CSE316

Academic task-3

(Operating System)

Submitted by:-

Name :-

Registration No :-

Section :- K18GT

Roll No :-

Email Id :-

GitHub Link :-

Submitted to:-

Priyanka Mittal Mam,

**Researchers designed one system that classified interactive and noninteractive processes automatically by looking at the amount of terminal I/O. If a process did not input or output to the terminal in a 1-second interval, the process was classified as noninteractive and was moved to a lower-priority queue. In response to this policy, one programmer modified his programs to write an arbitrary character to the terminal at regular intervals of less than 1 second. The system gave his programs a high priority, even though the terminal output was completely meaningless**.

Solution:-

If we consider this question here it says that if we are going to give priority to the smallest process, definitely each and every programmer is going to do that he will manage all larger process into small. In this way when we are going to to give priority to the process of Non-interactive, then the programmer that means we will definitely going to use interactive. This means whenever we are going to shift from one environment to another definitely there is a chance of code will become change and a loads of difficulties which we should going to face.

The code which was below present describes that the given question here after compile it was going ask the programmer enter how many process you need then it will collect the data from us and give priority.

#include<stdio.h>

int main()

{

int i, type[20], n;

int resptime[20];

printf("Number of process: ");

scanf("%d", &n);

printf("Enter the data\n");

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

{

printf("Response time of p%d:", i);

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

if(resptime[i]<1000)

{

type[i]=1;

}

else

{

type[i]=0;

}

}

printf("process number\tresponse time\ttype\t\tpriority");

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

{

printf("\np%d\t\t%dms\t\t",i,resptime[i]);

if(type[i]==1)

{

printf("Interactive \tHigh");

}

else

{

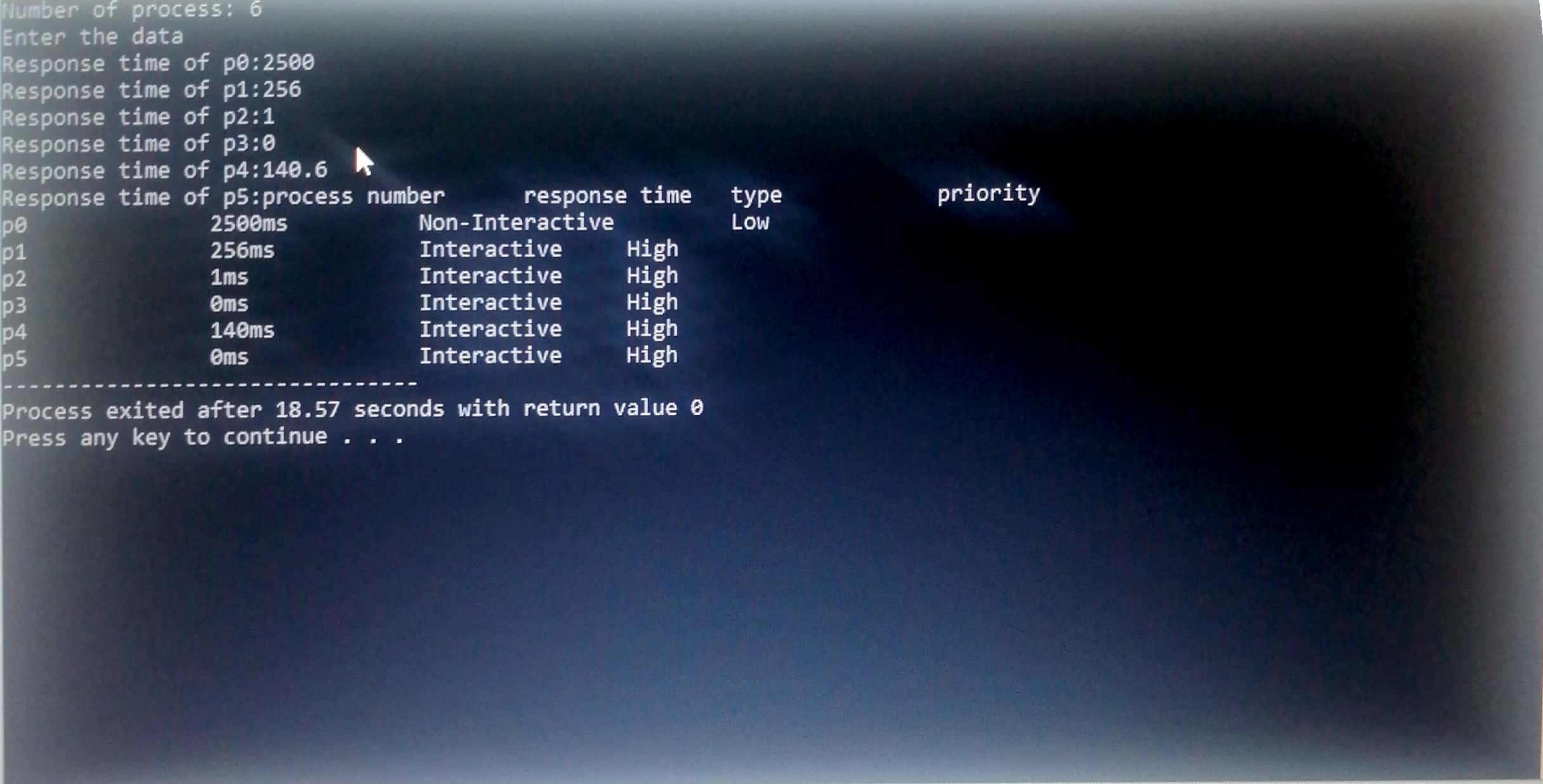
printf("Non-Interactive \tLow");

