**EXP NO :6 IMPLEMENTATION OF PROCESS SCHEDULING**

**(a)FIRST COME FIRST SERVED**

**AIM:**

To implement FCFS (First Come First Served) in CPU scheduling algorithm using C program.

**ALGORITHM:**

**STEP 1:** Start the program.

**STEP 2:** Declare the arrays waiting time(wt), burst time(bt), turn around time(tat), arrival time(at), completion time(ct) as integer data type.

**STEP 3:** Read the number of processes in which we want to implement the FCFS. Read the arrival time and burst time for each process.

**STEP 4:** Using for loop, completion time, turnaround time, waiting time are calculated

**STEP 5:** Completion time for all processes is equal to sum of the burst time. Turnaround time is equal to the difference between completion time and arrival time.

**STEP 6:** Waiting time is equal to the difference in turnaround time and burst time.

**STEP 7:** Then corresponding total turnaround time and total waiting time are calculated. Then calculated turnaround time is displayed.

**STEP 8:** Stop the program

**CODE:**

#include<stdio.h>

int main()

{

int n,bt[20],wt[20],tat[20],avwt=0,avtat=0,i,j;

printf("Enter total number of processes:");

scanf("%d",&n);

printf("Enter Burst Time");

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

{

printf("P[%d]:",i+1);

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

}

wt[0]=0;

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

{

wt[i]=0;

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

wt[i]+=bt[j];

}

printf("\nProcess\t\tBurst Time\tWaiting Time\tTurnaround Time");

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

{

tat[i]=bt[i]+wt[i];

avwt+=wt[i];

avtat+=tat[i];

printf("\nP[%d]\t\t%d\t\t%d\t\t%d",i+1,bt[i],wt[i],tat[i]);

}

avwt/=i;

avtat/=i;

printf("\nAverage Waiting Time:%d",avwt);

printf("\nAverage Turnaround Time:%d",avtat);

return 0;

}

**OUTPUT:**

Enter total number of processes:5

Enter Burst Time P[1]:5

P[2]:6

P[3]:3

P[4]:8

P[5]:4

Process Burst Time Waiting Time Turnaround Time

P[1] 5 0 5

P[2] 6 5 11

P[3] 3 11 14

P[4] 8 14 22

P[5] 4 22 26

Average Waiting Time:10

Average Turnaround Time:15

**IMPLEMENTATION OF PROCESS SCHEDULING**

**(b)SHORTEST JOB FIRST**

**AIM:**

To implement SJF (Shortest Job First) in CPU scheduling algorithm using C program.

**ALGORITHM:**

**STEP 1:** Start the program.

**STEP 2**: Declare the arrays burst time,turn around time,waiting time and the number of process of integer data type and average waiting time and average turn around time as float data type.

**STEP 3**: Read the values of the burst time for the processes. Then sorting of burst time takes place. This program executes according to the sorted order of the burst time

**STEP 4:** According to the burst time that are sorted the processes are rescheduled and also the waiting time and turn around time are calculated.

**STEP 5:** The corresponding average waiting time and average turnaround time are calculated. Then the result has been displayed.

**STEP 6:** Stop the program.

**CODE:**

#include<stdio.h>

int main()

{

int bt[20],p[20],wt[20],tat[20],i,j,n,total=0,pos,temp;

float avg\_wt,avg\_tat;

printf("Enter number of process:");

scanf("%d",&n);

printf("\nEnter Burst Time:\n");

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

{

printf("p[%d]:",i+1);

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

p[i]=i+1;

}

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

{

pos=i;

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

{

if(bt[j]<bt[pos])

pos=j;

}

temp=bt[i];

bt[i]=bt[pos];

bt[pos]=temp;

temp=p[i];

p[i]=p[pos];

p[pos]=temp;

}

wt[0]=0;

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

{

wt[i]=0;

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

wt[i]+=bt[j];

total+=wt[i];

}

avg\_wt=(float)total/n;

total=0;

printf("\nProcess\t Burst Time \tWaiting Time\tTurnaround Time");

