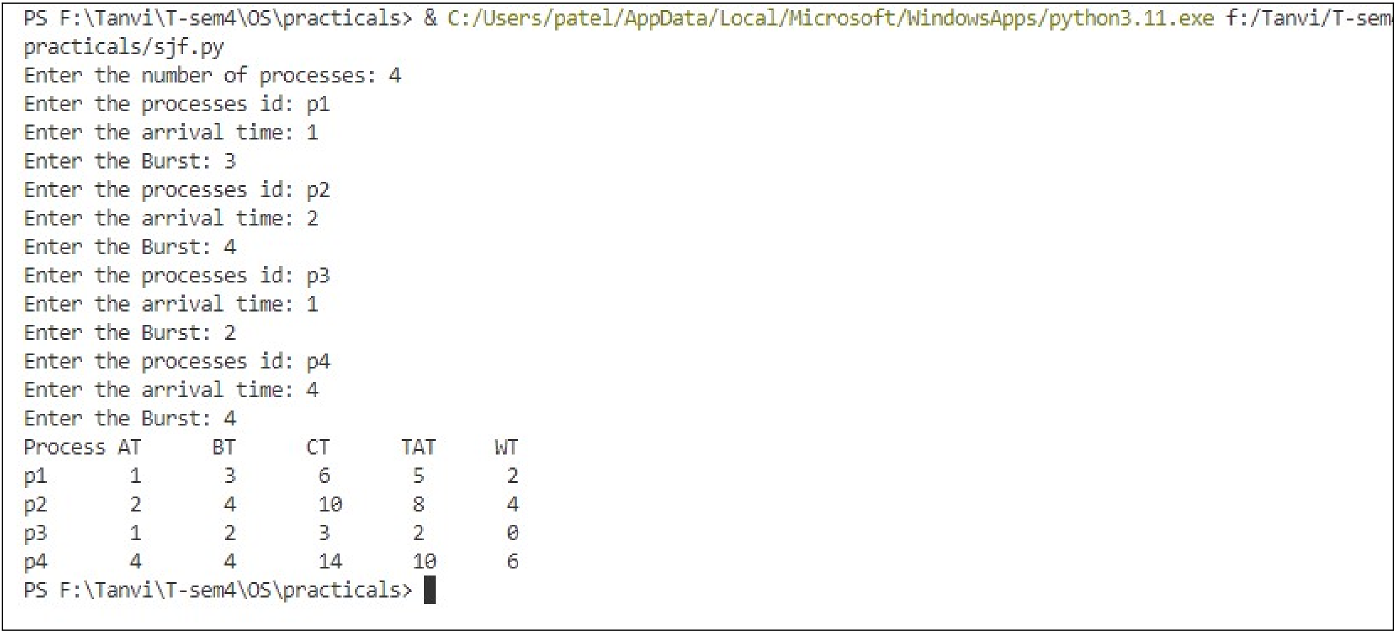
Practical-5: Write a python program to perform Shortest Job First (sjf) Algorithm

Input:

```
#Write a python program to perform Shortest Job First
#Name: Patel Tanvi Bharatkumar
#sjf

def printProccessesInTable(d1):
    print("Process AT\tBT\tCT\tTAT\tWT")
    for i in d1:
        print(i,"\t",d1[i][0],"\t",d1[i][1],"\t",d1[i][2],"\t",d1[i][3],"\t",d1[i][4])

def checkProcess(d1,t1):
    l1 = []
    l2 = []
    for i in d1:
        if(d1[i][0] <= t1 and d1[i][1] != 0 ):
            l1.append(i)
            l2.append(d1[i][1])
    k = min(l2)
    pi = l2.index(k)
    p = l1[pi]
    return p

np = int(input("Enter the number of processes: "))

proccesses = dict()
copyProcessor = dict()
listOfProcess = []
listOfBurstTime = []
listOfArrivalTime = []

for i in range(np):
    p_id = input("Enter the processes id: ")
    arrivalTime = int(input("Enter the arrival time: "))
    burstTime = int(input("Enter the Burst: "))
    proccesses[p_id] = [arrivalTime,burstTime,0,0,0]
    copyProcessor[p_id] = [arrivalTime,burstTime,0,0,0]
    listOfProcess.append(p_id)
    listOfArrivalTime.append(arrivalTime)
    listOfBurstTime.append(burstTime)
```

```
timer = min(listOfArrivalTime)

for i in range(np):
   process = checkProcess(copyProcessor,timer)
   timer += copyProcessor[process][1]
   copyProcessor[process][1] = 0
   proccesses[process][2] = timer
   proccesses[process][3] = proccesses[process][2] - proccesses[process][0]
   proccesses[process][4] = proccesses[process][3] - proccesses[process][1]



printProccessesInTable(proccesses)
```

