EXPERIMENT 1

VVIT
VASIREDDY VENKATADRI INSTITUTE OF TECHNOLOGY

VASIREDDY VENKATADRI INSTITUTE OF TECHNOLOGY
Permanently Affiliated to JNTU Kakinada, Approved by AICTE
Accredited by NAAC with 'A' Grade, ISO 9001:2008 Certified
Nambur, Pedakakani (M), Guntur (Dt) - 522508
DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
B.Tech Program is Accredited by NBA

CERTIFICATE

Name of the Lab : OPERATING SYSTEMS

Name of the Student : Thota Nagababu

Student Regd. No. : 18BQ1A05K3

CLASS : III B.TECH. I SEM CSE – D

GIT HUB LINK: https://github.com/nagababuthota984/5K3-OS-LAB

INDEX

EXPERIMENT NO: 1 (a)

AIM : Simulate FCFS CPU scheduling algorithm

DESCRIPTION : **First Come First Serve (FCFS)** is an operating system scheduling algorithm that automatically executes queued requests and processes in order of their arrival. It is the easiest and simplest CPU scheduling algorithm. In this type of algorithm, processes which requests the CPU first get the CPU allocation first. This is managed with a FIFO queue. The full form of FCFS is First Come First Serve.

As the process enters the ready queue, its PCB (Process Control Block) is linked with the tail of the queue and, when the CPU becomes free, it should be assigned to the process at the beginning of the queue.

PROGRAMMING LANGUAGE USED: * PYTHON

LIBRARIES USED: package texttable - from texttable import Texttable

PROGRAM:

```
from texttable import Texttable


def main():
    n = int(input("enter the number of process"))
    l=[]
    for i in range(n):
        name = input("enter the name of the process:")
        arrival = int(input("enter the arival time of process in ms:"))
        burst_time =int(input("enter the burst time in ms:"))
        #head = ['Process Name','Arrival Time','Burst Time','Wait Time','Turnaround Time']
        x = [name,arrival,burst_time,0,0]
        l.append(x)

    l = sorted(l,key=wrt_arrival_time)
    l[0][4]=l[0][2]
    for i in range(1,n):
        w=0
        for j in range(0,i):
            w+=l[j][2]
        l[i][3]=w-l[i][1]
        l[i][4]=l[i][2]+l[i][3]

    total_wt=0
    total_tt=0
    for i in l:
        total_wt +=int(i[3])
        total_tt += int(i[4])

    t = Texttable()
    head = ['Process Name','Arrival Time','Burst Time','Wait Time','Turnaround Time']
    l.insert(0,head)
    t.add_rows(l)
    print(t.draw())



    print("Total waiting time : ",total_wt)
    print("Average waiting time : ",total_wt/n)

    print("total turnaround time : ",total_tt)
    print("Average turnaround time : ",total_tt/n)





def wrt_arrival_time(x):
    return x[1]

if __name__ == "__main__":
    main()
```

Ask user number of processes:

Ask user each process burst time:

Ask user the arrival time of each process:

*Ask user for priority of process for priority scheduling

*Ask user the quantum time for round robin scheduling

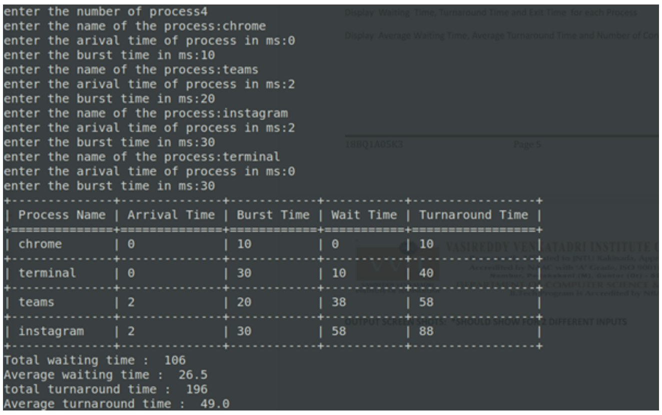
OUTPUT: * it should be same when your program gets executed

Display Waiting Time, Turnaround Time and Exit Time for each Process

Display Average Waiting Time, Average Turnaround Time and Number of Context Switches.

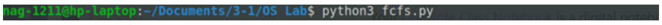
OUTPUT SCREEN SHOTS: *SHOULD SHOW FOR 2 DIFFERENT INPUTS

OUTPUT 1:

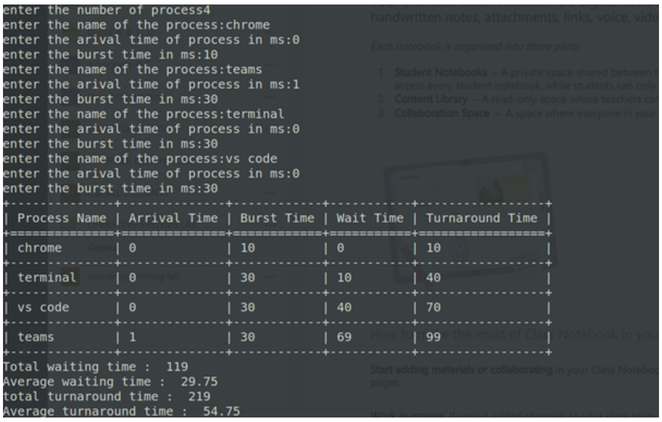


```
enter the number of process4
enter the name of the process:chrome
enter the arival time of process in ms:0
enter the burst time in ms:10
enter the name of the process:teams
enter the arival time of process in ms:2
enter the burst time in ms:20
enter the name of the process:instagram
enter the arival time of process in ms:2
enter the burst time in ms:30
enter the name of the process:terminal
enter the arival time of process in ms:0
enter the burst time in ms:30
+--------------+--------------+------------+-----------+-----------------+
| Process Name | Arrival Time | Burst Time | Wait Time | Turnaround Time |
+==============+==============+============+===========+=================+
| chrome       | 0            | 10         | 0         | 10              |
+--------------+--------------+------------+-----------+-----------------+
| terminal     | 0            | 30         | 10        | 40              |
+--------------+--------------+------------+-----------+-----------------+
| teams        | 2            | 20         | 38        | 58              |
+--------------+--------------+------------+-----------+-----------------+
| instagram    | 2            | 30         | 58        | 88              |
+--------------+--------------+------------+-----------+-----------------+
Total waiting time :  106
Average waiting time :  26.5
total turnaround time :  196
Average turnaround time :  49.0
```

OUTPUT2 :



```
nag-1211@hp-laptop:~/Documents/3-1/OS Lab$ python3 fcfs.py
```



```
enter the number of process4
enter the name of the process:chrome
enter the arival time of process in ms:0
enter the burst time in ms:10
enter the name of the process:teams
enter the arival time of process in ms:1
enter the burst time in ms:30
enter the name of the process:terminal
enter the arival time of process in ms:0
enter the burst time in ms:30
enter the name of the process:vs code
enter the arival time of process in ms:0
enter the burst time in ms:30
+--------------+--------------+------------+-----------+------------------+
| Process Name | Arrival Time | Burst Time | Wait Time | Turnaround Time |
+==============+==============+============+===========+==================+
| chrome       | 0            | 10         | 0         | 10               |
+--------------+--------------+------------+-----------+------------------+
| terminal     | 0            | 30         | 10        | 40               |
+--------------+--------------+------------+-----------+------------------+
| vs code      | 0            | 30         | 40        | 70               |
+--------------+--------------+------------+-----------+------------------+
| teams        | 1            | 30         | 69        | 99               |
+--------------+--------------+------------+-----------+------------------+
Total waiting time :  119
Average waiting time :  29.75
total turnaround time :  219
Average turnaround time :  54.75
```

-

EXPERIMENT NO: 1 (b)

AIM :   Simulate Non Preemptive SJF CPU scheduling algorithm

DESCRIPTION : Shortest job first (SJF) or shortest job next, is a scheduling policy that selects the waiting process with the smallest execution time to execute next. SJN is a non-preemptive algorithm.

Shortest Job first has the advantage of having a minimum average waiting time among all scheduling algorithms.

It is a Greedy Algorithm.

It may cause starvation if shorter processes keep coming. This problem can be solved using the concept of ageing.

It is practically infeasible as Operating System may not know burst time and therefore may not sort them. While it is not possible to predict execution time, several methods can be used to estimate the execution time for a job, such as a weighted average of previous execution times. SJF can be used in specialized environments where accurate estimates of running time are available.

PROGRAMMING LANGUAGE USED: *   PYTHON

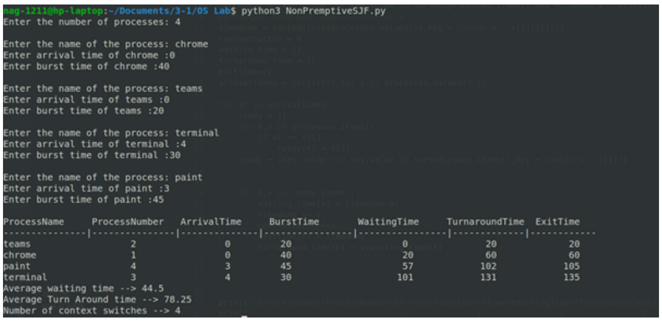
LIBRARIES USED: No additional libraries have been used.

