Operating System Lab

CEN-493

Program - 7

Code :-

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

struct Process
{
    string P_Name;
    int AT;
    int BT;
    int PT;
    int WT;
    int CT;
    int RT;
    int TAT;
};

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

```cpp
{
    if (P1.AT != P2.AT)
    {
        return P1.AT < P2.AT;
    }
    else if (P1.PT != P2.PT)
    {
        return P1.PT < P2.PT;
    }
    else
    {
        int num1 = stoi(P1.P_Name.substr(1));
        int num2 = stoi(P2.P_Name.substr(1));
        return num1 < num2;
    }
}

struct myCompPT
{
    bool operator()(Process &p1, Process const &p2)
    {
        if (p1.PT != p2.PT)
            return p1.PT > p2.PT;
        else
        {
            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 < 130; 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|\tPT\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].PT
             << "\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 Preemptive_Priority_Scheduling(Process P_Array[], int
    T_Process)
    {
        sort(P_Array, P_Array + T_Process, mycomp);
        priority_queue<Process, vector<Process>, myCompPT> pque;
        int processIterator = 0;
        vector<string> timeArray;
        pque.push(P_Array[0]);
        int timer = P_Array[processIterator].AT;
        if (timer != 0)
        {
            Process pnull;
            pnull.P_Name = "--";
            AddTimeToArray(pnull, timeArray, 0, timer);
        }
        processIterator++;
        while (!pque.empty() || processIterator < T_Process)
        {
            if (!pque.empty())
            {
```

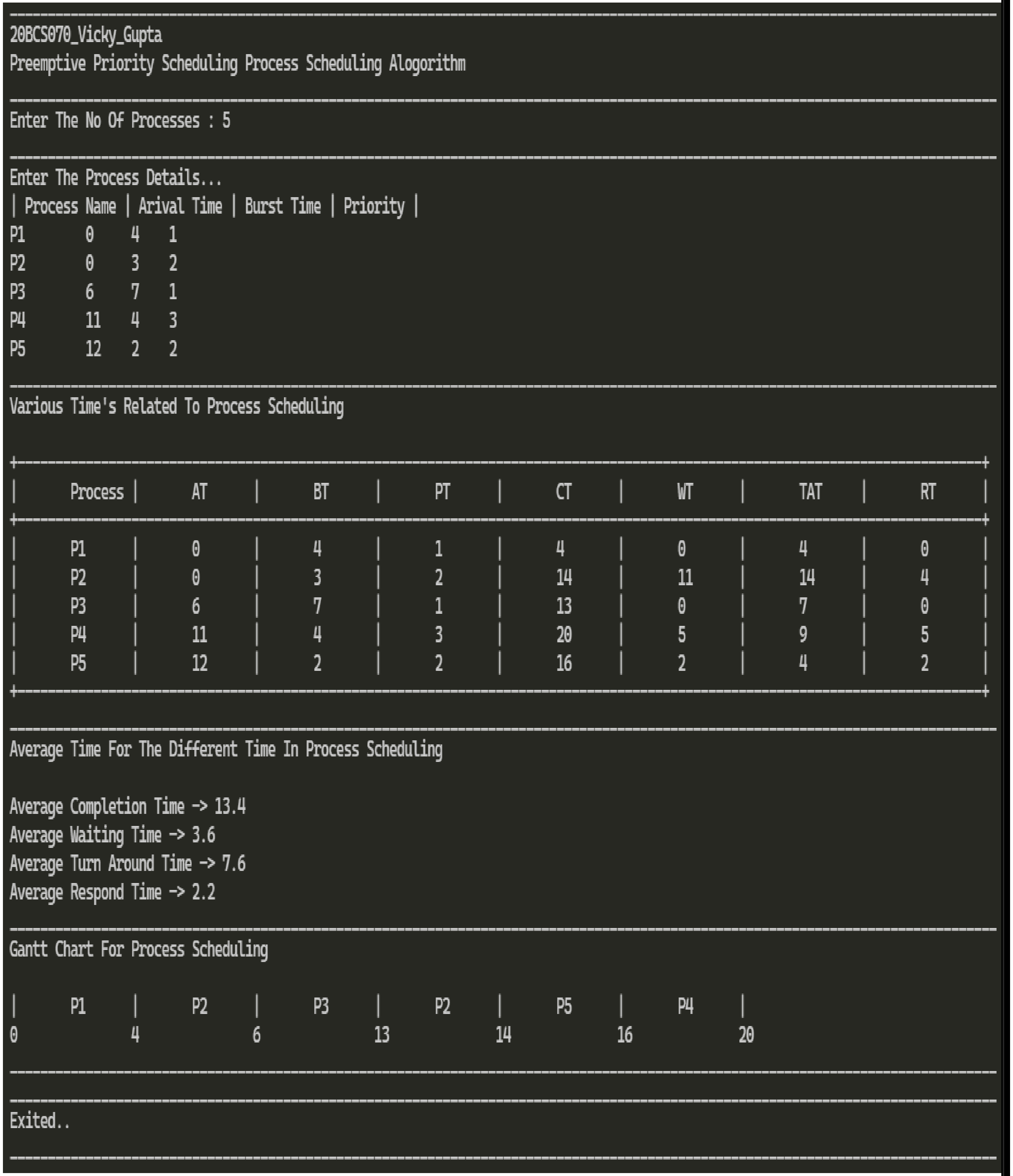
```
            Process processCpuAllocated = pque.top();
            pque.pop();
            AddTimeToArray(processCpuAllocated, timeArray, timer,
1);
            timer += 1;
            processCpuAllocated.BT--;
            if (processCpuAllocated.BT != 0)
            {
                pque.push(processCpuAllocated);
            }
        }
        else
        {
            timeArray.push_back("--");
            timer++;
        }
        while (processIterator < T_Process && timer >=
P_Array[processIterator].AT)
        {
            pque.push(P_Array[processIterator++]);
        }
    }
    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 << "Preemptive Priority Scheduling Process Scheduling
Alogorithm\n";
    Print_Bars();
    int T_Process;
    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 | Priority
|\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;
        cin >> P_Array[i].PT;
    }

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

Output :-



```
------------------------------------------------------------------------------------------------------------------------
20BCS070_Vicky_Gupta
Preemptive Priority Scheduling Process Scheduling Alogorithm

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

------------------------------------------------------------------------------------------------------------------------
Enter The Process Details...
| Process Name | Arival Time | Burst Time | Priority |
P1       0    4   1
P2       0    3   2
P3       6    7   1
P4       11   4   3
P5       12   2   2

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


+----------------------------------------------------------------------------------------------------------------------+
|      Process |      AT      |      BT      |      PT      |      CT      |      WT      |      TAT     |      RT      |
+----------------------------------------------------------------------------------------------------------------------+
|      P1      |      0       |      4       |      1       |      4       |      0       |      4       |      0       |
|      P2      |      0       |      3       |      2       |      14      |      11      |      14      |      4       |
|      P3      |      6       |      7       |      1       |      13      |      0       |      7       |      0       |
|      P4      |      11      |      4       |      3       |      20      |      5       |      9       |      5       |
|      P5      |      12      |      2       |      2       |      16      |      2       |      4       |      2       |
+----------------------------------------------------------------------------------------------------------------------+

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

Average Completion Time -> 13.4
Average Waiting Time -> 3.6
Average Turn Around Time -> 7.6
Average Respond Time -> 2.2

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

|      P1      |      P2      |      P3      |      P2      |      P5      |      P4      |
0              4              6              13             14             16             20

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

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