Output:



```
PS F:\Tanvi\T-sem4\OS\practicals> & C:/Users/patel/AppData/Local/Microsoft/WindowsApps/python3.11.exe f:/Tanvi/T-sem
practicals/sjf.py
Enter the number of processes: 4
Enter the processes id: p1
Enter the arrival time: 1
Enter the Burst: 3
Enter the processes id: p2
Enter the arrival time: 2
Enter the Burst: 4
Enter the processes id: p3
Enter the arrival time: 1
Enter the Burst: 2
Enter the processes id: p4
Enter the arrival time: 4
Enter the Burst: 4
Process AT      BT      CT      TAT     WT
p1       1       3       6       5       2
p2       2       4       10      8       4
p3       1       2       3       2       0
p4       4       4       14      10      6
PS F:\Tanvi\T-sem4\OS\practicals> 
```

Operating System

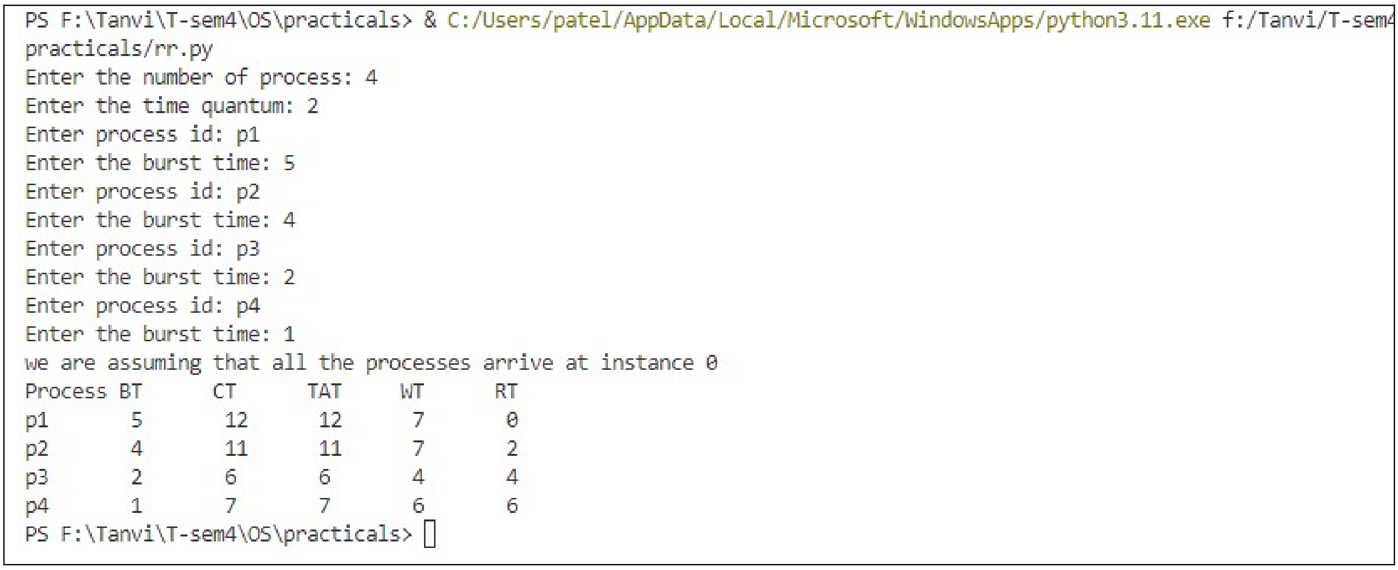
Semester 4

Practical-6: Write a python program to perform Round Robin (rr) Algorithm

Input:

```
#Write a python program to perform Round Robin
#Name: Patel Tanvi Bharatkumar
#rr
#we are assuming that all the processes arrive at instance 0

def printProccessesInTable(d1):
    print("Process BT\tCT\tTAT\tWT\tRT")
    for i in d1:
        print(i,"\t",d1[i][0],"\t",d1[i][1],"\t",d1[i][2],"\t",d1[i][3],"\t",d1[i][4])

n = int(input("Enter the number of process: "))

time_quantum = int(input("Enter the time quantum: "))
processes = dict()
listOfProcesses = []
processesInprocessor = dict()

for i in range(n):
    p_id = input("Enter process id: ")
    burstTime = int(input("Enter the burst time: "))
    processes[p_id] = [burstTime,0,0,0,0]
    processesInprocessor[p_id] = [burstTime,0,0,0,0]
    listOfProcesses.append(p_id)