PROGRAM :

```
no_of_processes = int(input("Enter the number of processes: "))
processes = {}
for i in range(1,no_of_processes+1):
    process_name = input("\nEnter the name of the process: ")
    arrival_time = int(input("Enter arrival time of "+process_name+" :"))
    burst_time = int(input("Enter burst time of "+process_name+" :"))
    processes[i] = [process_name,arrival_time,burst_time]

processes = {key:value for key,value in sorted(processes.items(),key = lambda x : x[1][1])}
#processes have been read successfully
timedone = sorted(list(processes.values()),key = lambda x : x[1])[0][1]
contextswitch = 0
waiting_time = {}
turnaround_time = {}
exittime={}
arrivaltimes = set([i[1] for i in processes.values() ])

for at in arrivaltimes:
    ready = {}
    for k,v in processes.items():
        if at == v[1]:
            ready[k] = v[2]
    ready = {key:value for key,value in sorted(ready.items(),key = lambda x : x[1])}


    for k,v in ready.items():
        waiting_time[k] = timedone-at
        timedone += v
        contextswitch+=1
        exittime[k] = timedone
        turnaround_time[k] = v+waiting_time[k]
```

```
print("\nProcessName\tProcessNumber\tArrivalTime\tBurstTime\tWaitingTime\tTurnaroundTime
\tExitTime")
print("---------------|---------------|--------------|----------------|----------------|-------------|------------")
for k,v in waiting_time.items():
    print(processes[k][0].ljust(15) + str(k).center(17)+ str(processes[k][1]).center(18) +
str(processes[k][2]).ljust(15) + str(v).center(15) +
str(turnaround_time[k]).center(15)+str(exittime[k]).center(16))
print("Average waiting time -->",round((sum(waiting_time.values()))/no_of_processes, 3))
print("Average Turn Around time -->",round((sum(turnaround_time.values()))/no_of_processes,3))
print("Number of context switches -->",contextswitch)
```

OUTPUT SCREENSHOTS:

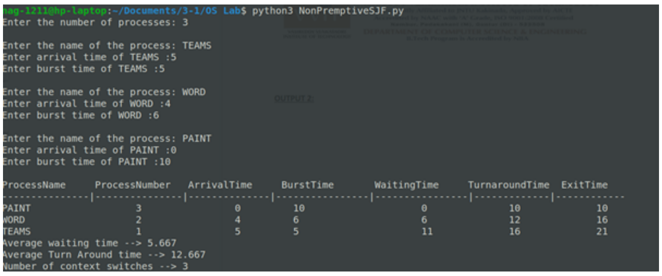


```
nag-1211@hp-laptop:~/Documents/3-1/OS Lab$ python3 NonPremptiveSJF.py
Enter the number of processes: 4

Enter the name of the process: chrome
Enter arrival time of chrome :0
Enter burst time of chrome :40

Enter the name of the process: teams
Enter arrival time of teams :0
Enter burst time of teams :20

Enter the name of the process: terminal
Enter arrival time of terminal :4
Enter burst time of terminal :30

Enter the name of the process: paint
Enter arrival time of paint :3
Enter burst time of paint :45

ProcessName     ProcessNumber   ArrivalTime    BurstTime       WaitingTime     TurnaroundTime  ExitTime
---------------|---------------|--------------|----------------|----------------|-------------|------------
teams                2                0         20                  0              20              20
chrome               1                0         40                 20              60              60
paint                4                3         45                 57             102             105
terminal             3                4         30                101             131             135
Average waiting time --> 44.5
Average Turn Around time --> 78.25
Number of context switches --> 4
```

-

-

OUTPUT 2:



```
nag-1211@hp-laptop:~/Documents/3-1/OS Lab$ python3 NonPremptiveSJF.py
Enter the number of processes: 3

Enter the name of the process: TEAMS
Enter arrival time of TEAMS :5
Enter burst time of TEAMS :5

Enter the name of the process: WORD
Enter arrival time of WORD :4
Enter burst time of WORD :6

Enter the name of the process: PAINT
Enter arrival time of PAINT :0
Enter burst time of PAINT :10

ProcessName     ProcessNumber   ArrivalTime    BurstTime       WaitingTime     TurnaroundTime  ExitTime
---------------|---------------|--------------|----------------|----------------|-------------|------------
PAINT                3                0         10                  0              10              10
WORD                 2                4         6                   6              12              16
TEAMS                1                5         5                  11              16              21
Average waiting time --> 5.667
Average Turn Around time --> 12.667
Number of context switches --> 3
```

-

EXPERIMENT NO: 1 (B) - II

AIM : Simulate Premptive SJF CPU Scheduling algorithm.

DESCRIPTION : preemptive version of SJF known as Shortest Remaining Time First (SRTF).

In this scheduling algorithm, the process with the smallest amount of time remaining until completion is selected to execute. Since the currently executing process is the one with the shortest amount of time remaining by definition, and since that time should only reduce as execution progresses, processes will always run until they complete or a new process is added that requires a smaller amount of time.

