\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Operating System Lab

CEN-493

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Program - 3

Code :-

#include <iostream>

#include <algorithm>

using namespace std;

struct Process

{

    string P\_Name;

    int AT;

    int BT;

    int WT;

    int CT;

    int RT;

    int TAT;

};

bool mycomp(Process P1, Process P2)

{

    if (P1.AT != P2.AT)

    {

        return P1.AT < P2.AT;

    }

    else if (P1.BT != P2.BT)

    {

        return P1.BT < P2.BT;

    }

    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 < 100; 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(Process P\_Array[], int T\_Process)

{

    cout << "Gantt Chart For Process Scheduling\n";

    cout << "\n";

    if (P\_Array[0].AT != 0)

    {

        cout << "|     |  ";

    }

    else

    {

        cout << "|  ";

    }

    for (int i = 0; i < T\_Process; i++)

    {

        if (i != 0 && P\_Array[i - 1].CT < P\_Array[i].AT)

        {

            cout << "    |   ";

        }

        cout << P\_Array[i].P\_Name << "  |   ";

    }

    cout << "\n";

    if (P\_Array[0].AT != 0)

    {

        cout << " 0     ";

        cout << P\_Array[0].AT << "     ";

    }

    else

    {

        cout << P\_Array[0].AT << "      ";

    }

    for (int i = 0; i < T\_Process; i++)

    {

        if (i != 0 && P\_Array[i - 1].CT < P\_Array[i].AT)

        {

            cout << P\_Array[i].AT << "      ";

        }

        cout << P\_Array[i].CT << "      ";

    }

    cout << "\n";

}

void Chart(Process P\_Array[], int T\_Process)

{

    cout << "Various Time's Related To Process Scheduling\n\n";

    cout << "|  Process   |  AT  |  BT  |   CT   |   WT  |  TAT  |  RT  |\n";

    for (int i = 0; i < T\_Process; i++)

    {

        cout << "   " << P\_Array[i].P\_Name << "\t\t" << P\_Array[i].AT

             << "\t" << P\_Array[i].BT << "\t" << P\_Array[i].CT

             << "\t" << P\_Array[i].WT << "\t" << P\_Array[i].TAT

             << "\t" << P\_Array[i].RT << "\n";

    }

}

void New\_Process\_Array(Process P\_Array[], Process N\_P\_Array[], int T\_Process)

{

    sort(P\_Array, P\_Array + T\_Process, mycomp);

    bool isProcessed[T\_Process] = {0};

    int Timer = P\_Array[0].AT;

    for (int i = 0; i < T\_Process; i++)

    {

        int p\_no = -1;

        for (int j = 0; j < T\_Process; j++)

        {

            if (Timer >= P\_Array[j].AT && isProcessed[j] == 0)

            {

                if (p\_no == -1)

                {

                    p\_no = j;

                }

                if (p\_no != -1 && P\_Array[p\_no].BT > P\_Array[j].BT)

                {

                    p\_no = j;

                }

            }

        }

        if (p\_no == -1) // when the process has gaps

        {

            for (int j = 0; j < T\_Process; j++)

            {

                if (isProcessed[j] == 0)

                {

                    p\_no = j;

                    break;

                }

            }

        }

        isProcessed[p\_no] = 1;

        N\_P\_Array[i] = P\_Array[p\_no];

        if (Timer < P\_Array[p\_no].AT)

        {

            Timer += (P\_Array[p\_no].AT - Timer);

        }

        Timer += P\_Array[p\_no].BT;

    }

}

void SJF(Process P\_Array[], int T\_Process)

{

    Process N\_P\_Array[T\_Process];

    New\_Process\_Array(P\_Array, N\_P\_Array, T\_Process);

    int Timer = 0;

    for (int i = 0; i < T\_Process; i++)

    {

        if (Timer < N\_P\_Array[i].AT)

        {

            Timer += (N\_P\_Array[i].AT - Timer);

        }

        Timer += N\_P\_Array[i].BT;

        N\_P\_Array[i].CT = Timer;

        N\_P\_Array[i].TAT = N\_P\_Array[i].CT - N\_P\_Array[i].AT;

        N\_P\_Array[i].WT = N\_P\_Array[i].TAT - N\_P\_Array[i].BT;

        N\_P\_Array[i].RT = N\_P\_Array[i].WT;

    }

    Print\_Bars();

    Chart(N\_P\_Array, T\_Process);

    Print\_Bars();

    Print\_Bars();

    GanttChart(N\_P\_Array, T\_Process);

    Print\_Bars();

    Print\_Bars();

    Average\_Time(N\_P\_Array, T\_Process);

    Print\_Bars();

}

int main()

{

    // system("cls");

    Print\_Bars();

    cout << "20BCS070\_Vicky\_Gupta\n";

    cout << "Shortest Job First 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 | \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;

    }

    SJF(P\_Array, T\_Process);

    Print\_Bars();

    cout << "Exited..\n";

    Print\_Bars();

    return 0;

}

Output :-

Text

Description automatically generated