timer = 0

while processes:
    for i in listOfProcesses:
        if processes[i][0] == processesInprocessor[i][0]:
            processesInprocessor[i][4] = timer
        if processes[i][0] > time_quantum:
            timer += time_quantum
            processes[i][0] -= time_quantum
            listOfProcesses.append(i)
        elif processes[i][0] <= time_quantum:
            timer += processes[i][0]
            processes[i][0] = 0
            processesInprocessor[i][1] = timer
            processesInprocessor[i][2] = processesInprocessor[i][1]
        processesInprocessor[i][3] = processesInprocessor[i][2] - processesInprocessor[i][0]
        del processes[i]

print("we are assuming that all the processes arrive at instance 0")
printProccessesInTable(processesInprocessor)
```

Output:



```
PS F:\Tanvi\T-sem4\OS\practicals> & C:/Users/patel/AppData/Local/Microsoft/WindowsApps/python3.11.exe f:/Tanvi/T-sem4
practicals/rr.py
Enter the number of process: 4
Enter the time quantum: 2
Enter process id: p1
Enter the burst time: 5
Enter process id: p2
Enter the burst time: 4
Enter process id: p3
Enter the burst time: 2
Enter process id: p4
Enter the burst time: 1
we are assuming that all the processes arrive at instance 0
Process BT      CT      TAT     WT      RT
p1      5       12      12      7       0
p2      4       11      11      7       2
p3      2       6       6       4       4
p4      1       7       7       6       6
PS F:\Tanvi\T-sem4\OS\practicals> 
```

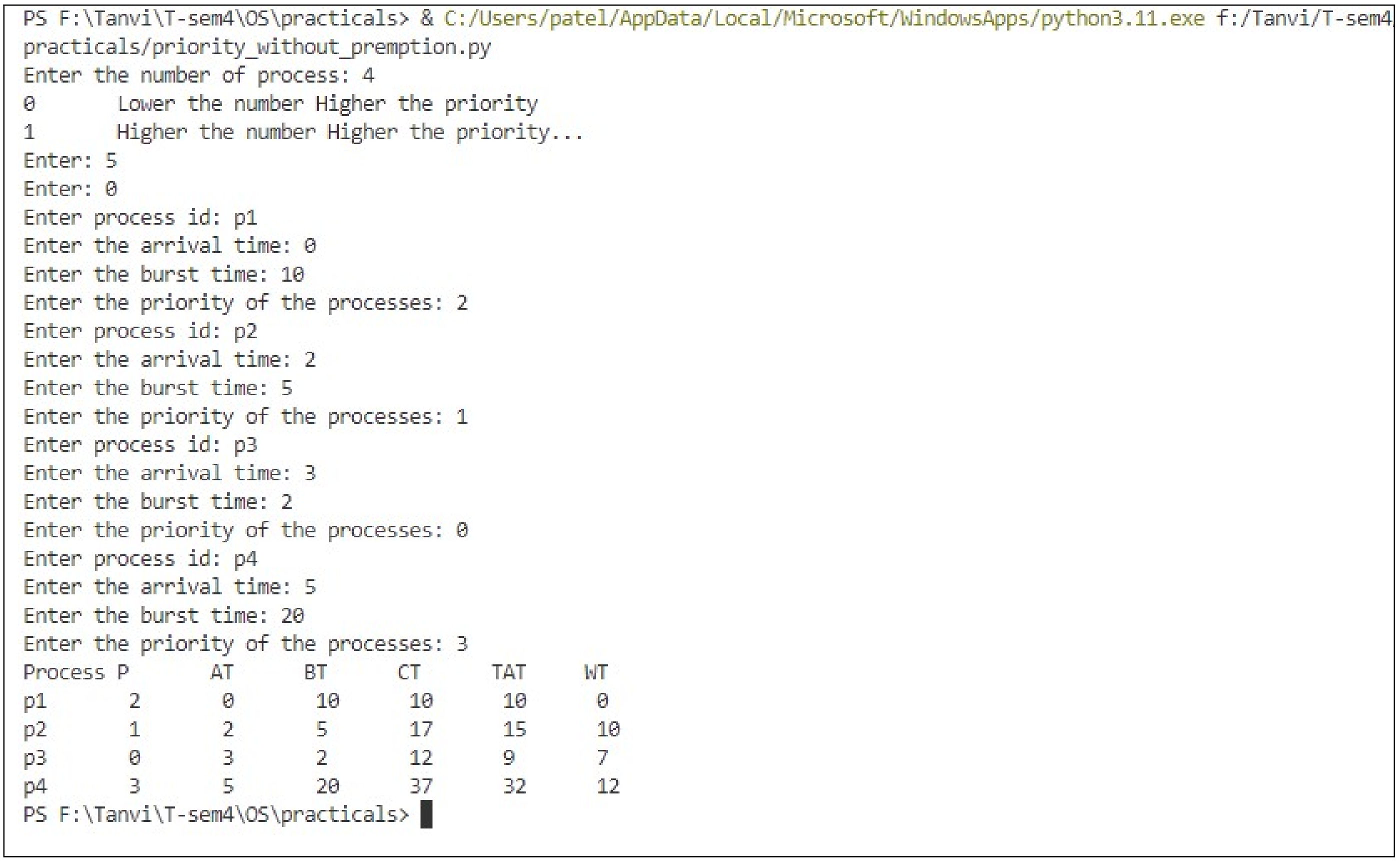
Practical-7: Write a python program to perform Priority without premption Algorithm