LIBRARIES USED:

Language: Python

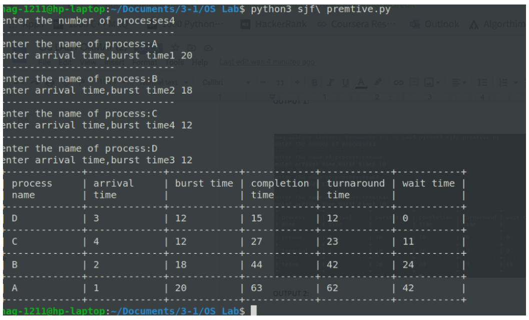
Libraries : texttable(can be installed using pip install texttable)

SYNTAX:

PROGRAM-1:

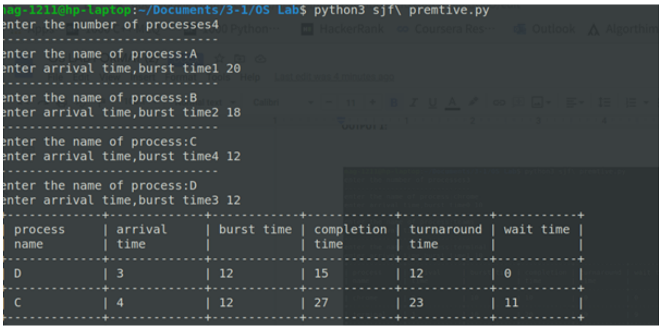
```
from texttable import Texttable

class process:
   def __init__(self, name, arrival, burst):
       self.name = name
       self.arrival = arrival
       self.rem = burst
       self.burst = burst
       self.wt = 0
       self.tt = 0
       self.ct = 0

   def to_list(self):
       return [self.name,self.arrival,self.burst,self.ct,self.tt,self.wt]

def priority(d):
   t = Texttable()
   t.add_row(["process name","arrival time","burst time","completion
time","turnaround time","wait time"])
   clock = 0
   temp = []
   while len(d) > 0:
       d= sorted(d,key=wrt_at)
       for at in d:
           if at.arrival <= clock:
               temp.append(at)
       temp = sorted(temp,key=wrt_bt)
       if len(temp)==0:
           clock+=1
           continue
       clock+=1
       temp[0].rem -=1
       if temp[0].rem ==0:
           temp[0].ct=clock
           temp[0].tt=temp[0].ct - temp[0].arrival
           temp[0].wt=temp[0].tt- temp[0].burst
           t.add_row(temp[0].to_list())
           d.remove(temp[0])
       temp.clear()
   print(t.draw())



def main():
   n = int(input("enter the number of processes"))
   d=[]
   for _ in range(n):
       print(30*'*')
       name = input("enter the name of process:")
       t = list(map(int,input("enter arrival time,birst time").split("
")))
       at = t[0]
       bt = t[1]
       d.append(process(name,at,bt))

   priority(d)

def wrt_at(x):
   return x.arrival

def wrt_bt(x):
   return x.burst

if __name__ == "__main__":
   main()
```

OUTPUT 1:

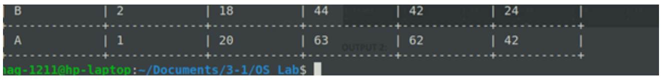


```
nag-1211@hp-laptop:~/Documents/3-1/OS Lab$ python3 sjf\ premtive.py
enter the number of processes4
------------------------------
enter the name of process:A
enter arrival time,burst time1 20
------------------------------
enter the name of process:B
enter arrival time,burst time2 18
------------------------------
enter the name of process:C
enter arrival time,burst time4 12
------------------------------
enter the name of process:D
enter arrival time,burst time3 12
+------------+------------+------------+------------+------------+-----------+
| process    | arrival    | burst time | completion | turnaround | wait time |
| name       | time       |            | time       | time       |           |
+------------+------------+------------+------------+------------+-----------+
| D          | 3          | 12         | 15         | 12         | 0         |
+------------+------------+------------+------------+------------+-----------+
| C          | 4          | 12         | 27         | 23         | 11        |
+------------+------------+------------+------------+------------+-----------+
| B          | 2          | 18         | 44         | 42         | 24        |
+------------+------------+------------+------------+------------+-----------+
| A          | 1          | 20         | 63         | 62         | 42        |
+------------+------------+------------+------------+------------+-----------+
nag-1211@hp-laptop:~/Documents/3-1/OS Lab$
```

OUTPUT 2:



```
nag-1211@hp-laptop:~/Documents/3-1/OS Lab$ python3 sjf\ premtive.py
enter the number of processes4
------------------------------
enter the name of process:A
enter arrival time,burst time1 20
------------------------------
enter the name of process:B
enter arrival time,burst time2 18
------------------------------
enter the name of process:C
enter arrival time,burst time4 12
------------------------------
enter the name of process:D
enter arrival time,burst time3 12
+------------+------------+------------+------------+------------+-----------+
| process    | arrival    | burst time | completion | turnaround | wait time |
| name       | time       |            | time       | time       |           |
+------------+------------+------------+------------+------------+-----------+
| D          | 3          | 12         | 15         | 12         | 0         |
+------------+------------+------------+------------+------------+-----------+
| C          | 4          | 12         | 27         | 23         | 11        |
+------------+------------+------------+------------+------------+-----------+
```



```
| B           | 2           | 18          | 44          | 42          | 24          |
+-------------+-------------+-------------+-------------+-------------+-------------+
| A           | 1           | 20          | 63          | 62          | 42          |
+-------------+-------------+-------------+-------------+-------------+-------------+
nag-1211@hp-laptop:~/Documents/3-1/OS Lab$
```

EXPERIMENT NO: 1 (C)

AIM : Simulate Preemptive Priority CPU Scheduling algorithm.

