Ex.No.: 4 Implementation of various CPU Scheduling Algorithms

Ex.No.4a FIRST COME FIRST SERVE SCHEDULING

Date :

Aim:

To write a program in C to implement the FCFS Gantt Chart

Algorithm:

1. Start the program.
2. Get the number of process to be executed
3. Get the process name and its burst time.
4. Calculate the waiting time and turn around time for each process
5. Draw the Gantt chart using the graphics mode.
6. Stop the program

Program:

```
#include<stdio.h>
struct process
{
int btime,wtime,ttime;
}
p[50];
main()
{
   int n,i,j,h,c;
float tot_turn=0.0,tot_wait=0.0,avg_turn=0.0,avg_wait=0.0;
printf("\n\n\t\t\tFIRST COME FIRST SERVE
SCHEDULING\n\n");
printf("\t\t\t**********************************\n");
printf("Enter the number of process=");
scanf("%d",&n); printf("\n");
for(i=1;i<=n;i++)
{
printf("Enter the burst time %d:",i);
scanf("%d",&p[i].btime);
}
i=1; p[i].wtime=0;
p[i].ttime=p[i].btime;
tot_wait=p[i].wtime;
tot_turn=p[i].ttime;
for(i=2;i<=n;i++)
{
```

```
p[i].wtime=p[i-1].wtime+p[i-1].btime;
p[i].ttime=p[i].wtime+p[i].btime;
tot_wait=tot_wait+p[i].wtime;

tot_turn=tot_turn+p[i].ttime;
}
avg_wait=tot_wait/n;
avg_turn=tot_turn/n;
printf("\nProcess No \tBurst Time\tWaiting Time\tTurn Around
Time");
for(i=1;i<=n;i++)
{
printf("\n%d \t\t\t%d\t\t\t%d
\t\t\t\t%d",i,p[i].btime,p[i].wtime,p[i].ttime);
}
printf("\n\nAverage Waiting Time=%f",avg_wait);
printf("\nAverage Turn Around Time=%f",avg_turn);
printf("\n");
printf("\n\t\t\t\tGANTT CHART");
printf("\n\t\t\t\t**********\n\n");
for(i=1;i<=n;i++)
{
   printf("%d",p[i].wtime);
   for(j=1;j<=p[i].btime;j++)
   printf("_"); }
   for(i=1;i<=n;i++)
{
c=p[i].wtime+p[i].btime;
}
printf("%d",c);
printf("\n\n");
return 0;
}
```

Output:

```
                FIRST COME FIRST SERVE SCHEDULING

                *********************************
Enter the number of process=3

Enter the burst time 1:8
Enter the burst time 2:6
Enter the burst time 3:2

Process No  Burst Time  Waiting Time    Turn Around Time
1           8           0               8
2           6           8               14
3           2           14              16

Average Waiting Time=7.333333
Average Turn Around Time=12.666667

                GANTT CHART
                ***********

0________8______14__16
```

Result:

Thus the program to implement FCFS scheduling algorithm has been written and executed successfully.

Ex.No.4b **Shortest Job First Scheduling**
Date :

Aim:

To write a program in C to implement the SJF scheduling algorithm.

Algorithm:

1. Start the process.
2. Declare the array size.
3. Get the number of elements to be inserted.
4. Select the process which has shortest burst time will execute first.
5. If two processes have same burst length then FCFS scheduling algorithm used.
6. Make the average waiting length of next process.
7. Start with the first process from its selection as above and let the other process in queue.
8. Calculate the total number of burst time
9. Display the values.
10. Terminate the process.

