Operating System Lab

CEN-493

Program - 8

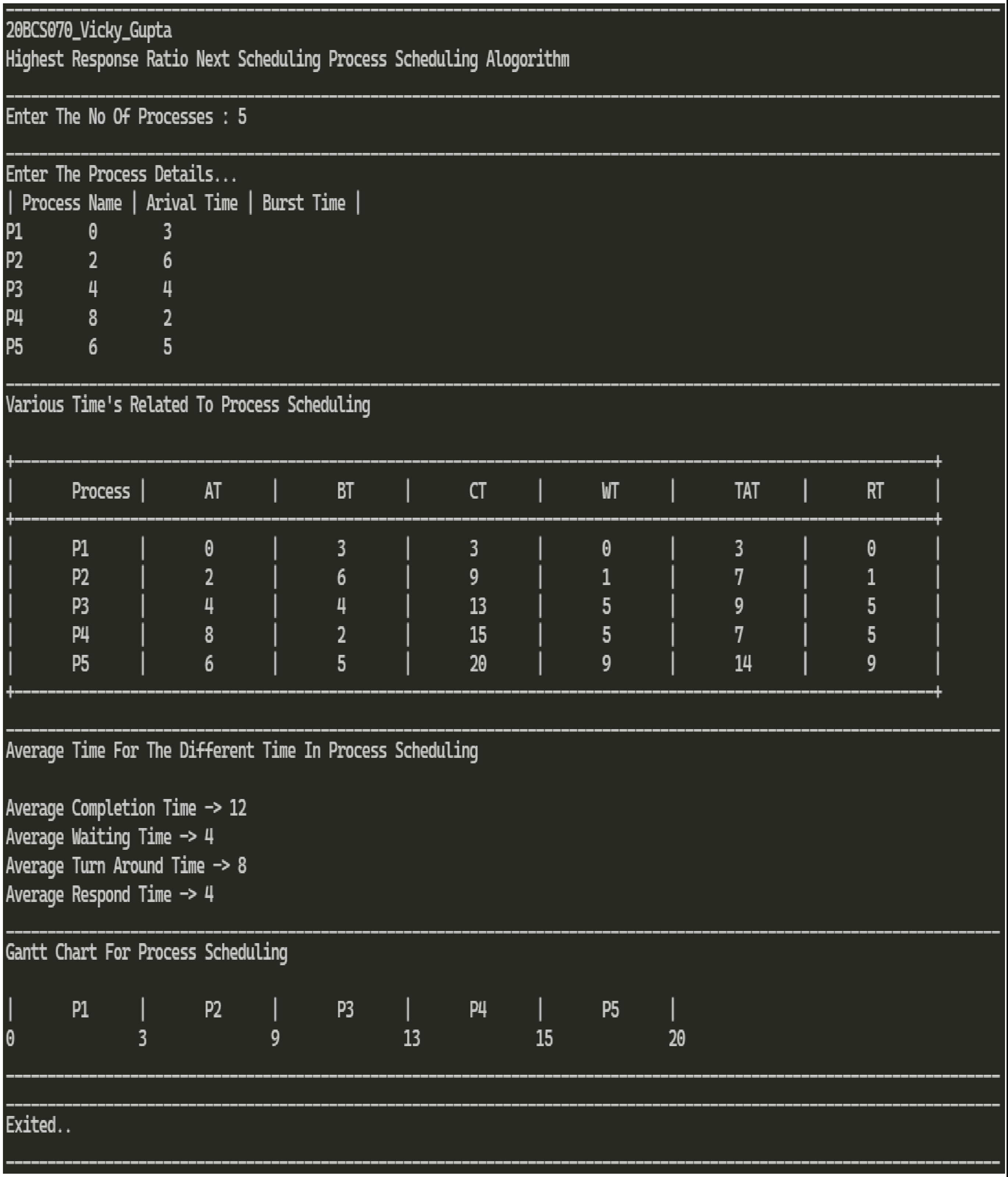
Code :-

```
#include <iostream>
#include <algorithm>
#include <vector>
#include <queue>
using namespace std;

struct Process
{
    string P_Name;
    int AT;
    int BT;
    int WT;
    int CT;
    int RT;
    int TAT;
};
int timer = 0;

bool mycomp(Process P1, Process P2)
```

```cpp
{

    if (timer < max(P1.AT, P2.AT) && P1.AT != P2.AT)
    {
        return P1.AT < P2.AT;
    }

    double rr1 = ((timer - P1.AT) + P1.BT) / (double)P1.BT;
    double rr2 = ((timer - P2.AT) + P2.BT) / (double)P2.BT;
    if (rr1 != rr2)
    {
        return rr1 > rr2;
    }
    int num1 = stoi(P1.P_Name.substr(1));
    int num2 = stoi(P2.P_Name.substr(1));
    return num1 < num2;
}

void Print_Bars()
{
    for (int i = 0; i < 120; i++)
        cout << "_";
    cout << "\n";
}

void Average_Time(Process P_Array[], int T_Process)
{
    double Av_CT = 0, Av_RT = 0, Av_WT = 0, Av_TAT = 0;
    for (int i = 0; i < T_Process; i++)
    {
        Av_CT += P_Array[i].CT;
        Av_RT += P_Array[i].RT;
        Av_TAT += P_Array[i].TAT;
        Av_WT += P_Array[i].WT;
    }
    Av_WT /= T_Process;
    Av_TAT /= T_Process;
    Av_RT /= T_Process;
    Av_CT /= T_Process;
    cout << "Average Time For The Different Time In Process 
Scheduling\n\n";

    cout << "Average Completion Time -> " << Av_CT << "\n";
    cout << "Average Waiting Time -> " << Av_WT << "\n";
    cout << "Average Turn Around Time -> " << Av_TAT << "\n";
    cout << "Average Respond Time -> " << Av_RT << "\n";
}

void GanttChart(vector<pair<string, pair<int, int>>>
&All_Interval)
{
    int size = All_Interval.size();
    cout << "Gantt Chart For Process Scheduling\n";
    cout << "\n";
    if (All_Interval[0].second.first != 0)
    {
        cout << "|\t\t|  ";
    }
    else
    {
        cout << "|\t";
    }

    for (int i = 0; i < size; i++)
    {
        if (i != 0 && All_Interval[i - 1].second.second <
All_Interval[i].second.first)
        {
            cout << "\t|\t";
        }
        cout << All_Interval[i].first << "\t|\t";
    }
    cout << "\n";

    if (All_Interval[0].second.first != 0)
    {
        cout << " 0\t";
        cout << All_Interval[0].second.first << "\t";
    }
    else
    {
        cout << All_Interval[0].second.first << "\t\t";
    }

    for (int i = 0; i < size; i++)
    {
        if (i != 0 && All_Interval[i - 1].second.second <
All_Interval[i].second.first)
        {
            cout << All_Interval[i].second.first << "\t\t";
        }
        cout << All_Interval[i].second.second << "\t\t";
    }
    cout << "\n";
}

void Chart(Process P_Array[], int T_Process)
{
    cout << "Various Time's Related To Process Scheduling\n\n";
    cout << "+---------------------------------------------------
-----------------------------------------------------------+\n";
    cout <<
"|\tProcess\t|\tAT\t|\tBT\t|\tCT\t|\tWT\t|\tTAT\t|\tRT      |\n";
    cout << "+---------------------------------------------------
-----------------------------------------------------------+\n";
    for (int i = 0; i < T_Process; i++)
    {
        cout << "|\t" << P_Array[i].P_Name
             << "\t|\t" << P_Array[i].AT
