Student Name: Mausam Neupane

Student ID :1179997

Email Address:kazibhai1999@gmail.com

GitHub Link : <https://github.com/TARIKAHO/OS-PROJECT>

Question: 14

 Write a program to implement priority scheduling algorithm with context switching time. Prompt to user to enter the number of processes and then enter their priority, burst time and arrival time also. Now whenever operating system preempts a process and shifts cpu shifts control to some another process of higher priority assume that it takes 2 seconds for context switching (dispatcher latency).Form a scenario, where we can give the processes are assigned with priority where the lower integer number is higher priority and then context switch .. as the process waits the priority of the process increase at rate of one per 2 time units of wait.

Calculate waiting time and turnaround time for each process.

## Entire Code:

#include<stdio.h>

#include<conio.h>

int main()

{

const int max=30;

char first[12],last[12],feed1[100],q[2],b[12],d[100];

int te=0,a,w,i,n,temp,j,pos,pr[max],qq[max],at[max],tat[max],ll[max];

float ta=0,aw=0;

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

printf("\t \t \t \t \t |====>PREEMPTIVE PRIORITY SCHEDULING CALCULATOR<====| \n");

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

printf("ENTER YOUR DETAIL TO ACCESS THE CALCULATOR:");

printf("\nenter your first name:\n");

gets(first);

fflush(stdin);

printf("\nenter your last name:\n");

gets(last);

printf("\n!!!!!!!!!!!!!!!!!!!!!!!!!! DONT FORGET TO FIVE US FEEDBACK MR.%s %s",first,last);

printf(" AT LAST !!!!!!!!!!!!!!!!!!!!!!!!\n");

printf("\n\n\t\t\tKNOW DETAIL ABOUT THE SCHEDULING CALCULATOR ");

fflush(stdin);

printf("\n\n\t\t=============> PRIORITY SCHEDULING<=============\n\n");

printf("\n Priority scheduling is a non-preemptive algorithm and one of the most common scheduling algorithms in batch systems.\n Each process is assigned a priority.\n Process with the highest priority is to be executed first and so on.\n Processes with the same priority are executed on first come first served basis.\n Priority can be decided based on memory requirements, time requirements or any other resource requirement.");

printf("\n\n");

printf("\n \n ===>1....BURST TIME::");

printf("CPU burst is when the process is being executed in the CPU.\n I/O burst is when the CPU is waiting for I/O for further execution.");

printf("\n \n\n===>2...ARRIVAL TIME::");

printf("It is the time at which a process arrives at the ready queue");

printf("\n\n \n ===>3....COMPLETION TIME::");

printf("It is the time at which a process completes its execution");

printf("\n \n\n ===>4....WAITING TIME::");

printf("\n waiting time is how much time a process spends in the ready queue waiting for theifr turn to get on the cpu");

printf("\n \n Waiting Time is calculated by ::");

printf("TURN AROUND TIME - BURST TIME ");

printf("\n \n\n ===>5....TURN AROUND TIME::");

printf("\n Turn Around Time is Time required for a particular process to complete from submission time to completion");

printf("\n \n Turn Around Time is calculated by ::");

printf("TURN AROUND TIME - BURST TIME ");

printf("\n");

printf("\n");

printf("\t \t \t NOW PRESS ANY KEY TO CONTINUE TO START CALCULATOR OR PRESS (N/n) TO END HERE: ");

gets(q);

if(q=="N" || q=="n")

{

printf("\n\n============>THANKS FOR USING OUR CALCULATOR<=====================");

printf("\n\n=======>ENTER YOUR FEEDBACK HERE: ");

gets(feed1);

}

else

{

loop3:

printf("\n \n enter the number of process :: ");

scanf("%d",&n);

printf("\n \t \t \t \t ENTER THE BURST TIME:- ");

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

{

printf("\nenter the burst time of process %d ::",i);

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

}

printf("\n \t \t \t \tENTER THE PRIORITY:-");

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

{

printf("\n enter priority of process %d ::",i);

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

}

printf("\n \t \t \t \t ENTER THE ARRIVAL TIME:- ");

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

{

printf("\n enter the arrival time of process %d ::",i);

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

}

printf("\n\n\n ------------------------------------------------------------------------------------------------------------------------------------");

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

{

pos=i;

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

{

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

{

pos=j;

te==1;