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

{

tat[i]=bt[i]+wt[i];

total+=tat[i];

printf("\np[%d]\t\t %d\t\t %d\t\t\t%d",p[i],bt[i],wt[i],tat[i]);

}

avg\_tat=(float)total/n;

printf("\nAverage Waiting Time=%f",avg\_wt);

printf("\nAverage Turnaround Time=%fn",avg\_tat);

}

**OUTPUT:**

Enter number of process:6

Enter Burst Time:

p[1]:4

p[2]:8

p[3]:6

p[4]:7

p[5]:9

p[6]:1

Process Burst Time Waiting Time Turnaround Time

p[6] 1 0 1

p[1] 4 1 5

p[3] 6 5 11

p[4] 7 11 18

p[2] 8 18 26

p[5] 9 26 35

Average Waiting Time=10.166667

Average Turnaround Time=16.000000

**IMPLEMENTATION OF PROCESS SCHEDULING**

**(c)PRIORITY SCHEDULING**

**AIM:**

To implement Priority Scheduling in CPU scheduling algorithm using C program.

**ALGORITHM:**

**STEP 1:** Start the program.

**STEP 2:** Declare the array for burst time, priority, turn around time and some iteratable variables for using it in a looping structure.

**STEP 3:** Read the burst time for all processes. The processes are sorted based on the priority.

**STEP 4:** As the name suggests, the processes are sorted based on the priority. The waiting time has been initialised to zero. The waiting time and turn around time has been calculated.

**STEP 5:** The average waiting time and average turn around time has been calculated.

**STEP 6:** Stop the program.

**CODE:**

#include<stdio.h>

int main ()

{

int bt[20],p[20],wt[20],tat[20],i,j,n,total=0,pos,temp;

float avg\_wt,avg\_tat;

printf("Enter number of process:");

scanf("%d",&n);

printf("\nEnter Burst Time:\n");

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

{

printf("p[%d]:",i+1);

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

p[i]=i+1;

}

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

{

pos=i;

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

{

if(bt[j]<bt[pos])

pos=j;

}

temp=bt[i];

bt[i]=bt[pos];

bt[pos]=temp;

temp=p[i];

p[i]=p[pos];

p[pos]=temp;

}

wt[0]=0;

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

{

wt[i]=0;

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

wt[i]+=bt[j];

total+=wt[i];

}

avg\_wt=(float)total/n;

total=0;

printf("\nProcess\t Burst Time \tWaiting Time\tTurnaround Time");

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

{

tat[i]=bt[i]+wt[i];

total+=tat[i];

printf("\np[%d]\t\t %d\t\t %d\t\t\t%d",p[i],bt[i],wt[i],tat[i]);

}

avg\_tat=(float)total/n;

printf("\nAverage Waiting Time=%f",avg\_wt);

printf("\nAverage Turnaround Time=%fn",avg\_tat);

}

**OUTPUT:**

Enter Total Number of Process:5

Enter Burst Time and Priority

P[1]

Burst Time:2

Priority:2

P[2]

Burst Time:5

Priority:1

P[3]

Burst Time:8

Priority:3

P[4]

Burst Time:5

Priority:5

P[5]

Burst Time:9

Priority:4

Process Burst Time Waiting Time Turnaround Time

P[2] 5 0 5

P[1] 2 5 7

P[3] 8 7 15

P[5] 9 15 24

P[4] 5 24 29

Average Waiting Time=10

Average Turnaround Time=16

**IMPLEMENTATION OF PROCESS SCHEDULING**

**(d)ROUND ROBIN SCHEDULING**

**AIM:**

To implement Priority Scheduling in CPU scheduling algorithm using C program.

**ALGORITHM:**

**STEP 1:** Start the program.

**STEP 2:** Declare the arrays arrival time, burst time, turn around time as integer data type and average waiting time and average turn around time as float data type. Read the total number of processes.

**STEP 3:** Read the time quantum for the processes and burst time for each process.

**STEP 4:** Each process should run till the time quantum. After that it skips that processes and move on to the next process. waiting time and turn around time are calculated.