             << "\t|\t" << P_Array[i].BT
             << "\t|\t" << P_Array[i].CT
             << "\t|\t" << P_Array[i].WT
             << "\t|\t" << P_Array[i].TAT
             << "\t|\t" << P_Array[i].RT << "\t|\n";
    }
    cout << "+---------------------------------------------------
-----------------------------------------------------------+\n";
}

void Timing(vector<pair<string, pair<int, int>>> &All_Interval,
Process P_Array[], int T_Process)
{
    int size = All_Interval.size();
    for (int i = 0; i < T_Process; i++)
    {
        for (int j = size - 1; j >= 0; j--)
        {
            if (P_Array[i].P_Name == All_Interval[j].first)
            {
                P_Array[i].CT = All_Interval[j].second.second;
                break;
            }
        }
        P_Array[i].TAT = P_Array[i].CT - P_Array[i].AT;
        P_Array[i].WT = P_Array[i].TAT - P_Array[i].BT;
        for (int j = 0; j < size; j++)
        {
            if (P_Array[i].P_Name == All_Interval[j].first)
            {
                P_Array[i].RT = All_Interval[j].second.first -
P_Array[i].AT;
                break;
            }
        }
    }
    Print_Bars();
    Chart(P_Array, T_Process);
    Print_Bars();
    Average_Time(P_Array, T_Process);
    Print_Bars();
    GanttChart(All_Interval);
    Print_Bars();
}

vector<pair<string, pair<int, int>>> Time_Intervals(vector<string>
&timeArray)
{
    vector<pair<string, pair<int, int>>> processTimeInterval;
    for (int i = 0; i < timeArray.size(); i++)
    {
        int end = timeArray.size();
        for (int j = i + 1; j < timeArray.size(); j++)
        {
            if (timeArray[i] != timeArray[j])
            {
                end = j;
                break;
            }
        }
        processTimeInterval.push_back({timeArray[i], {i, end}});
        i = end - 1;
    }
    return processTimeInterval;
}

void AddTimeToArray(Process process, vector<string> &timeArray,
int timer, int BT)
{
    for (int i = timer; i < timer + BT; i++)
    {
        timeArray.push_back(process.P_Name);
    }
}

void HRRN(Process P_Array[], int T_Process)
{
    vector<Process> New_P_Array(P_Array, P_Array + T_Process);
    sort(New_P_Array.begin(), New_P_Array.end(), mycomp);
    vector<string> timeArray;
    timer = New_P_Array[0].AT;
    if (timer != 0)
    {
        Process pnull;
        pnull.P_Name = "--";
        AddTimeToArray(pnull, timeArray, 0, timer);
    }
    while (!New_P_Array.empty())
    {

        Process processCpuAllocated = New_P_Array[0];
        New_P_Array.erase(New_P_Array.begin());
        while (timer < processCpuAllocated.AT)
        {
            timeArray.push_back("--");
            timer++;
        }
        AddTimeToArray(processCpuAllocated, timeArray, timer,
processCpuAllocated.BT);
        timer += processCpuAllocated.BT;
        sort(New_P_Array.begin(), New_P_Array.end(), mycomp);
    }
    vector<pair<string, pair<int, int>>> Intervals =
Time_Intervals(timeArray);
    Timing(Intervals, P_Array, T_Process);
}

int main()
{
    // system("cls");
    Print_Bars();
    cout << "20BCS070_Vicky_Gupta\n";
    cout << "Highest Response Ratio Next Scheduling Process
Scheduling Alogorithm\n";
    Print_Bars();
