****

**Term Work**

**On**

**OPERATING SYSTEM**

**(PCS 506)**

**Submitted to: Submitted by:**

Dr. Pardeep Singh Prahlad Singh Aswal  
Associate Professor University Roll. No.: 2018550  
GEHU, D. Dun Class Roll No./Section: 39/A

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**GRAPHIC ERA HILL UNIVERSITY, DEHRADUN**

A picture containing logo

Description automatically generated

**DEPARTMENT OF CSE**

|  |
| --- |
| Photograph  Passport Size |

**STUDENT LAB REPORT SHEET**

**Name of Student ....................................................………Mob. No ...............................**

**Address Permanent .....................................................................................................**

**Father’s Name ................................. Occupation .................. Mob. No .......................**

**Mother’s Name ........................... Occupation ...................... Mob. No .......................**

**Section ............ Branch ............ Semester ............ Class Roll No ............ Grade A B C**

**Local Address .....................................................Email ............................................ Marks 5 3 1**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **S.No.** | **Practical** | **D.O.P.** | **Date of Submission** | **Grade (Viva)** | **Grade (Report File)** | **Total Marks (out of 10)** | **Student’s Signature** | **Teacher’s Signature** |
| **1** |  |  |  |  |  |  |  |  |
| **2** |  |  |  |  |  |  |  |  |
| **3** |  |  |  |  |  |  |  |  |
| **4** |  |  |  |  |  |  |  |  |
| **5** |  |  |  |  |  |  |  |  |
| **6** |  |  |  |  |  |  |  |  |
| **7** |  |  |  |  |  |  |  |  |
| **8** |  |  |  |  |  |  |  |  |
| **9** |  |  |  |  |  |  |  |  |
| **10** |  |  |  |  |  |  |  |  |
| **11** |  |  |  |  |  |  |  |  |
| **12** |  |  |  |  |  |  |  |  |

**PRACTICAL 5**

**Problem Statement:** Write a C program to implement first come first serve scheduling algorithm.

**FCFS**: FCFS is an operating system scheduling algorithm that automatically executes queued requests and processes in order of their arrival. It is the easiest and simplest CPU scheduling algorithm. In this type of algorithm, processes which requests the CPU first get the CPU allocation first. This is managed with a FIFO queue. The full form of FCFS is First Come First Serve.

**Source Code:**

#include <stdio.h>

#include <stdlib.h>

typedef struct FCFS\_scheduling

{

int pid, at, bt, ct, tat; // arrival,burst,completion,turnaround time

int wt, rt, st; // waiting,response,start time

} fcfs;

int comparator(const void \*num1, const void \*num2)

{

struct FCFS\_scheduling \*a = (struct FCFS\_scheduling \*)num1;

struct FCFS\_scheduling \*b = (struct FCFS\_scheduling \*)num2;

if (a->at >= b->at)

return 1;

else

return -1;

}

int main()

{

printf("\nName-Prahlad Singh Aswal\nSec - A\n");

printf("\nStudent ID - 20011854");

int n;

printf("\nEnter the total number of Processs: ");

scanf("%d", &n);

fcfs arr[n];

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

{

printf("Enter Arrival time and Burst time respectively for process P%d : ", i + 1);

arr[i].pid = i + 1;

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

}

qsort(arr, n, sizeof(fcfs), comparator);

float cycle\_length;

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

{

if (i == 0)

{

arr[i].ct = arr[i].at + arr[i].bt;

}

else if (arr[i].at >= arr[i - 1].ct)

{

arr[i].ct = arr[i].at + arr[i].bt;

}

else

arr[i].ct = arr[i - 1].ct + arr[i].bt;

}

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

{

if (i == 0)

{

arr[0].tat = arr[0].ct - arr[0].at;

arr[0].wt = arr[0].tat - arr[0].bt;

arr[0].rt = arr[0].wt;

}

else

{

arr[i].tat = arr[i].ct - arr[i].at;

arr[i].wt = arr[i].tat - arr[i].bt;

arr[i].rt = arr[i].wt;

}

}

float avg\_tat = arr[0].tat, avg\_wt = arr[0].wt;

float avg\_rt = arr[0].rt;

int idt = 0;

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

{

int x;

avg\_tat += arr[i].tat;

avg\_wt += arr[i].wt;

avg\_rt += arr[i].rt;

if (arr[i].at > arr[i - 1].ct)

{

x = arr[i].at - arr[i - 1].ct;

idt += x;

}

}

// printing table

printf(" PID\tAT\tBT\tCT\tTAT\tWT\tRT\n");

for (int j = 0; j < n; j++)

{

printf(" P%d\t%d\t%d\t%d\t%d\t%d\t%d\n", arr[j].pid, arr[j].at, arr[j].bt, arr[j].ct, arr[j].tat, arr[j].wt, arr[j].rt);

}

cycle\_length = arr[n - 1].ct - arr[0].at;

float cpu\_util = ((cycle\_length - idt) / cycle\_length) \* 100;

float tp = (float)n / (cycle\_length); // Throughput

printf("Average turn around time of all processes is: %f\n", avg\_tat / (float)n);

printf("Average waiting time of total processes is: %f\n", avg\_wt / (float)n);

printf("Average response time of total processes is: %f\n", avg\_rt / (float)n);

printf("CPU idle time is: %d\n", idt);

printf("CPU utilization(in percentage) is: %.2f \n", (float)cpu\_util);

printf("Throughput of all processes is: %f\n\n", tp);

return 0;

}

**Output**