}

}

temp=pr[i];

pr[i]=pr[pos];

pr[pos]=temp;

temp=qq[i];

qq[i]=qq[pos];

qq[pos]=temp;

temp=at[i];

at[i]=at[pos];

at[pos]=temp;

}

ll[0]=0;

printf("\n \n process \tburst time \t \t priority \t \t arrival time \t \twaiting time \t \tturn around time \n");

printf("\n ------------------------------------------------------------------------------------------------------------------------------------");

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

{

ll[i]=0;

if(te==1)

{

ll[i]+=2;

pr[i]=pr[i]\*2;

}

tat[i]=0;

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

{

ll[i]=ll[i]+qq[j];

}

tat[i]=ll[i]+qq[i];

aw=aw+ll[i];

ta=ta+tat[i];

printf("\n %d\t \t %d\t\t\t %d\t\t\t %d \t\t\t %d \t\t\t\t\t%d \n",i,qq[i],pr[i],at[i],ll[i],tat[i]);

}

aw=aw/n;

ta=ta/n;

printf("\n ------------------------------------------------------------------------------------------------------------------------------------");

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

printf("\n average waiting time is :: %f\n",aw);

printf("\n average turnaround time is :: %f\n",ta);

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

printf("\n\n\n\t\t\t\t==>enter 1 to continue CALCULTOR:");

printf("\n\t\t\t\t==>enter any charcter to end CALCULATOR:");

printf("\n\t \t \t\t\t\t\t==> CHOOSE ANY FROM ABOVE:-");

scanf("%d",&w);

switch(w)

{

case 1:

goto loop3;

default:

{

printf("\n\n============>THANKS FOR USING OUR CALCULATOR<=====================");

printf("\n\n=======>ENTER YOUR FEEDBACK HERE: ");

fflush(stdin);

gets(d);

printf("\n\t--------------------------------------------------------press any key double to save-----------------------------------------------");

getch();

getch();

printf("\n\n\n\n\t\t\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*!!!!!\*\*\*\*\*\*\* FEEDBACK SAVED THANKS FOR FEEDBACK \*\*\*\*\*\*\*\*!!!!!\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n\n\n\n\n");

}

}

}

return 0;

}

## Description:

# PRIORITY SCHEDULING :

Priority scheduling is a non-preemptive algorithm and one of the most common scheduling algorithms in batch systems. Each process is assigned a priority. Process with the highest priority is to be executed first and so on. Processes with the same priority are executed on first come first served basis. Priority can be decided based on memory requirements, time requirements or any other resource requirement.

## ****Priority Scheduling Working:****

The Priority scheduling working is based on priority of each process. In this method the scheduler chooses the tasks to work as per the priority. Which is different from other types .Priority scheduling involves priority assignment to each process ,and processes with higher priorities are carries out first where as processes with equal priorities are carried out on a first-come -first- served A general example of priority scheduling is Shortest-job-first(SJF) algorithm.

Priorities can be either dynamic or static. Static priorities are allocated during creation, whereas dynamic priorities are assigned depending on the behavior of the processes while in the system. To illustrate, the scheduler could favor input/output (I/O) intensive tasks, which lets expensive requests to be issued as soon as possible.

Priorities may be defined internally or externally. Internally defined priorities make use of some measurable quantity to calculate the priority of a given process. In contrast, external priorities are defined using criteria beyond the operating system (OS), which can include the significance of the process, the type as well as the sum of resources being utilized for computer use, user preference, commerce and other factors like politics, etc.

Priority scheduling can be either of the following:

Preemptive: This type of scheduling may preempt the central processing unit (CPU) in the case the priority of the freshly arrived process being greater than those of the existing processes.

Non-preemptive: This type of scheduling algorithm simply places the new process at the top of the ready queue.

In this question user has to enter number of processes and arrvival time,burst time,priority of each process

So whenever CPU preempts a process and shifts cpu shifts control to some another process of higher priority i.e Context Switching happens

## Algorithm:

1. Priority is assigned for each process.
2. Process with highest priority is executed first and so on.
3. Processes with same priority are executed in FCFS manner.
4. Priority can be decided based on memory requirements, time requirements or any other resource requirement.
5. Waiting time and Arrival time are calculated accordingly.

## Attributes used in program:

1. **Completion Time (C.T):** Time at which process completes its execution.