DESCRIPTION : Priority Scheduling is a method of scheduling processes that is based on priority. In this algorithm, the scheduler selects the tasks to work as per the priority.

The processes with higher priority should be carried out first, whereas jobs with equal priorities are carried out on a round-robin or FCFS basis. Priority depends upon memory requirements, time requirements, etc.

LIBRARIES USED:

Language: Python

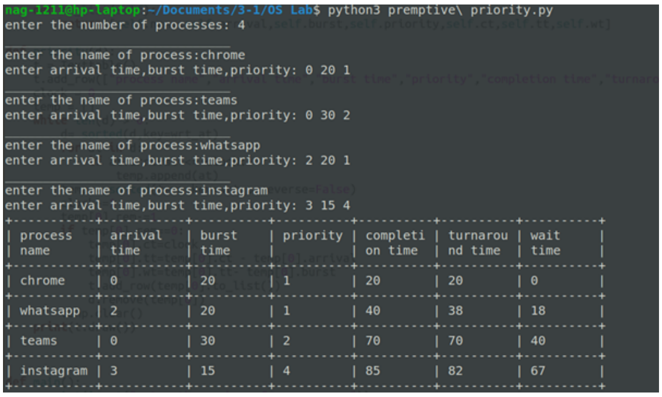
Libraries : texttable(can be installed using pip install texttable)

PROGRAM-1:

```
from texttable import Texttable

class process:
    def __init__(self, name, arrival, burst,priority):
        self.name = name
        self.arrival = arrival
        self.burst = burst
        self.rem = burst
        self.wt = 0
        self.tt = 0
        self.ct = 0
        self.priority=priority

    def to_list(self):
        return [self.name,self.arrival,self.burst,self.priority,self.ct,self.tt,self.wt]

def priority(d):
    t = Texttable()
    t.add_row(["process name","arrival time","burst time","priority","completion time","turnaround time","wait time"])
    clock = 0
    temp = []
    while len(d) > 0:
        d= sorted(d,key=wrt_at)
        for at in d:
            if at.arrival <= clock:
                temp.append(at)
        temp = sorted(temp,key=wrt_p,reverse=True)
        clock+=1
        temp[0].rem-=1
        if temp[0].rem==0:
            temp[0].ct=clock
            temp[0].tt=temp[0].ct - temp[0].arrival
            temp[0].wt=temp[0].tt- temp[0].burst
            t.add_row(temp[0].to_list())
            d.remove(temp[0])
        temp.clear()
    print(t.draw())



def main():
    n = int(input("enter the number of processes"))
    d=[]
    for _ in range(n):
        print(30*'*')
        name = input("enter the name of process:")
        t = list(map(int,input("enter arrival time,birst time,priority").split(" ")))
        at = t[0]
        bt = t[1]
        p = t[2]
        d.append(process(name,at,bt,p))
    priority(d)
```

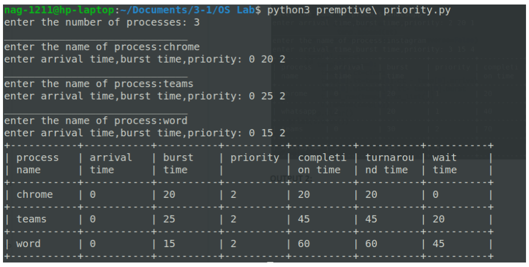
```
def wrt_at(x):
    return x.arrival


def wrt_p(x):
    return x.priority


if __name__ == "__main__":
    main()
```

OUTPUT SCREENSHOTS:



```
nag-1211@hp-laptop:~/Documents/3-1/OS Lab$ python3 premptive\ priority.py
enter the number of processes: 4
______________________________
enter the name of process:chrome
enter arrival time,burst time,priority: 0 20 1
______________________________
enter the name of process:teams
enter arrival time,burst time,priority: 0 30 2
______________________________
enter the name of process:whatsapp
enter arrival time,burst time,priority: 2 20 1
______________________________
enter the name of process:instagram
enter arrival time,burst time,priority: 3 15 4
+-----------+----------+----------+----------+----------+----------+----------+
| process   | arrival  | burst    | priority | completi | turnarou | wait     |
| name      | time     | time     |          | on time  | nd time  | time     |
+-----------+----------+----------+----------+----------+----------+----------+
| chrome    | 0        | 20       | 1        | 20       | 20       | 0        |
+-----------+----------+----------+----------+----------+----------+----------+
| whatsapp  | 2        | 20       | 1        | 40       | 38       | 18       |
+-----------+----------+----------+----------+----------+----------+----------+
| teams     | 0        | 30       | 2        | 70       | 70       | 40       |
+-----------+----------+----------+----------+----------+----------+----------+
| instagram | 3        | 15       | 4        | 85       | 82       | 67       |
+-----------+----------+----------+----------+----------+----------+----------+
```

