Lab Report

Course Code: CSE 332

Course title: Operating System And System Programming Lab

Signature:

Date:26/5/24



**Submitted By:**

Name:Raihan Rimon

ID:213311004

Semester:7th.

Section:A.

Department of CSE

Varendra University Rajshahi.

**I**

Submitted To:

**Ayesha Akter Lima**

**Lecturer**

Department of CSE

**Varendra University Rajshahi.**

**Sumaia Rahman**

**Lecturer**

Department of CSE

**Varendra University Rajshahi**

1. There are three processes. It follows the non-preemptive approach i.e. once a process has control over the CPU it will not preempt until it terminates.

The criteria for the selection of processes are arrival time. The dispatcher selects the first job in the ready queue and this job runs to completion of its CPU burst.

Not optimal for interactive systems.

Now calculate the average the waiting time for the given data.

|  |  |  |
| --- | --- | --- |
| P | Burst time | Arrival time |
| P1 | 16 | 0 |
| P2 | 5 | 1 |
| P3 | 4 | 2 |

Also plot the Gantt chart.

Answer:

#include <iostream>

using namespace std;

int main(){

    cout <<"\n\n";

    string process[3]   = {"p1", "p2", "p3"};

    int burst\_time[3]   = {  16,    5,    4};

    int arrival\_time[3] = {   0,    1,    2};

    int process\_number  = 3;

    int current\_time = 0;

    int total\_waiting\_time = 0;

    for (int i = 0; i < process\_number; i++){

        cout << ">> For the Process : " <<process[i] <<"\n";

        cout << "Starting Time      : " << current\_time <<"\n";

        cout << "Arrival Time of " <<process[i] << " : " << arrival\_time[i] <<"\n";

        int current\_waiting\_time = current\_time - arrival\_time[i];

        cout << "------------------(-)--------\n";

        cout << "Waiting Time of " <<process[i] << " : " << current\_waiting\_time <<"\n";

        total\_waiting\_time += current\_waiting\_time;

        current\_time += burst\_time[i];

        cout << "  Burst Time of " <<process[i] << " : " << burst\_time[i] <<"\n";

        cout <<"\n";

    }

    float average\_waiting\_time = total\_waiting\_time / process\_number;

    cout << "Total Waiting Time   = " << total\_waiting\_time <<"\n";

    cout << "Total Process Count  = " << process\_number<<"\n";

    cout << "--------------------(/)------\n";

    cout << "Average Waiting Time = " << average\_waiting\_time <<"\n\n\n";;

    cout << " The Gantt Chart >> \n";

    cout << " +";

    for (int i = 0; i < process\_number; i++){

        for (int j = 0; j < burst\_time[i]; j++){

            cout << "-";

        }

        cout << "+";

    } cout <<"\n";

    cout << " |";

    for (int i = 0; i < process\_number; i++){

        for (int j = 0; j < burst\_time[i]; j++){

            cout << " ";

        }

        cout << "|";

    } cout <<"\n";

    cout << " +";

    for (int i = 0; i < process\_number; i++){

        for (int j = 0; j < burst\_time[i]; j++){

            cout << "-";

        }

        cout << "+";

    } cout <<"\n";

    cout << " ";

    for (int i = 0; i < process\_number; i++){

        cout << process[i];

        for (int j = 0; j < burst\_time[i]-1; j++){

            cout << " ";

        }

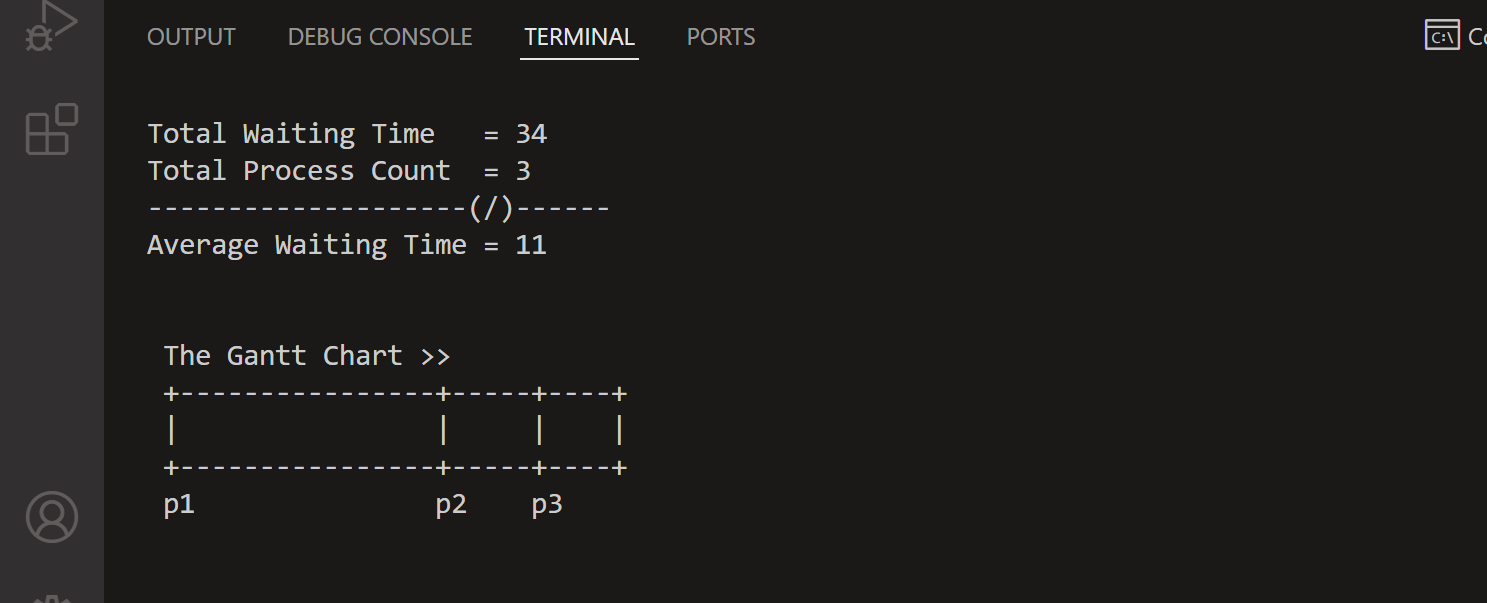
    }  cout <<"\n";

    cout <<"\n\n";

    return 0;

}

Output:



2.Create a new process P1. Now make a clone of the process P1 named N1. Consider N1 as a base process now create another new clone process named S1. Inside S1 write your university name,

semester and session. Now release the memory of session and allocate the memory for your ID.

Answer:

