**Student Name: Sahil**

**Student ID: 11712767**

**Email Address: ss773sahil@gmail.com**

**GitHub Link:** <https://github.com/sahil7023/Os_project>

Ques. 5. CPU schedules N processes which arrive at different time intervals and each process is allocated the CPU for a specific user input time unit, processes are scheduled using a preemptive round robin scheduling algorithm. Each process must be assigned a numerical priority, with a higher number indicating a higher relative priority. In addition to the processes one task has priority 0. The length of a time quantum is T units, where T is the custom time considered as time quantum for processing. If a process is preempted by a higher-priority process, the preempted process is placed at the end of the queue. Design a scheduler so that the task with priority 0 does not starve for resources and gets the CPU at some time unit to execute. Also compute waiting time, turn around.

Ans:-

#include<stdio.h>

int main()

{

int count,j,n,time,remain,flag=0,time\_quantum;

int wait\_time=0,turnaround\_time=0,at[10],bt[10],rt[10];

printf("Enter Total Process:\t ");

scanf("%d",&n);

remain=n;

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

{

printf("Enter Arrival Time and Burst Time for Process Process Number %d :",count+1);

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

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

rt[count]=bt[count];

}

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

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

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

for(time=0,count=0;remain!=0;)

{

if(rt[count]<=time\_quantum && rt[count]>0)

{

time+=rt[count];

rt[count]=0;

flag=1;

}

else if(rt[count]>0)

{

rt[count]-=time\_quantum;

time+=time\_quantum;

}

if(rt[count]==0 && flag==1)

{

remain--;

printf("P[%d]\t|\t%d\t|\t%d\n",count+1,time-at[count],time-at[count]-bt[count]);

wait\_time+=time-at[count]-bt[count];

turnaround\_time+=time-at[count];

flag=0;

}

if(count==n-1)

count=0;

else if(at[count+1]<=time)

count++;

else

count=0;

}

printf("\nAverage Waiting Time= %f\n",wait\_time\*1.0/n);

printf("Avg Turnaround Time = %f",turnaround\_time\*1.0/n);

return 0;

}

Q32. Write a C program to solve the following problem:

Suppose that a disk drive has 5,000 cylinders, numbered 0 to 4999. The drive is currently serving a request at cylinder143, and the previous request was at cylinder 125. The queue of pending requests, in FIFO order is:

86, 1470, 913, 1774, 948, 1509, 1022, 1750, 130

Starting from the current head position, what is the total distance (in cylinders) that the disk arm moves to satisfy all the pending requests for each of the SCAN disk-scheduling algorithms?

Ans:-

#include<math.h>

#include<stdio.h>

int main()

{

int i,n,j=0,k=0,x=0,l,req[50],mov=0,cp,ub,end,pv, lower[50],upper[50], temp,a[50];

printf("Enter total number of cylinders \n");

scanf("%d",&ub);

printf("Enter the current position\n");

scanf("%d",&cp);

printf("Enter the previous position\n");

scanf("%d",&pv);

printf("Enter the number of elements of the queue\n");

scanf("%d",&n);

printf("Enter the request queue\n");

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

{

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

}

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

{

if(req[i]<cp)

{

lower[j]=req[i];

j++;

}

if(req[i]>cp)

{

upper[k]=req[i];

k++;

}

}

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

{

for(l=0;l<j-1;l++)

{

if(lower[l]<lower[l+1])

{

temp=lower[l];

lower[l]=lower[l+1];

lower[l+1]=temp;

}

}

}

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

{

for(l=0;l<k-1;l++)

{

if(upper[l]>upper[l+1])

{

temp=upper[l];

upper[l]=upper[l+1];

upper[l+1]=temp;

}

}

}

printf("Enter the end to which the head is moving (0 - for lower end(zero) and 1 - for upper end)\n");

scanf("%d",&end);

switch(end)

{

case 0:

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

{

a[x]=lower[i];

x++;

}

a[x]=0;

x++;

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

{

a[x]=upper[i];

x++;

}

break;

case 1:

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

{

a[x]=upper[i];

x++;

}

a[x]=ub;

x++;

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

{

a[x]=lower[i];

x++;

}

break;

}

printf("\nSCAN disk scheduled queue:\n");

mov=mov+abs(cp-a[0]);

printf("%d , %d",cp,a[0]);

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

{

mov=mov+abs(a[i]-a[i-1]);

printf(" , %d",a[i]);

}

printf("\n");

printf("total distance (cylinders) = %d\n",mov);

}