Program

```
#include<stdio.h>

main()

{

int i,j,n,t,d,h,tot=0,tt=0,p[20],c[20],a[20];

printf("\n\t\t\t\tSHORTEST JOB FIRST SCHEDULING\n");

printf("\t\t\t\t*****************************\n\n");

printf("Enter the number of process:");

scanf("%d",&n);

printf("\nEnter the %d process\n",n);

for(i=0;i<n;i++)

scanf("%d",&p[i]);

for(i=0;i<n-1;i++)

for(j=i+1;j<n;j++)

if(p[i]>p[j])

{

   t=p[i];

p[i]=p[j];
```

```
p[j]=t;
    }
printf("\nSorted Process\n");
for(i=0;i<n;i++)
printf("%d\n",p[i]); c[0]=0; for(i=0;i<n-1;i++) c[i+1]=c[i]+p[i];
for(i=0;i<n;i++) a[i]=c[i]+p[i];
printf("\nP.No \tProcess \tWaiting Time \tTurn Around Time");
for(i=0;i<n;i++)
{
printf("\n%d\t%d\t\t%d\t\t%d",i+1,p[i],c[i],a[i]);
tot=tot+c[i]; tt=tt+a[i]; }
printf("\n\nAverage Waiting Time %f",((float)tot/n));
printf("\nAverage Turn Around Time %f",((float)tt/n));
printf("\n");
printf("\n\n\t\t\t\t\tGANTT CHART");
printf("\n\n\t\t\t\t\t**********");
printf("\n\n\t\t\t\t");
for(i=0;i<n;i++)
{
printf("%d",c[i]);
for(j=1;j<p[i];j++)
printf("_"); }
for(i=1;i<n;i++)
{
   d=c[i]+p[i];
}
printf("%d",d);
printf("\n\n"); return 0;
}
```

Output:

```
                SHORTEST JOB FIRST SCHEDULING
                ******************************

Enter the number of process:3

Enter the 3 process
9
7
3

Sorted Process
3
7
9

P.No    Process     Waiting Time    Turn Around Time
1       3           0               3
2       7           3               10
3       9           10              19

Average Waiting Time 4.333333
Average Turn Around Time 10.666667


                    GANTT CHART

                    ***********

               0__3_____10_______19
```

Result:

Thus the program to implement shortest job first scheduling algorithm has been written and executed successfully

Ex.No.4c **Priority Scheduling**

Date :

Aim:

To write a program in C to implement the priority scheduling algorithm.

Algorithm:

1. Start the program.
2. Initialize the variables in structure.
3. Get the number of process, priority and burst time from the user.
4. Start the process execution according to the priority.
5. The total execution time is calculated by adding the burst time.
6. Calculate the average waiting time and turnaround time using total execution and waiting time
7. Terminate the program.

Program:

```
#include<stdio.h>
main()
{ int n,b[10],w[10],i,j,h,t,tt,d;
int stime[10],a[10],p[10];
float avg=0;
printf("\n\t\t\t\tPRIORITY SCHEDULING ALGORITHM");
printf("\n\t\t\t\t*****************************\n");
printf("Enter how many jobs:");
scanf("%d",&n);
printf("\nEnter burst time & priority for corresponding job\n\n");
for(i=1;i<=n;i++)
{
   printf("Process %d:",i);
scanf("%d %d",&b[i],&p[i]);
a[i]=i; }
for(i=1;i<=n;i++)
for(j=i;j<=n;j++)
```

```
if(p[i]>p[j])
{
   t=b[i];
tt=a[i];
b[i]=b[j];
a[i]=a[j];
b[j]=t;
a[j]=tt;
   }
w[1]=0;
printf("\nProcess %d Waiting Time:0",a[1]);
for(i=2;i<=n;i++)
{
w[i]=b[i-1]+w[i-1];
printf("\nProcess %d waiting time:%d",a[i],w[i]); avg+=w[i]; }
printf("\nTotal Waiting Time:%f",avg);
printf("\nAverage Waiting Time=%f\n",avg/n);
printf("\nGANTT CHART"); printf("\n**********\n\n");
for(i=1;i<=n;i++)
{
   printf("%d ",b[i]); }
printf("\n\n");
for(i=1;i<=n;i++)
{
printf("%d",w[i]);
for(j=1;j<=b[i];j++)
printf("_"); }
for(i=1;i<=n;i++)
{ d=w[i]+b[i];
}
```

```
printf("%d",d);
return 0;
}
```

Output:

```
                PRIORITY SCHEDULING ALGORITHM
                ****************************
Enter how many jobs:3

Enter burst time & priority for corresponding job

Process 1:4 2
Process 2:6 1
Process 3:10 3

Process 2 Waiting Time:0
Process 1 waiting time:6
Process 3 waiting time:10
Total Waiting Time:16.000000
Average Waiting Time=5.333333

GANTT CHART
***********

6 4 10

0______6____10__________20
```

Result:

Thus the program to implement priority scheduling algorithm has been written and executed successfully.