

**School of computer science and engineering**

**Name of the faculty : Dr.Baljit Singh Saini**

Programining assignment

**Course Title** : Operating System

**Course code**: CSE316

**NAME : Kondapaka Nithish**

**REGISTRATION NUMBER : 11805798**

**SECTION : K18VQ**

CA \_3

MAX MARKS : 30

Q2) Consider a scheduling approach which is non pre-emptive similar to shortest job next in nature. The priority of each job is dependent on its estimated run time, and also the amount of time it has spent waiting. Jobs gain higher priority the longer they wait, which prevents indefinite postponement. The jobs that have spent a long time waiting compete against those estimated to have short run times. The priority can be computed as : Priority = 1+ Waiting time / Estimated run time Write a program to implement such an algorithm

Code:

#include<stdio.h>

void 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 Estimated Time:\n");

   for(i=0;i<n;i++)

 {

 printf("p%d:",i+1);

 scanf("%d",&bt[i]);

 p[i]=i+1;           //contains process number

}

  //sorting burst time in ascending order using selection sort

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; //waiting time for first process will be zero

   //calculate waiting time

 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; //average waiting time

 total=0;

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

   for(i=0;i<n;i++)

   {

tat[i]=bt[i]+wt[i];  //calculate turnaround time

       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; //average turnaround time

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

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

}

Output

Order in which solution we get from above program

4 2 3 1

Processes Burst time Waiting time Turn around time

4 2 0 2

2 3 2 5

3 6 5 11

1 21 11 32

Average waiting time = 4.5

Average turn around time = 12.5

**What is an Operating system ?**

An Operating System (OS) is an interface between a computer user and computer hardware. An operating system is a software which performs all the basic tasks like file management, memory management, process management, handling input and output, and controlling peripheral devices such as disk drives and printers.

**Following are some of important functions of an operating System**.

• Memory Management.

• Processor Management.

• Device Management.

• File Management.

• Security.

• Control over system performance.

• Job accounting.

• Error detecting aids.

• Coordination between other software and users.

**What is Process Scheduling?**

Process Scheduling is an OS task that schedules processes of different states like ready, waiting, and running. Process scheduling allows OS to allocate a time interval of CPU execution for each process. Another important reason for using a process scheduling system is that it keeps the CPU busy all the time. This allows you to get the minimum response time for programs.

**Scheduling Objectives**

Here, are important objectives of Process scheduling

• Maximize the number of interactive users within acceptable response times.

• Achieve a balance between response and utilization.

• Avoid indefinite postponement and enforce priorities.

• It also should give reference to the processes holding the key resources.

**Shortest Job First**

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.

**How to compute below times in SJF ?**

1. Completion Time: Time at which process completes its execution.

2. Turn Around Time: Time Difference between completion time and arrival time. Turn Around Time = Completion Time – Arrival Time

3. Waiting Time(W.T): Time Difference between turn around time and burst time. Waiting Time = Turn Around Time – Burst Time