    int T_Process;
```

```cpp
    cout << "Enter The No Of Processes : ";
    cin >> T_Process;
    fflush(stdin);
    Process P_Array[T_Process];
    Print_Bars();
    cout << "Enter The Process Details...\n";
    cout << "| Process Name | Arival Time | Burst Time |\n";

    for (int i = 0; i < T_Process; i++)
    {
        cin >> P_Array[i].P_Name;
        cin >> P_Array[i].AT;
        cin >> P_Array[i].BT;
    }

    HRRN(P_Array, T_Process);
    Print_Bars();
    cout << "Exited..\n";
    Print_Bars();
    return 0;
}
```

Output :-



```
--------------------------------------------------------------------------------------------------------
20BCS070_Vicky_Gupta
Highest Response Ratio Next Scheduling Process Scheduling Alogorithm

--------------------------------------------------------------------------------------------------------
Enter The No Of Processes : 5

--------------------------------------------------------------------------------------------------------
Enter The Process Details...
| Process Name | Arival Time | Burst Time |
P1       0       3
P2       2       6
P3       4       4
P4       8       2
P5       6       5

--------------------------------------------------------------------------------------------------------
Various Time's Related To Process Scheduling


+-------------------------------------------------------------------------------------------------+
|       Process |       AT      |       BT      |       CT      |       WT      |       TAT     |       RT      |
+-------------------------------------------------------------------------------------------------+
|       P1      |       0       |       3       |       3       |       0       |       3       |       0       |
|       P2      |       2       |       6       |       9       |       1       |       7       |       1       |
|       P3      |       4       |       4       |       13      |       5       |       9       |       5       |
|       P4      |       8       |       2       |       15      |       5       |       7       |       5       |
|       P5      |       6       |       5       |       20      |       9       |       14      |       9       |
+-------------------------------------------------------------------------------------------------+

--------------------------------------------------------------------------------------------------------
Average Time For The Different Time In Process Scheduling

Average Completion Time -> 12
Average Waiting Time -> 4
Average Turn Around Time -> 8
Average Respond Time -> 4

--------------------------------------------------------------------------------------------------------
Gantt Chart For Process Scheduling

|       P1      |       P2      |       P3      |       P4      |       P5      |
0               3               9               13              15              20

--------------------------------------------------------------------------------------------------------

--------------------------------------------------------------------------------------------------------
Exited..

--------------------------------------------------------------------------------------------------------
```