OUTPUT 2:



```
nag-1211@hp-laptop:~/Documents/3-1/OS Lab$ python3 premptive\ priority.py
enter the number of processes: 3
______________________________
enter the name of process:chrome
enter arrival time,burst time,priority: 0 20 2
______________________________
enter the name of process:teams
enter arrival time,burst time,priority: 0 25 2
______________________________
enter the name of process:word
enter arrival time,burst time,priority: 0 15 2
+----------+----------+----------+----------+----------+----------+----------+
| process  | arrival  | burst    | priority | completi | turnarou | wait     |
| name     | time     | time     |          | on time  | nd time  | time     |
+----------+----------+----------+----------+----------+----------+----------+
| chrome   | 0        | 20       | 2        | 20       | 20       | 0        |
+----------+----------+----------+----------+----------+----------+----------+
| teams    | 0        | 25       | 2        | 45       | 45       | 20       |
+----------+----------+----------+----------+----------+----------+----------+
| word     | 0        | 15       | 2        | 60       | 60       | 45       |
+----------+----------+----------+----------+----------+----------+----------+
```

EXERCISE - 1(C)

AIM : Simulate Non-Preemptive Priority CPU Scheduling algorithm.

DESCRIPTION : Priority Scheduling is a method of scheduling processes that is based on priority. In this algorithm, the scheduler selects the tasks to work as per the priority.

The processes with higher priority should be carried out first, whereas jobs with equal priorities are carried out on a round-robin or FCFS basis. Priority depends upon memory requirements, time requirements, etc.

LIBRARIES USED:

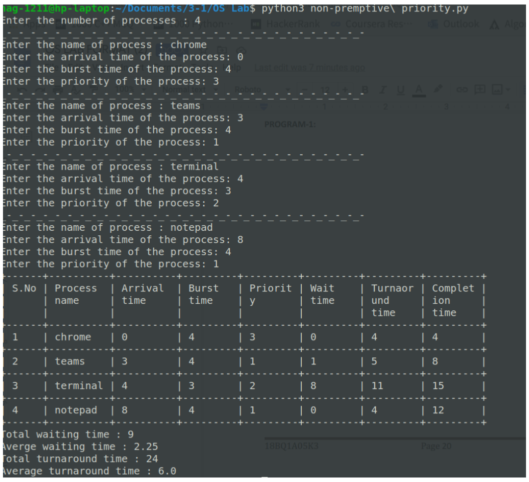
Language: Python

Libraries : texttable(can be installed using pip install texttable)

PROGRAM-1:

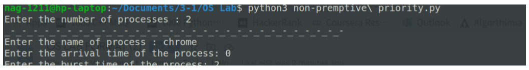
```
from texttable import Texttable


class process:
    def __init__(self, sno, name, arrival, burst,priority):
        self.sno = sno
        self.name = name
        self.arrival = arrival
        self.burst = burst
        self.wt = 0
        self.tt = 0
        self.ct = 0
        self.priority=priority


    def to_list(self):
        return [self.sno,self.name,self.arrival,self.burst,self.priority,self.wt,self.tt,self.ct]


def priority(d):
    t = Texttable()
    t.add_row(["S.No","Process name","Arrival time","Burst time","Priority","Wait
time","Turnaorund time","Completion time"])
    clock = 0
    temp = []
    l = []
    total_wt=0
    total_tt=0
    n = len(d)
    while len(d) > 0:
        d= sorted(d,key=wrt_at)
        for at in d:
            if at.arrival <= clock:
                temp.append(at)
        temp = sorted(temp,key=wrt_p)
        clock+=temp[0].burst
        temp[0].ct=clock
        temp[0].tt=temp[0].ct - temp[0].arrival
        temp[0].wt=temp[0].tt-temp[0].burst
        total_tt+=temp[0].tt
        total_wt+=temp[0].wt
        l.append(temp[0])
        d.remove(temp[0])
        temp.clear()
    l = sorted(l,key = wrt_sno)
    for i in l:
        t.add_row(i.to_list())
    print(t.draw())

    print("Total waiting time :",total_wt)
    print("Averge waiting time :",total_wt/n)

    print("Total turnaround time :",total_tt)
    print("Average turnaround time :",total_tt/n)

def wrt_at(x):
    return x.arrival

def wrt_p(x):
    return x.priority

def wrt_sno(x):
    return x.sno

if __name__ == "__main__":
    n = int(input("Enter the number of processes : "))
    d=[]
    for i in range(n):
        print(30*'_-')
        name = input("Enter the name of process : ")
        at = int(input("Enter the arrival time of the process: "))
        bt = int(input("Enter the burst time of the process: "))
        p = int(input("Enter the priority of the process: "))
        d.append(process(i+1,name,at,bt,p))


    priority(d)
```

OUTPUT SCREENSHOTS:

