OS EXPERIMENT 5

Name: Mohammed Arhaan Ali
Div: S11
Roll No: 02

AIM : a) To study and implement non preemptive scheduling algorithm FCFS

b) To study and implement preemptive scheduling algorithm SRTF

THEORY:

Non preemptive algorithm : FCFS, SJF

1] FCFS : First Come First Serve

It is a non-preemptive scheduling algorithm and the criteria for this is the arrival time of the process CPU is allotted to the process that requires it first. Jobs arriving later are placed at the end of the queue.

FCFS (Example)

Process	Duration	Oder	Arrival Time
P1	24	1	0
P2	3	2	0
P3	4	3	0

Gantt Chart :

P1 waiting time : 0
P2 waiting time : 24
P3 waiting time : 27

The Average waiting time :

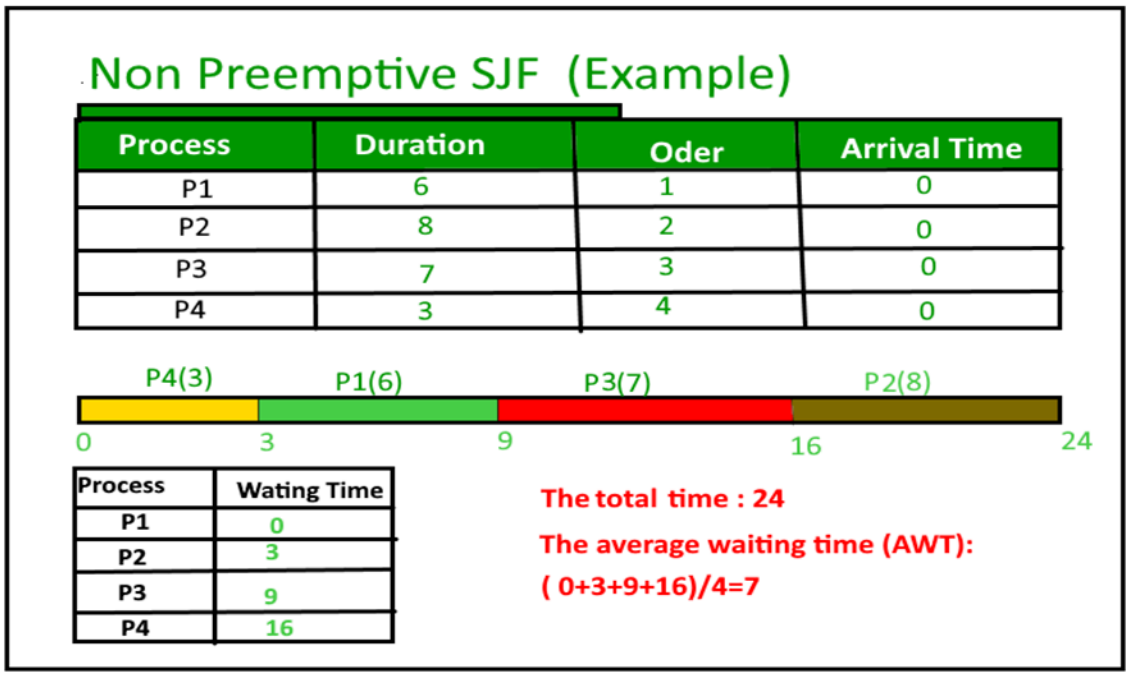
(0+24+27)/3 = 17

2] SJF : shortest job first

This is a non preemptive scheduling algorithm, which associates with each process the length of the processes next CPU first.

Criteria : Burst Time.

This algorithm assigns processes according to BT if BT of two processes is the same we use FCFS scheduling criteria.



3] Priority Scheduling

This is a non preemptive scheduling algorithm. Each process here has a priority that is either assigned already or externally done.

Process	Burst Time	Priority
P1	10	2
P2	5	0
P3	8	1

P1	P3	P2

0 10 18 23

Preemptive Scheduling Algorithm : SRTF/STRN

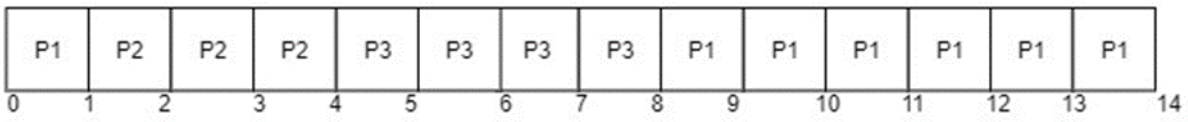
Shortest Remaining Time First / Shortest Remaining Time Next scheduling.

It is a preemptive SJF Algorithm. The choice arrives when a new process arrives as the ready queue. While a previous process is still executing. The next CPU burst if the newly arrived process may be shorter than what is left of the currently executing process. A preemptive SJF will preempt the current executing process and not allow the currently running process to finish its CPU burst.

Round Robin is also one preemptive algorithm.