2. **Turn Around Time (T.A.T):** 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

## Test case 1:

**Process Priority ArrivalTime BurstTime WT TAT**

P0 1 7 5 0 5

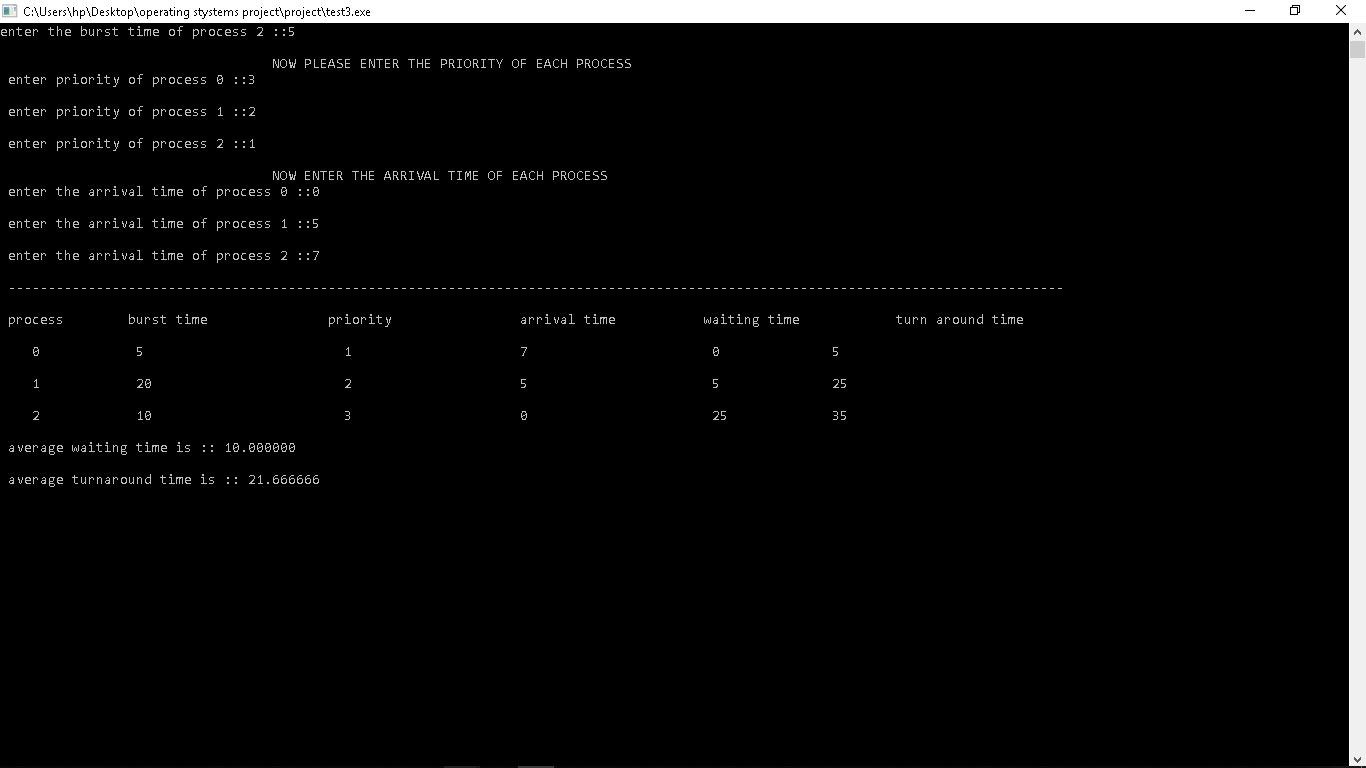
P1 2 5 20 5 25

P2 3 0 10 25 35

Average Turn Around Time :- 21.6

Average Waiting Time :- 10

Reference: -



Test case 2:

**Process Priority ArrivalTime BurstTime WT TAT**

P1 1 5 5 0 5

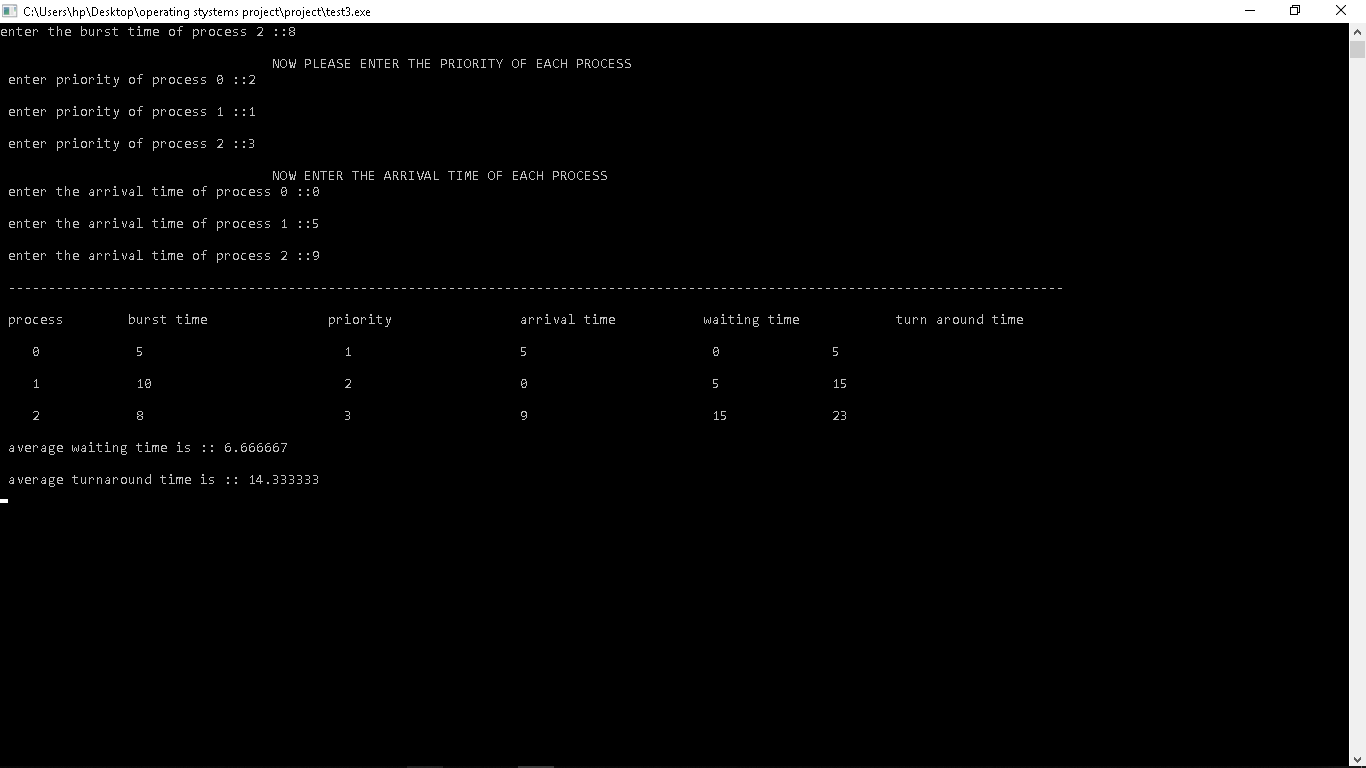
P0 2 0 10 5 15

P2 3 9 8 15 23

Average Turn Around Time :- 14.3

Average Waiting Time :- 6.66

Reference: -



**Constarints:**

**Whenever there is context switching in the process the priority of the process increases at the rate of two**

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

**{**

**wt[i]=0;**

**if(tem==1)**

**{**

**wt[i]+=2;**

**prio[i]=prio[i]\*2;**

**}**

* **Condition for priority scheduling:**

Here is the condition for priority scheduling

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

{pos=i;

for(j=i+1;j<n;j++) /\*here we are using position variable to check whether priority of the next process is

greater or smaller then the previous one\*/{

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

{pos=j;

tem==1;

//position is shifted to j//

}

}temp=prio[i];

prio[i]=prio[pos];

prio[pos]=temp;

temp=bt[i];

bt[i]=bt[pos];

bt[pos]=temp;

temp=at[i];

at[i]=at[pos];

at[pos]=temp;

}

**Boundary Condition: -**

Here the main boundary condition is whenever operating system pre-empts a process and shifts CPU shifts control to some another process of higher priority assume that it takes 2 seconds for context switching(dispatcher latency). In this period of time the other process will execute which has less arrival time as compare to previous holding process and it will go on with this boundary condition. If the time limit exceeds for a single process among all processes(i.e. 2 units will be as 3 or 4 units ) then the program or the output will become error or incorrect.