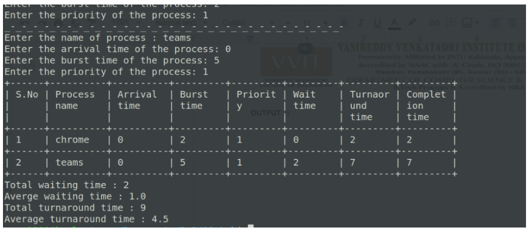


```
nag-1211@hp-laptop:~/Documents/3-1/OS Lab$ python3 non-premptive\ priority.py
Enter the number of processes : 4
_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-
Enter the name of process : chrome
Enter the arrival time of the process: 0
Enter the burst time of the process: 4
Enter the priority of the process: 3
_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-
Enter the name of process : teams
Enter the arrival time of the process: 3
Enter the burst time of the process: 4
Enter the priority of the process: 1
_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-
Enter the name of process : terminal
Enter the arrival time of the process: 4
Enter the burst time of the process: 3
Enter the priority of the process: 2
_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-
Enter the name of process : notepad
Enter the arrival time of the process: 8
Enter the burst time of the process: 4
Enter the priority of the process: 1
+------+----------+----------+---------+---------+---------+---------+---------+
| S.No | Process  | Arrival  | Burst   | Priorit | Wait    | Turnaor | Complet |
|      | name     | time     | time    | y       | time    | und     | ion     |
|      |          |          |         |         |         | time    | time    |
+------+----------+----------+---------+---------+---------+---------+---------+
| 1    | chrome   | 0        | 4       | 3       | 0       | 4       | 4       |
+------+----------+----------+---------+---------+---------+---------+---------+
| 2    | teams    | 3        | 4       | 1       | 1       | 5       | 8       |
+------+----------+----------+---------+---------+---------+---------+---------+
| 3    | terminal | 4        | 3       | 2       | 8       | 11      | 15      |
+------+----------+----------+---------+---------+---------+---------+---------+
| 4    | notepad  | 8        | 4       | 1       | 0       | 4       | 12      |
+------+----------+----------+---------+---------+---------+---------+---------+
Total waiting time : 9
Averge waiting time : 2.25
Total turnaround time : 24
Average turnaround time : 6.0
```

OUTPUT 2:



```
nag-1211@hp-laptop:~/Documents/3-1/OS Lab$ python3 non-premptive\ priority.py
Enter the number of processes : 2
_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-
Enter the name of process : chrome
Enter the arrival time of the process: 0
Enter the burst time of the process: 2
```



```
Enter the burst time of the process: 2
Enter the priority of the process: 1
 - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
Enter the name of process : teams
Enter the arrival time of the process: 0
Enter the burst time of the process: 5
Enter the priority of the process: 1
+------+----------+----------+---------+---------+---------+---------+---------+
| S.No | Process  | Arrival  | Burst   | Priorit | Wait    | Turnaor | Complet |
|      | name     | time     | time    | y       | time    | und     | ion     |
|      |          |          |         |         |         | time    | time    |
+------+----------+----------+---------+---------+---------+---------+---------+
| 1    | chrome   | 0        | 2       | 1       | 0       | 2       | 2       |
+------+----------+----------+---------+---------+---------+---------+---------+
| 2    | teams    | 0        | 5       | 1       | 2       | 7       | 7       |
+------+----------+----------+---------+---------+---------+---------+---------+
Total waiting time : 2
Averge waiting time : 1.0
Total turnaround time : 9
Average turnaround time : 4.5
```

EXERCISE - 1(D)

AIM : Simulate Round Robin CPU Scheduling algorithm.

DESCRIPTION : Round-robin (RR) is one of the algorithms employed by process and network schedulers in computing.

As the term is generally used, time slices (also known as time quanta)[3] are assigned to each process in equal portions and in circular order, handling all processes without priority (also known as cyclic executive).

Round-robin scheduling is simple, easy to implement, and starvation-free. Round-robin scheduling can be applied to other scheduling problems, such as data packet scheduling in computer networks.

LIBRARIES USED:

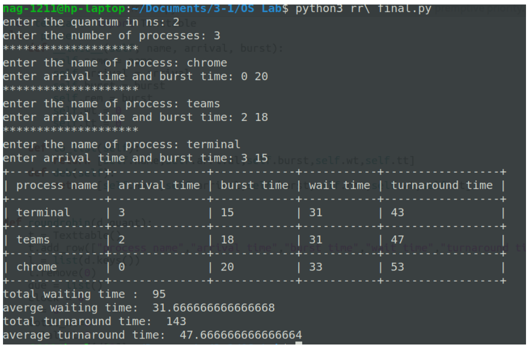
Language: Python

Libraries : texttable(can be installed using pip install texttable)

PROGRAM-1:

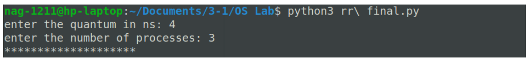
```
from texttable import Texttable
class process:
    def __init__(self, name, arrival, burst):
        self.name = name
        self.arrival = arrival
        self.burst = burst
        self.rem = burst
        self.wt = 0
        self.tt = 0

    def to_list(self):
        return [self.name,self.arrival,self.burst,self.wt,self.tt]
    def deb(self):
        return [self.name,self.arrival,self.burst,self.rem,self.wt,self.tt]



def roundrobin(d,quant):
    t = Texttable()
    t.add_row(["process name","arrival time","burst time","wait time","turnaround time"])
    l = list(d.keys())
    l.remove(0)
    que = list()
    clock = 0
    i = 0
    total_wt=0
    total_tt=0
    que.append(d[0])
    while len(que)>0:


        if que[i].rem>quant:
            clock+=quant
            que[i].rem -=quant
            for at in l:
                if at <= clock:
                    if isinstance(d[at],list):
                        que.extend(d[at])
                    else:
                        que.append(d[at])
                    l.remove(at)
                else:
                    break
            #print(que[i].deb()," clock=",clock)
            que.append(que[i])
            que.remove(que[i])
        elif que[i].rem == quant:
            clock+=quant
            que[i].rem = 0
            que[i].wt = clock - que[i].burst-que[i].arrival
            que[i].tt = que[i].wt + que[i].burst-que[i].arrival
            total_tt+=que[i].tt
            total_wt+=que[i].wt
            #print(que[i].deb()," clock=",clock)
            t.add_row(que[i].to_list())
            que.remove(que[i])

        else:
            clock+=que[i].rem
            que[i].rem=0
            #print(que[i].deb()," clock=",clock)
            que[i].wt = clock - que[i].burst-que[i].arrival
            que[i].tt = que[i].wt + que[i].burst-que[i].arrival
            total_tt+=que[i].tt
            total_wt+=que[i].wt
            t.add_row(que[i].to_list())
            que.remove(que[i])

    print(t.draw())
    print("total waiting time : ",total_wt)
    print("averge waiting time: ",(total_wt/len(d)))
    print("total turnaround time: ",total_tt)
    print("average turnaround time: ",(total_tt/len(d)))



def main():
    q = int(input("enter the quantum in ns"))
    n = int(input("enter the number of processes"))
    d,l={},[]
    for _ in range(n):
        print(20*'*')
        name = input("enter the name of process:")
        t = list(map(int,input("enter arrival time and burst time").split(" ")))
        at = t[0]
        bt = t[1]
        if at not in d.keys():
            d[at]=process(name,at,bt)
        else:
            d[at]=[d[at],process(name,at,bt)]

    roundrobin(d,q)

if __name__ == "__main__":
    main()
```

OUTPUT SCREENSHOTS:

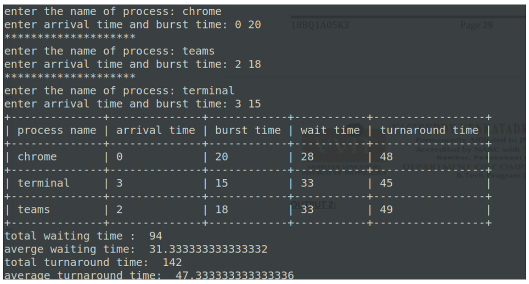


```
nag-1211@hp-laptop:~/Documents/3-1/OS Lab$ python3 rr\ final.py
enter the quantum in ns: 2
enter the number of processes: 3
********************
enter the name of process: chrome
enter arrival time and burst time: 0 20
********************
enter the name of process: teams
enter arrival time and burst time: 2 18
********************
enter the name of process: terminal
enter arrival time and burst time: 3 15
+--------------+--------------+------------+-----------+-----------------+
| process name | arrival time | burst time | wait time | turnaround time |
+--------------+--------------+------------+-----------+-----------------+
| terminal     | 3            | 15         | 31        | 43              |
+--------------+--------------+------------+-----------+-----------------+
| teams        | 2            | 18         | 31        | 47              |
+--------------+--------------+------------+-----------+-----------------+
| chrome       | 0            | 20         | 33        | 53              |
+--------------+--------------+------------+-----------+-----------------+
total waiting time :  95
averge waiting time:  31.666666666666668
total turnaround time:  143
average turnaround time:  47.666666666666664
```

OUTPUT 2:



```
nag-1211@hp-laptop:~/Documents/3-1/OS Lab$ python3 rr\ final.py
enter the quantum in ns: 4
enter the number of processes: 3
********************
```



```
enter the name of process: chrome
enter arrival time and burst time: 0 20
********************
enter the name of process: teams
enter arrival time and burst time: 2 18
********************
enter the name of process: terminal
enter arrival time and burst time: 3 15
+--------------+--------------+------------+-----------+-----------------+
| process name | arrival time | burst time | wait time | turnaround time |
+--------------+--------------+------------+-----------+-----------------+
| chrome       | 0            | 20         | 28        | 48              |
+--------------+--------------+------------+-----------+-----------------+
| terminal     | 3            | 15         | 33        | 45              |
+--------------+--------------+------------+-----------+-----------------+
| teams        | 2            | 18         | 33        | 49              |
+--------------+--------------+------------+-----------+-----------------+
total waiting time :  94
averge waiting time:  31.333333333333332
total turnaround time:  142
average turnaround time:  47.333333333333336
```