DATE : __/__/____
PAGE _____

Experiment - 3

Aim :- To WAP to implement CPU Scheduling for Shortest Job First (SJF) algorithm.

Theory :-

In SJF scheduling, the process with the lowest burst time, among the list of all available processes in the ready queue, is going to be scheduled next.

The processor should know in advance how much time process will take. Easy to implement in Batch systems where required CPU time is known in advance

Advantage :- Shortest Job First has the advantage of having minimum average waiting time among all scheduling algorithms.

Disadvantage :- It may cause starvation if shorter processes keep coming.

This scheduling algorithm is optimal if all the jobs or processes are available at the same time (either Arrival time is 0 for all, or Arrival time is same for all processes).

GOOD WRITE

DATE: __/__/____
PAGE _____

SJF is of two types :-

1. Non Pre-emptive.
2. Pre-emptive.

1.) Non Pre-emptive Shortest Job First :-

In Non Pre-emptive SJF, we schedule the processes on the basis of their burst time. We arrange the processes in increasing order of their burst time and the process having lowest burst time gets executed first and then the second.

If some processes arrive after some time, in such situation, the process have to wait for current processes' execution to finish. After execution of all the processes in the queue, new processes are added and executed on the basis of their burst time.

2.) Pre-emptive Shortest Job First :-

In Pre-emptive SJF, jobs are put into ready queue as they arrive, but as a process with short burst time arrives, the existing process is preempted or removed from execution, and the shorter job is executed first.

GOOD WRITE

Experiment-3

shortest job first

Aim: WAP to implement CPU scheduling for ~~first come first serve~~ algorithm

code:

```
print("enter burst times")
bt=(input().split(" "))
for i in range(len(bt)):
        bt[i]=int(bt[i])

print(bt)

def findAll(bt):
        print("----------------------------------------------------------")
        bt.sort()
        wt =[]
        tt=[]
        avg_wt=0
        avg_tt=0
        for i in range(len(bt)):
                if(i==0):
                        wt.append(0)
                        tt.append(wt[i]+bt[i])
                        continue
                wt.append(bt[i-1]+wt[i-1])
                tt.append(wt[i]+bt[i])
        print("sno.\tBT\tWT\tTT")
        for i in range(len(bt)):
                print(i,"\t",bt[i],"\t",wt[i],"\t",tt[i])
                avg_wt+=wt[i]
                avg_tt+=tt[i]

        print("average wt = ",avg_wt/len(wt),"\n","average tt = ",avg_tt/len(wt))
        print("----------------------------------------------------------")

findAll(bt)
print("press y to input again")
a=input()

while(a=='y'):
        print("enter new process burst time")
        bt.append(int(input()))
        findAll(bt)
        print("press y to input again")
        a=input()
```

15

4/2/19