CS 3305A: Operating Systems
Department of Computer Science
Western University
Assignment 4
Fall 2023
Due Date: November 6th 2023

Purpose:
The main goal of this assignment is to gain hands-on experience with CPU scheduling algorithms using C language in a Linux environment.

Performance Evaluation of CPU Scheduling Algorithms
You will develop CPU Scheduling Algorithms using the C programming language in Linux. A sample input file is provided with the assignment, which must be used to develop the **Shortest Job First (SJF)** CPU Scheduling Algorithm.

Format of the Input File:
You must use the same input file name (i.e., *sjf_input.txt)* in your code. The input file contains several test cases for this assignment. Every line of the input file represents one individual test case. For example, if there are four lines inside the input file, it means your program must execute four different test cases. Every input line consists of several processes and their corresponding information, such as *process name, arrival time (art_1), and burst time (brt_1)* which follows the format below:

Input format: process_1 art_1 brt_1 process_2 art_2 brt_2 process_n art_n brt_n
Input example: p1 0 23 p2 1 3 p3 2 3

In the example input given above, there are three processes such as p1, p2, and p3. The arrival and burst time of process p1 is 0 and 23, respectively. The arrival and burst time for process p2 is 1 and 3, respectively. The arrival and burst time of process p3 is 2 and 3, respectively. The individual entries in the input line are separated by space.

What you need to do:
You must supply the given *sjf_input.txt* file to your program. First, you need to parse the input file line-wise where every line in the input file is a test case. For every test case in the input file, apply Shortest Job First (SJF) Scheduling and output the process schedule details. The output of your program must follow the format of the sample output given below. You must consider two decimal points for printing the fractional number; marks will be deducted otherwise.

Sample Content in sjf_input.txt :
p1 0 3 p2 1 5 p3 2 3
p1 1 3 p2 2 2 p3 2 1

Sample Output:
Test case #1: p1 0 3 p2 1 5 p3 2 3
Number of Processes: 3
Process Scheduling Started:

CPU Time 0: [P1 Arrived] P1 [0/3]
CPU Time 1: [P2 Arrived] P1 [1/3]
CPU Time 2: [P3 Arrived] P1 [2/3]
CPU Time 3: P1 [3/3]
Process P1 completed with Turnaround Time: 3, Waiting Time: 0
CPU Time 3: P3 [0/3]
CPU Time 4: P3 [1/3]
CPU Time 5: P3 [2/3]
CPU Time 6: P3 [3/3]
Process P3 completed with Turnaround Time: 4, Waiting Time: 1
CPU Time 6: P2 [0/5]
CPU Time 7: P2 [1/5]
CPU Time 8: P2 [2/5]
CPU Time 9: P2 [3/5]
CPU Time 10: P2 [4/5]
CPU Time 11: P2 [5/5]
Process P2 completed with Turnaround Time: 10, Waiting Time: 5
Process scheduling completed with Avg Turnaround Time: 5.67, Avg Waiting Time:2.00

Test case #2: p1 1 3 p2 2 2 p3 2 1
Number of Processes: 3
Process Scheduling Started:
CPU Time 0: None
CPU Time 1: [P1 Arrived] P1 [0/3]
CPU Time 2: [P2 Arrived] [P3 Arrived] P1 [1/3]
CPU Time 3: P1 [2/3]
CPU Time 4: P1 [3/3]
Process P1 completed with Turnaround Time: 3, Waiting Time: 0
CPU Time 4: P3 [0/1]
CPU Time 5: P3 [1/1]
Process P3 completed with Turnaround Time: 3, Waiting Time: 2
CPU Time 5: P2 [0/2]
CPU Time 6: P2 [1/2]
CPU Time 7: P2 [2/2]
Process P2 completed with Turnaround Time: 5, Waiting Time: 3
Process scheduling completed with Avg Turnaround Time: 3.67, Avg Waiting Time:1.67

Mark Distribution

A. Providing the program with *sjf_input.txt*: 5 points
B. Parsing the input file correctly: 10 points
C. Printing the number of processes for every test case: 5 points
D. Detecting process arrival correctly: 15 points
E. Detecting process completion correctly: 15 points
F. Calculating Turnaround Time and Waiting Time for every process: 15 points

G. Maintaining Shortest Job First sequence: 25 points
H. Calculating Avg Turnaround Time and Avg Waiting Time for every test case: 10 points

Computing Platform for Assignments

You are responsible for ensuring that your program compiles and runs without error on the computing platform mentioned below. **Marks will be deducted** if your program fails to compile or runs into errors on the specified computing platform (see below).

- Students have virtual access to the MC 244 lab, which contains 30 Fedora 28 systems. Linux machines available to you are **linux01.gaul.csd.uwo.ca** through **linux30.gaul.csd.uwo.ca**.
- It is your responsibility to ensure that your code compiles and runs on the above systems. You can SSH into MC 244 machines (please see the Assignment 1 file transfer tutorial).
- If you are off-campus, you have to SSH to **compute.gaul.csd.uwo.ca** first (this server is also known as sylvia.gaul.csd.uwo.ca, in honour of Dr. Sylvia Osborn), and then to one of the MC 244 systems **(linux01.gaul.csd.uwo.ca** through **linux30.gaul.csd.uwo.ca**) (please see the Assignment 1 file transfer tutorial).
- https://wiki.sci.uwo.ca/sts/computer-science/gaul

Assignment Submission

You need to submit only one C file. The name of your submitted C file must be **"assignment4.c".** Marks will be deducted if your submitted C file name is different. You must submit your assignment through OWL. Be sure to test your code on one of MC 244 systems (see "Computing Platform for Assignments" section above). **Marks will be deducted** if your program fails to compile or runs into errors on the computing platform mentioned above.

Assignment 4 FAQ will be made available on OWL as needed. Also, consult TAs and the Instructor for any questions you may have regarding this assignment.

Good Luck!!