Input:

```
#Write a python program to perform Priority without premption
#Name: Patel Tanvi Bharatkumar
#priority without premption

def checkProcess(d1,t1,c):
    l1 = []
    l2 = []
    for i in d1:
        if(d1[i][1]<=t1 and d1[i][2] != 0):
            l1.append(i)
            l2.append(d1[i][0])
    if c==0:
        k = min(l2)
    elif c==1:
        k = max(l2)
    pi = l2.index(k)
    p = l1[pi]
    return p

def printProccessesInTable(d1):
    print("Process P\tAT\tBT\tCT\tTAT\tWT")
    for i in d1:
        print(i,"\t",d1[i][0],"\t",d1[i][1],"\t",d1[i][2],"\t",d1[i][3],"\t",d1[i][4],"\t",d1[i][5])

n = int(input("Enter the number of process: "))

print("0\tLower the number Higher the priority\n1\tHigher the number Higher the priority...")
while True:
    condition = int(input("Enter: "))
    if condition == 0 or condition == 1:
        break



processes = dict()
listOfProcesses = []
listOfArrivalTime = []
lpriority = []
sumOfBurstTime = 0
processesInprocessor = dict()
```

```
for i in range(n):
   p_id = input("Enter process id: ")
   arrivalTime = int(input("Enter the arrival time: "))
   burstTime = int(input("Enter the burst time: "))
   sumOfBurstTime += burstTime
   priority = int(input("Enter the priority of the processes: "))
   processes[p_id] = [priority,arrivalTime,burstTime,0,0,0]
   processesInprocessor[p_id] = [priority,arrivalTime,burstTime,0,0,0]
   listOfProcesses.append(p_id)
   listOfArrivalTime.append(arrivalTime)
   lpriority.append(priority)

timer = min(listOfArrivalTime)

for i in range(n):
   process = checkProcess(processes,timer,condition)
   timer += processes[process][2]
   processes[process][2] = 0
   processesInprocessor[process][3] = timer
   processesInprocessor[process][4] = processesInprocessor[process][3] -
processesInprocessor[process][1]
   processesInprocessor[process][5] = processesInprocessor[process][4] -
processesInprocessor[process][2]]

printProccessesInTable(processesInprocessor)
```

Output:



```
PS F:\Tanvi\T-sem4\OS\practicals> & C:/Users/patel/AppData/Local/Microsoft/WindowsApps/python3.11.exe f:/Tanvi/T-sem4
practicals/priority_without_premption.py
Enter the number of process: 4
0       Lower the number Higher the priority
1       Higher the number Higher the priority...
Enter: 5
Enter: 0
Enter process id: p1
Enter the arrival time: 0
Enter the burst time: 10
Enter the priority of the processes: 2
Enter process id: p2
Enter the arrival time: 2
Enter the burst time: 5
Enter the priority of the processes: 1
Enter process id: p3
Enter the arrival time: 3
Enter the burst time: 2
Enter the priority of the processes: 0
Enter process id: p4
Enter the arrival time: 5
Enter the burst time: 20
Enter the priority of the processes: 3
Process P       AT      BT      CT      TAT     WT
p1      2       0       10      10      10      0
p2      1       2       5       17      15      10
p3      0       3       2       12      9       7
p4      3       5       20      37      32      12
PS F:\Tanvi\T-sem4\OS\practicals> 
```