Assignment No.5

Name: Bhavin Patil

Roll No. 66

Write a Program to implement FCFS Algorithm.

```
#include <bits/stdc++.h>

using namespace std;


struct process

{

   int PID;

   int AT;

   int BT;

   int CT;

   int TAT;

   int WT;

};


bool isValid(struct process *P, int n)

{

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

   {

       if (P[i].AT > P[i - 1].AT)
```

```
            continue;
        else
            return false;
    }
    return true;
}

void sortProcesses(struct process *P, int n)
{
    for (int i = 0; i < n; i++)
    {
        for (int j = i; j < n; j++)
        {
            if (P[i].AT > P[j].AT)
            {
                int a = P[j].AT;
                P[j].AT = P[i].AT;
                P[i].AT = a;

                int b = P[j].BT;
                P[j].BT = P[i].BT;
                P[i].BT = b;
```

```
                int d = P[j].PID;
                P[j].PID = P[i].PID;
                P[i].PID = d;
            }
        }
    }
}

int main()
{
    cout << "Enter Number of Process: ";
    int n;
    cin >> n;
    process P[n];

    for (int i = 0; i < n; i++)
    {
        cout << "\n\nEnter Process ID : ";
        cin >> P[i].PID;
        cout << "Enter Arrival Time for Process " << P[i].PID << " :
";
        cin >> P[i].AT;
        cout << "Enter Burst Time for Process " << P[i].PID << " : ";
        cin >> P[i].BT;
```

```
}


if (!isValid(P, n))
{
    sortProcesses(P, n);
}
int Ttime = 0;
for (int i = 0; i < n; i++)
{
    Ttime += P[i].BT;
    P[i].CT = Ttime;
}


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

    P[i].TAT = P[i].CT - P[i].AT;
}


for (int i = 0; i < n; i++)
{
    P[i].WT = P[i].TAT - P[i].BT;
}
```

```
   cout << "\n\n\nProcess ID\tAT\tBT\tCT\tTAT\tWT\n"
        << endl;
   cout <<
"=========================================================\n";
   for (int i = 0; i < n; i++)
   {
       cout << P[i].PID << "\t\t" << P[i].AT << "\t" << P[i].BT <<
"\t" << P[i].CT << "\t" << P[i].TAT << "\t" << P[i].WT << endl;
   }
   cout <<
"=========================================================";
   int maxCT, minAT;
   maxCT = P[0].CT;
   minAT = P[0].AT;
   for (int i = 1; i < n; i++)
   {
       maxCT = max(maxCT, P[i].CT);
       minAT = min(minAT, P[i].AT);
   }


   float AWT, tmp;
   for (int i = 0; i < n; i++)
   {
       tmp += P[i].WT;
```

```cpp
    }

    AWT = tmp / n;

    cout << "\nAverage Waiting Time: " << AWT << endl;

    cout << "\nNumber of Process: " << n << endl;

    float SL = maxCT - minAT;

    cout << "\nSchedule Length: " << SL << endl;

    float TP = n / SL;

    cout << "\nThroughtput: " << TP << endl;



    return 0;

}
```

```
bhavin@bhavin-Predator-PH315-53:~/Temp/os/assignment$ cd "/home/bhavin/Temp/os/assignment/" && g++ main.cpp -o main && "/home/bhavin/Temp/os/assignment/"main
Enter Number of Process: 4


Enter Process ID : 0
Enter Arrival Time for Process 0 : 0
Enter Burst Time for Process 0 : 5


Enter Process ID : 1
Enter Arrival Time for Process 1 : 1
Enter Burst Time for Process 1 : 3


Enter Process ID : 2
Enter Arrival Time for Process 2 : 2
Enter Burst Time for Process 2 : 8


Enter Process ID : 3
Enter Arrival Time for Process 3 : 3
Enter Burst Time for Process 3 : 6


Process ID      AT      BT      CT      TAT     WT

=====================================================
0               0       5       5       5       0
1               1       3       8       7       4
2               2       8       16      14      6
3               3       6       22      19      13
=====================================================
Average Waiting Time: 5.75

Number of Process: 4

Schedule Length: 22

Throughtput: 0.181818
bhavin@bhavin-Predator-PH315-53:~/Temp/os/assignment$
```