**Experiment No 3**

**Aim -: To write a program to implement CPU & Scheduling algorithm for first.come first server**

#include<stdio.h>

// Define a structure to represent a process

struct process {

int pid; // Process ID

int burst; // Burst time

int arrival; // Arrival time

int waiting; // Waiting time

int turnaround; // Turnaround time

};

// Function to calculate waiting time and turnaround time for all processes

void calculate\_times(struct process p[], int n) {

int total\_waiting = 0;

int total\_turnaround = 0;

int completion\_time[n];

completion\_time[0] = p[0].burst + p[0].arrival;

p[0].turnaround = completion\_time[0] - p[0].arrival;

p[0].waiting = 0;

// Calculate completion time, waiting time, and turnaround time for each process

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

completion\_time[i] = p[i].burst + completion\_time[i-1];

p[i].turnaround = completion\_time[i] - p[i].arrival;

p[i].waiting = p[i-1].burst + p[i-1].arrival - p[i].arrival;

}

// Calculate total waiting time and total turnaround time

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

total\_waiting += p[i].waiting;

total\_turnaround += p[i].turnaround;

}

// Print the results

printf("Process ID\tBurst Time\tArrival Time\tWaiting Time\tTurnaround Time\n");

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

printf("%d\t\t%d\t\t%d\t\t%d\t\t%d\n", p[i].pid, p[i].burst, p[i].arrival, p[i].waiting, p[i].turnaround);

}

printf("Average waiting time: %f\n", (float)total\_waiting/n);

printf("Average turnaround time: %f\n", (float)total\_turnaround/n);

}

int main() {

int n; // Number of processes

printf("Enter the number of processes: ");

scanf("%d", &n);

// Initialize an array of processes

struct process p[n];

printf("Enter the burst time and arrival time for each process:\n");

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

printf("Process %d:\n", i+1);

p[i].pid = i+1;

scanf("%d %d", &p[i].burst, &p[i].arrival);

}

// Sort the processes in order of arrival time

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

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

if(p[i].arrival > p[j].arrival) {

struct process temp = p[i];

p[i] = p[j];

p[j] = temp;

}

}

}

// Calculate waiting time and turnaround time for each process

calculate\_times(p, n);

return 0;

}

**Output -:**

