

**Algorithm**

LN, FN, MI

|  |
| --- |
| *Submitted to:*  **<\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_>**  Professor  Date of submission |

**(CC0007)**

**<Machine Problem 3>**

*Submitted by:*

1X1

PICTURE

1. **PROGRAM OUTCOME/S (PO) ADDRESSED BY THE LABORATORY EXERCISE**

* Apply knowledge of computing appropriate to the discipline [PO: A]

1. **COURSE LEARNING OUTCOME/S (CLO) ADDRESSED BY THE LABORATORY EXERCISE**

* To understand and apply knowledge of computing concepts and fundamentals. [CLO: 1]

1. **INTENDED LEARNING OUTCOME/S OF THE LABORATORY EXERCISE**

At the end of this exercise, students must be able to:

* Solve the given problem
* Create documentation base on the given problem
* Create test cases for the problem

1. **Problem definition**

Applying the brute force algorithm, design a C++ program that implements the following.

1. CPU Scheduling Algorithm
   1. **SET A - 1,3,5,7,9 (GROUP) Non-PREEMPTIVE**
      1. first come first serve.
      2. short job first
      3. priority scheduling
   2. **SET B - 2,4,6,8,10 (GROUP) PREEMPTIVE**
      1. Shortest remaining time first
      2. Priority Scheduling
      3. Round Robin

The program must use an array and recursion. You are not allowed to use built-in C++ functions but rather create a user-defined function.

**Requirements.**

**Algorithm**

For each of the CPU scheduling algorithms, define the logic in a step-by-step manner. Describe how you will implement the scheduling, considering that you must use arrays and recursion.

**Pseudocode**

Translate the algorithms into pseudocode. This should be a high-level representation of how your program will function, written in a way that closely resembles C++ syntax without being actual code.

**Flowchart**

Creating a flowchart for a CPU Scheduling program involves visually mapping out the logic and the flow of the program.

**Program in C++**

Develop the actual C++ program based on your pseudocode. Create user-defined functions for each scheduling algorithm. Ensure that you do not use built-in C++ functions that are restricted and utilize recursion where possible.

**Program Simulation**

Simulate the execution of your program with test cases. You can use an array of structures or classes to hold process information and implement the scheduling logic.

**Program Discussion**

Discuss how your program implements each algorithm and any C++ language features you used. Explain the flow of the program and how it meets the requirements. Include File-handling capability to store and retrieve the data. Include program menu for the different operations.

**Program Output**

Showcase the output of your program for each algorithm. The output should include the order of execution, waiting times, turnaround times, and any other relevant information.

**Video Demonstration**

Creating a video demonstration involves several steps to ensure that your content is clear, informative, and engaging.

**Difficulties and Learnings**

Discuss any challenges you faced during the development and how you overcame them. Mention any new concepts learned during the process.

**References**

List any resources you used to learn about CPU scheduling algorithms, C++ programming, or recursion.

1. **ASSESSMENT**

|  |  |
| --- | --- |
| Department | Computer Science |
| Subject Code | CS0007 |
| Description | Algorithm |
| Term/Academic Year |  |

**Note: The following rubrics/metrics will be used to grade students’ output in the lab exercise**

# Rubrics

**The following rubrics/metrics will be used to grade students’ output in the lab exercise.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Program (50 pts)** | **(Excellent)** | **(Good)** | **(Fair)** | **(Poor)** |
| Program execution and correct simulation(20pts) | Program executes correctly with no syntax or runtime errors (18 – 20) | Program executes with less than 3 errors (15 – 17 ) | Program executes with more than 3 errors (12 – 14) | Program does not execute (10 – 11) |
| Correct output  (20pts) | Program displays correct output with no errors (18 – 20) | Output has minor errors (15 – 17) | Output has multiple errors (12 – 14 ) | Output is incorrect (10 - 11) |
| Design of output  (10pts) | Program displays more than expected (10) | Program displays minimally expected output (8 – 9) | Program does not display the required output (6-7) | Output is poorly designed (5) |
| Design of logic and Algorithm  (20pts) | Program is logically well designed (18 – 20) | Program has slight logic errors that do no significantly affect the results (15 – 17) | Program has significant logic errors (12 – 14 ) | Program is incorrect (10 - 11) |
| Standards  (20pts) | Program is stylistically well designed (18 – 20) | Few inappropriate design choices (i.e. poor variable names, improper indentation) (15 – 17) | Several inappropriate design choices (i.e. poor variable names, improper indentation) (12 – 14 ) | Program is poorly written (10 - 11) |
| Delivery  (10pts) | The program was delivered on time. (10) | The program was delivered within a week of the due date. (8 – 9) | The code was within 2 weeks of the due date. (6 – 7) | The code was more than 2 weeks overdue (5) |

1. **REFERENCES**

* Mueller(2018) Python for Data Science for dummies, Wiley
* Tuner (2018) Statistics For Machine Learning: Techniques For Exploring Supervised, Unsupervised, And Reinforcement Learning Models With Python And R
* Tuner(2018) Python Programming: 3 Books In 1: Beginner’s Guide + Intermediate Guide + Expert Guide To Learn Python Step-By-Step, Wiley
* Romano(2018) Learn Python Programming: The No-Nonsense, Beginner's Guide To Programming, Data Science, And Web Development With Python 3.7,Packt Publishing
* Morgan (2017), Data Analysis From Scratch With Python: Step-By-Step Guide , Createspace Independent Publishing Platform.
* Barry(2017), Head First Python: A Brain-Friendly Guide , O'reilly Media
* Guttag(2016), Introduction To Computation And Programming Using Python (2016), The Mit Press

Courseware Materials available at www.feu.instructure.com