**STEP 5:** Average waiting time and average turnaround time are calculated.

**STEP 6:** Stop the program.

**CODE:**

#include<stdio.h>

int main()

{

int i, limit, total = 0, x, counter = 0, time\_quantum;

int wait\_time = 0, turnaround\_time = 0, arrival\_time[10], burst\_time[10], temp[10];

float average\_wait\_time, average\_turnaround\_time;

printf("\nEnter Total Number of Processes:\t");

scanf("%d", &limit);

x = limit;

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

{

printf("\nEnter Details of Process[%d]\n", i + 1);

printf("Arrival Time:\t");

scanf("%d", &arrival\_time[i]);

printf("Burst Time:\t");

scanf("%d", &burst\_time[i]);

temp[i] = burst\_time[i];

}

printf("\nEnter Time Quantum:\t");

scanf("%d", &time\_quantum);

printf("\nProcess ID\t\tBurst Time\t Turnaround Time\t Waiting Time\n");

for(total = 0, i = 0; x != 0;)

{

if(temp[i] <= time\_quantum && temp[i] > 0)

{

total = total + temp[i];

temp[i] = 0;

counter = 1;

}

else if(temp[i] > 0)

{

temp[i] = temp[i] - time\_quantum;

total = total + time\_quantum;

}

if(temp[i] == 0 && counter == 1)

{

x--;

printf("\nProcess[%d]\t\t%d\t\t %d\t\t\t %d", i + 1, burst\_time[i], total - arrival\_time[i], total - arrival\_time[i] - burst\_time[i]);

wait\_time = wait\_time + total - arrival\_time[i] - burst\_time[i];

turnaround\_time = turnaround\_time + total - arrival\_time[i];

counter = 0;

}

if(i == limit - 1)

{

i = 0;

}

else if(arrival\_time[i + 1] <= total)

{

i++;

}

else

{

i = 0;

}

}

average\_wait\_time = wait\_time \* 1.0 / limit;

average\_turnaround\_time = turnaround\_time \* 1.0 / limit;

printf("\nAverage Waiting Time:\t%f", average\_wait\_time);

printf("\nAvg Turnaround Time:\t%f\n", average\_turnaround\_time);

return 0;

}

**OUTPUT:**

Enter Total Number of Processes: 4

Enter Details of Process[1]

Arrival Time: 0

Burst Time: 4

Enter Details of Process[2]

Arrival Time: 1

Burst Time: 7

Enter Details of Process[3]

Arrival Time: 2

Burst Time: 5

Enter Details of Process[4]

Arrival Time: 3

Burst Time: 6

Enter Time Quantum: 3

Process ID Burst Time Turnaround Time Waiting Time

Process[1] 4 13 9

Process[3] 5 16 11

Process[4] 6 18 12

Process[2] 7 21 14

Average Waiting Time: 11.500000

Avg Turnaround Time: 17.000000

**IMPLEMENTATION OF PROCESS SCHEDULING**

**(e)SHORTEST REMAINING TIME FIRST**

**AIM:**

To implement Shortest Remining Time First in CPU scheduling algorithm using C program.

**ALGORITHM:**

**STEP 1:** Start the program.

**STEP 2:** Declare the arrays burst time, arrival time, waiting time, turn around time and iteratable variables are of integer data type

**STEP 3:** Read the number of processes, burst time, arrival time as the input.

**STEP 4:** Using arrival time calculate which process has to execute first by comparing the burst time.

**STEP 5:** Calculate the waiting time, turn around time, and their average waiting time and average turnaround time.

**STEP 6:** Stop the program.

**CODE:**

#include<stdio.h>

#include<conio.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);

printf("\n Enter arrival time and burst time : ");

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

{

printf("\nP[%d]\t",i+1);

printf("Arrival Time: ");

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

printf("\tBurst time:");

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

}

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

x[i]=b[i];

b[9]=9999;

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]--;

if(b[smallest]==0)

{

count++;

end=time+1;

completion[smallest] = end;

waiting[smallest] = end - a[smallest] - x[smallest];

turnaround[smallest] = end - a[smallest];

