**KARNATAK LAW SOCIETY’S**

**GOGTE INSTITUTE OF TECHNOLOGY**

**UDYAMBAG, BELAGAVI – 590008**

**(An Autonomous Institution under Visvesvaraya Technological University, Belagavi)**

**(Approved By AICTE, New Delhi)**

**DEPARTMENT OF INFORMATION SCIENCE AND ENGINEERING**

 

**COURSE PROJECT**

**OPERATING SYSTEM**

RACHANA KAMPLI 2GI18IS032

LAXMI NYAMAGOUD 2GI18IS020

HEMANTH I T 2GI18IS015

ROHAN KOKATANUR 2GI18IS066

Guided by:

**Prof. S.G. Sanu**

1. **Title**: First Come First Serve Scheduler
2. **Problem Statement:**

To Calculate Turnaround time, waiting time, Average Turnaround time, and Average waiting time.

1. **Objectives:**

To simulate First Come First Served (FCFS) scheduling algorithm using C/C++ • To simulate Priority Scheduling algorithm using C/C++ 2. Procedures 1) Simulate First Come First Served (FCFS) Scheduling Algorithm a. In this part of the program, you will:

Prompt the user to enter the number of processes (limit to 5)

Prompt the user to enter the burst time for each process

Calculate waiting time for each process

Calculate turnaround time for each process

Calculate average waiting time and turnaround time for all the processes

Display all the information similar to the sample output window below 3) Simulate Priority Scheduling Algorithm a. In this part of the program, you will (just modify your last program):

Prompt the user to enter the number of processes (limit to 5)

Prompt the user to enter the burst time and priority number for each process (1 is the highest)

Sort the processes in ascending order according to their priorities.

Calculate waiting time for each process based on the priority list

Calculate turnaround time for each process based on the priority list

Calculate average waiting time and turnaround time for all the processes • Display all the information similar to the sample output window below You can simply use arrays for the processes, priority, process number, wait time and turnaround time in the program. You can also pointers and structure, etc. Please i need the code so i can run

1. **Implementation:**

#include <stdio.h>

#include <stdlib.h>

int total\_no\_process;

int process\_limit=19;

int main()

{

int i;

printf("\n");

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

printf("TIME CALCULATOR FOR FIRST COME FIRST SERVE (FCFS) SCHEDULER \n");

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

printf("\n");

printf("Enter The Total Number Of Processes : \n");

scanf("%d",&total\_no\_process);

while(total\_no\_process>process\_limit)

{

printf("Please Enter Number Of Processes Less Than %d :)\n",(process\_limit+1));

printf("Enter The Total Number Of Processes : \n");

scanf("%d",&total\_no\_process);

}

float burst\_time[total\_no\_process];

float arrival\_time[total\_no\_process];

float turnaround\_time[total\_no\_process];

float waiting\_time[total\_no\_process];

float avg\_turnaround\_time;

float avg\_waiting\_time;

for(i=1;i<(total\_no\_process+1);i++)

{

printf("enter the burst time for Process %d >>",i);

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

printf("enter the arrival time for Process %d >>",i);

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

printf("\n");

}

//sorting

arrival\_time[0]=0;

burst\_time[0]=0;

int j;

float temp,temp1;

for(i=1;i<(total\_no\_process+1);i++)

for(j=1;j<(total\_no\_process-i+1);j++)

if(arrival\_time[j]>arrival\_time[j+1])

{

temp=arrival\_time[j];

arrival\_time[j]=arrival\_time[j+1];

arrival\_time[j+1]=temp;

temp1=burst\_time[j];

burst\_time[j]=burst\_time[j+1];

burst\_time[j+1]=temp1;

}

//this is calculating waiting time

float sum\_burst\_time=0;

for(i=1;i<(total\_no\_process+1);i++)

{

if(i==1)

{

waiting\_time[i]=arrival\_time[i];

}

else

{

waiting\_time[i]=sum\_burst\_time-arrival\_time[i];

}

sum\_burst\_time =sum\_burst\_time+ burst\_time[i];

}

// this is for calculating turnaround time

for(i=1;i<(total\_no\_process+1);i++)

{

turnaround\_time[i]=waiting\_time[i]+burst\_time[i];

}

//avg waiting time

for(i=1;i<(total\_no\_process+1);i++)

{

avg\_waiting\_time=avg\_waiting\_time+waiting\_time[i];

}

avg\_waiting\_time=avg\_waiting\_time/total\_no\_process;

//avg waiting time

for(i=1;i<(total\_no\_process+1);i++)

{

avg\_turnaround\_time=avg\_turnaround\_time+turnaround\_time[i];

}

avg\_turnaround\_time=avg\_turnaround\_time/total\_no\_process;

printf("\n");

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

printf("Process No \t Arrival Time \t Burst Time \t Waiting Time \t Turnaround Time \n");

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

for(int j=1;j<(total\_no\_process+1);j++)

{

printf(" %d\t %0.3f\t %0.3f\t %0.3f\t %0.3f\n\n",j,arrival\_time[j],burst\_time[j],waiting\_time[j],turnaround\_time[j]);

}

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

printf("Average Waiting Time is %0.3f \n\n",avg\_waiting\_time);

printf("Average Turnaround Time is %0.3f \n",avg\_turnaround\_time);

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

}

1. **Working Model Of Final Solution**