}

}

}

The **boundary conditions** of the code is here we are providing response time as 1000 if we are providing less than that it will automatically going to give high else providing more than that low. And the for loop is going to iterates n number of times.



**Q20. There are 3 student processes and 1 teacher process. Students are supposed to do their assignments and they need 3 things for that pen, paper and question paper. The teacher has an infinite supply of all the three things. One students has pen,an other has paper and another has question paper. The teacher places two things on a shared table and the student having the third complementary thing makes the assignment and tells the teacher on completion. The teacher then places another two things out of the three and again the student having the third thing makes the assignment and tells the teacher on completion. This cycle continues. WAP to synchronize the teacher and the students**.

Solution :-

Here we can discuss the two types of person one is student process and another is teacher process, we can explain this as in the form of concept of Operating System as, Suppose the two types of persons are able to enter into the library are one is student and second one is teacher. And the one who had entered into library searches the book he wants. After finding the book he will go the place where the counter presents and a single CPU process will present for issuing the books. As you know that queue is an FIFO so may be at the counter there may be arranged two types of queues in which one may belong to student and another belongs to teacher. In that both queue the persons of student and teacher are stand at the position of tail of the line. And the person whoever comes first will serve first for example, if teacher comes first and serviced and then student next……….and so on. Which means giving Priority having Non- Preemptive case. In this way a FIFO process is continued in an manner. For suppose if teacher first being served then student will may be next and after that another student may be next in this way they being served. And This type of process will may continue and may be there is chances of increase in waiting time.Here the programme in which I have written the waiting time for students and that time have been minimized.

The following code in which it also satisfies the given question in which support and here in which the programmer will going to receive a resources menu while the code compile successfully and that programmer has to select any two resources from that then the message will complete and end.

#include<stdio.h>

#include<stdbool.h>

struct requirement

{

bool pen;

bool paper;

bool question\_paper;

bool all\_three;

};

int main()

{

int n=3;

struct requirement s[n];

s[0].pen=true;

s[0].paper=false;

s[0].question\_paper=false;

s[0].all\_three=false;

s[1].pen=false;

s[1].paper=true;

s[1].question\_paper=true;

s[1].all\_three=false;

s[2].pen=false;

s[2].paper=false;

s[2].question\_paper=true;

s[2].all\_three=false;

while(s[0].all\_three==false||s[1].all\_three==false||s[2].all\_three==false)

{

int ch1,ch2;

printf("\nthe resources should be as follows:\n1.pen\n2.paper\n3.question\_paper\n enter the any two resources during which we have to be place on table:");

scanf("%d%d",&ch1,&ch2);

if(ch1==1 && ch2==2 && s[2].all\_three==false)

{

s[2].all\_three=true;

printf("Third student has completed the task\n");

}

if(ch1==2 && ch2==3 && s[0].all\_three==false)

{

s[0].all\_three=true;

printf("First student has completed the task\n");

}

if(ch1==1 && ch2==3 && s[1].all\_three==false)

{

s[1].all\_three=true;

printf("second student has completed the task\n");

}

}

printf("Therefore the students who have their tasks done succesfully\n");

return 0;

}