**OS Lab Assignment 9**

**Name: Siddhant Devsale**

**Class: SY CS-B**

**Roll No.: 13**

**PRN No.: 12311463**

**Q. Write a C Program to implement FCFS process scheduling algorithm.**

**Code :**

#include<stdio.h>

struct Process

{

int processID;

int arrivalTime;

int burstTime;

int waitingTime;

int turnaroundTime;

int completionTime;

int responseTime;

int startTime;

};

void sortProcesses(struct Process p[], int n)

{

struct Process temp;

for (int i = 0; i < n - 1; i++)

{

for (int j = 0; j < n - i - 1; j++)

{

if (p[j].arrivalTime > p[j + 1].arrivalTime)

{

temp = p[j];

p[j] = p[j + 1];

p[j + 1] = temp;

}

}

}

}

void printGanttChart(struct Process p[], int n)

{

printf("| ");

printf("P%d |", p[0].processID);

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

{

if(p[i-1].completionTime < p[i].arrivalTime)

{

printf(" Idle |");

}

printf(" P%d |", p[i].processID);

}

printf("\n");

printf("%d", p[0].arrivalTime);

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

{

if(p[i-1].completionTime < p[i].arrivalTime)

{

printf(" %d", p[i].arrivalTime);

}

else

{

printf(" %d", p[i].completionTime);

}

}

printf("\n");

}

int main()

{

int n;

printf("Enter the number of processes\n");

scanf("%d", &n);

struct Process p[n];

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

{

printf("Enter process ID, arrival time, and burst time for process %d\n", i + 1);

scanf("%d %d %d", &p[i].processID, &p[i].arrivalTime, &p[i].burstTime);

}

sortProcesses(p, n);

p[0].completionTime = p[0].arrivalTime + p[0].burstTime;

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

{

if (p[i].arrivalTime <= p[i - 1].completionTime)

{

p[i].completionTime = p[i - 1].completionTime + p[i].burstTime;

}

else

{

p[i].completionTime = p[i].arrivalTime + p[i].burstTime;

}

}

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

{

if (i == 0)

{

p[i].startTime = p[i].arrivalTime;

}

else

{

if (p[i - 1].completionTime <= p[i].arrivalTime)

{

p[i].startTime = p[i].arrivalTime;

}

else

{

p[i].startTime = p[i - 1].completionTime;

}

}

}

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

{

p[i].turnaroundTime = p[i].completionTime - p[i].arrivalTime;

p[i].waitingTime = p[i].turnaroundTime - p[i].burstTime;

p[i].responseTime = p[i].startTime - p[i].arrivalTime;

}

printf("FCFS algorithm:\n");

printf("Process ID\tArrival Time\tBurst Time\tCompletion Time\tTAT\tWT\tRT\n");

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

{

printf("%d\t\t%d\t\t%d\t\t%d\t\t%d\t%d\t%d\n",

p[i].processID,

p[i].arrivalTime,

p[i].burstTime,

p[i].completionTime,

p[i].turnaroundTime,

p[i].waitingTime,

p[i].responseTime);

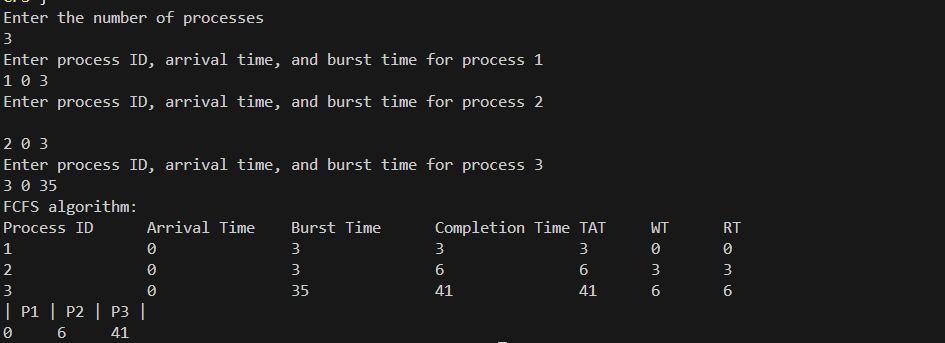
}

printGanttChart(p, n);

return 0;

}

**Output :**

****