}

}

printf("Process\tburst\t arrival\t waiting\tturnaround\tcompletion");

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

{

printf("\n %d \t %d\t\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\nAverage waiting time = %lf\n",avg/n);

printf("Average Turnaround time = %lf",tt/n);

getch();

}

**OUTPUT:**

Enter the number of Processes: 5

Enter arrival time and burst time :

P[1] Arrival Time: 5

Burst time:4

P[2] Arrival Time: 6

Burst time:4

P[3] Arrival Time: 8

Burst time:1

P[4] Arrival Time: 6

Burst time:5

P[5] Arrival Time: 8

Burst time:2

Process burst arrival waiting turnaround completion

1 4 5 0 4 9

2 4 6 6 10 16

3 1 8 1 2 10

4 5 6 10 15 21

5 2 8 2 4 12

Average waiting time = 3.800000 Average Turnaround time = 7.000000

**IMPLEMENTATION OF PROCESS SCHEDULING**

**(f)MULTILEVEL QUEUE SCHEDULING**

**AIM:**

To implement Multilevel Queue Scheduling in CPU scheduling algorithm using C program.

**ALGORITHM:**

**STEP 1:** Start the program.

**STEP 2:** Read the number of processes and also using for loop get the burst time for all the processes.

**STEP 3:** Also get the value whether the process is a system process or user process.

**STEP 4:** According to the process that it is a system process or user process, sort the process according to the priority.

**STEP 5:** Initialise the waiting time and average waiting time as zero.

**STEP 6:** The waiting time is calculated by adding the waiting time and burst time of the previous process. Turnaround time is calculated by adding the turnaround time of the previous process and burst time of the current process.

**STEP 7:** The average waiting time and average turnaround time are calculated.

**STEP 8:** Stop the program.

**CODE:**

#include<stdio.h>

int main()

{

int p[20],bt[20], su[20], wt[20],tat[20],i, k, n, temp;

float wtavg, tatavg;

printf("Enter the number of processes --- ");

scanf("%d",&n);

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

{

p[i] = i;

printf("Enter the Burst Time of Process %d --- ", i);

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

printf("System/User Process (0/1) ? --- ");

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

}

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

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

if(su[i] > su[k])

{

temp=p[i];

p[i]=p[k];

p[k]=temp;

temp=bt[i];

bt[i]=bt[k];

bt[k]=temp;

temp=su[i];

su[i]=su[k];

su[k]=temp;

}

wtavg = wt[0] = 0;

tatavg = tat[0] = bt[0];

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

{

wt[i] = wt[i-1] + bt[i-1];

tat[i] = tat[i-1] + bt[i];

wtavg = wtavg + wt[i];

tatavg = tatavg + tat[i];

}

printf("\nPROCESS\tSYSTEM/USER PROCESS\tBURST TIME\tWAITING TIME\tTURNAROUND TIME");

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

printf("\n%d \t\t %d \t\t %d \t\t %d \t%d ",p[i],su[i],bt[i],wt[i],tat[i]);

printf("\nAverage Waiting Time is --- %f",wtavg/n);

printf("\nAverage Turnaround Time is --- %f",tatavg/n);

return 0;

}

**OUTPUT:**

Enter the number of processes --- 4

Enter the Burst Time of Process 0 --- 5

System/User Process (0/1) ? --- 0

Enter the Burst Time of Process 1 --- 6

System/User Process (0/1) ? --- 1

Enter the Burst Time of Process 2 --- 6

System/User Process (0/1) ? --- 1

Enter the Burst Time of Process 3 --- 7

System/User Process (0/1) ? --- 0

Process (system/user) burst time waiting time turn around time

0 0 5 0 5

3 0 7 5 12

2 1 6 12 18

1 1 6 18 24

Average Waiting Time is --- 8.750000

Average Turnaround Time is --- 14.750000

**RESULT:**

Thus, the given scheduling algorithms FCFS, SJF, Priority Scheduling, Round Robin Scheduling, Shortest remaining time first, Multilevel Queue Scheduling in a CPU scheduling algorithm are executed and the required output is obtained.