Process	Burst Time	Arrival Time
P1	7	0
P2	3	1
P3	4	3

The Gantt Chart for SRTF will be:



FCFS:

CODE:

```
#include <stdio.h>
int main()
{
   int pid[15];
   int bt[15];
   int n;
   printf("Enter the number of processes: ");
   scanf("%d",&n);

   printf("Enter process id of all the processes: ");
   for(int i=0;i<n;i++)
   {
      scanf("%d",&pid[i]);
   }

   printf("Enter burst time of all the processes: ");
```

```
    for(int i=0;i<n;i++)
    {
        scanf("%d",&bt[i]);
    }

    int i, wt[n];
    wt[0]=0;

    //for calculating waiting time of each process
    for(i=1; i<n; i++)
    {
        wt[i]= bt[i-1]+ wt[i-1];
    }

    printf("Process ID     Burst Time     Waiting Time     TurnAround Time\n");
    float twt=0.0;
    float tat= 0.0;
    for(i=0; i<n; i++)
    {
        printf(" %d\t", pid[i]);
        printf("              %d\t", bt[i]);
        printf("              %d\t", wt[i]);

        //calculating and printing turnaround time of each process
        printf("              %d\t", bt[i]+wt[i]);
        printf("\n");

        //for calculating total waiting time
        twt += wt[i];

        //for calculating total turnaround time
        tat += (wt[i]+bt[i]);
    }
    float att,awt;

    //for calculating average waiting time
    awt = twt/n;

    //for calculating average turnaround time
    att = tat/n;
    printf("Avg. waiting time= %f\n",awt);
    printf("Avg. turnaround time= %f",att);
}
```

OUTPUT:

```
/tmp/7kxbNLnREN.o
Enter the number of processes: 5
Enter process id of all the processes: 1 2 3 4 5
Enter burst time of all the processes: 2 4 7 8 3
Process ID     Burst Time     Waiting Time     TurnAround Time
 1                 2                  0                  2
 2                 4                  2                  6
 3                 7                  6                  13
 4                 8                  13                 21
 5                 3                  21                 24
Avg. waiting time= 8.400000
Avg. turnaround time= 13.200000
```

SRTF:

CODE:

```
#include<stdio.h>

void main()
{
 int a[10],b[10],x[10];
 int waiting[10],turnaround[10],completion[10];
 int i,j,smallest,count=0,time,n;
 double avg=0,tt=0,end;


 printf("\nEnter the number of Processes: ");
 scanf("%d",&n);
 for(i=0;i<n;i++)
 {
   printf("\nEnter arrival time of process %d : ",i+1);
   scanf("%d",&a[i]);
 }
 for(i=0;i<n;i++)
 {
   printf("\nEnter burst time of process %d : ",i+1);
   scanf("%d",&b[i]);
 }
 for(i=0;i<n;i++)
 x[i]=b[i];

  b[9]=9999;
 //printf("time => process number");
 for(time=0;count!=n;time++)
```

```
{
  smallest=9;
 for(i=0;i<n;i++)
 {
  if(a[i]<=time && b[i]<b[smallest] && b[i]>0 )
  smallest=i;
 }
 b[smallest]--;
 //printf("\n%d => p%d",time+1,smallest);
 if(b[smallest]==0)
 {
  count++;
  end=time+1;
  completion[smallest] = end;
  waiting[smallest] = end - a[smallest] - x[smallest];
  turnaround[smallest] = end - a[smallest];
  // printf("\n %d  %d   %d",smallest,wt[smallest],ttp[smallest]);
 }
}
 printf("pid \t burst \t arrival \twaiting \tturnaround \tcompletion");
 for(i=0;i<n;i++)
 {
   printf("\n   %d \t  %d \t %d\t\t%d
\t\t%d\t\t%d",i+1,x[i],a[i],waiting[i],turnaround[i],completion[i]);
   avg = avg + waiting[i];
   tt = tt + turnaround[i];
 }
 printf("\n  %lf   %lf",avg,tt);
 printf("\n\nAverage waiting time = %lf\n",avg/n);
 printf("Average Turnaround time = %lf",tt/n);
}
```

OUTPUT:

```
PS C:\Users\arhaa\OneDrive\Desktop\OS> cd "c:\Users\a
o srtf } ; if ($?) { .\srtf }

Enter the number of Processes: 5

Enter arrival time of process 1 : 1

Enter arrival time of process 2 : 2

Enter arrival time of process 3 : 4

Enter arrival time of process 4 : 6

Enter arrival time of process 5 : 7
```

```
Enter burst time of process 1 : 2

Enter burst time of process 2 : 3

Enter burst time of process 3 : 1

Enter burst time of process 4 : 1

Enter burst time of process 5 : 6
```

```
pid      burst   arrival        waiting         turnaround      completion
   1      2      1              0               2               3
   2      3      2              2               5               7
   3      1      4              0               1               5
   4      1      6              1               2               8
   5      6      7              1               7               14
  If   If

Average waiting time = 0.800000
Average Turnaround time = 3.400000
PS C:\Users\arhaa\OneDrive\Desktop\OS>
```

Conclusion : Thus we have successfully implemented pre-emptive and non preemptive scheduling algorithms.