*Prof. Dr. Burkhard Stiller, University of Zurich, Binzmühlestrasse 14, CH-8050 Zürich*
*Telephone: +41 44 635 6710, Fax: +41 44 635 6809, stiller@ifi.uzh.ch*
*Bruno Rodrigues, Telephone: +41 44 635 4381, rodrigues@ifi.uzh.ch*
*Sina Rafati, Telephone: +41 44 635 6702, rafati@ifi.uzh.ch*

# Exercises in SSDS

## Exercise 4

Hand out date: Mar 11, 2020
Discussion: Mar 19, 2020 08:30

### 1. Theoretical Exercises

1.1) Provide two programming examples in which multithreading does not provide better performance than a single-threaded solution.

1:
2:

1.2) The operating system is a program or set of programs whose function is to manage system resources, as well as to provide an interface between the computer and the user. On this subject, mark the **incorrect** alternative:

| | |
|---|---|
| | Process management is an operating system task |
| | Memory management is an operating system task |
| | The file system is defined and manipulated by the operating system |
| | All supported file types are defined by the operating system |
| | Managing the input and output devices is the task of the operating system |

1.3) Five processes must be run on one computer. The expected execution times for each of the processes are 10, 8, 4, 6 and X measured in some time unit. The responsible for the administration of the operating system of this computer decides to organize the order of execution of these processes, aiming at minimizing the average response time. It is known that the process with time X will be the third process to be executed. Under these conditions, a possible value for X is (mark the **correct** alternative):

| | |
|---|---|
| | 3 |
| | 5 |
| | 7 |
| | 9 |
| | 11 |

1.4) In the context of operating systems, the table below presents 5 processes with their start times and total required to run your service.

| Process | Start Time | CPU Burst |
|---|---|---|
| P1 | 0 | 3 |
| P2 | 2 | 6 |
| P3 | 4 | 4 |
| P4 | 6 | 5 |
| P5 | 8 | 2 |

Note:
- **Start time** is the time in which the process executed for the first time (i.e., scheduled)
- **Arrival time** is the time the process arrives at the ready queue

Knowing that the system has only one processor and that the scheduling algorithm is **Round-robin** (circular), with **context switch in each time unit (quantum = 1)**, when (in which time unit) process P5 will finish its execution? Mark the **correct** alternative.
*Processes are not interrupted for I/O.

| | |
|---|---|
| | 11 |
| | 13 |
| | 15 |
| | 17 |
| | 19 |

1.5) Consider the following set of processes and the CPU burst duration expressed in milliseconds:

| Process | CPU Burst |
|---|---|
| P1 | 28 |
| P2 | 4 |
| P3 | 7 |

If the processes arrive in the order P1, P2 and P3, where P1 arrives at time 0, what is the **average wait time** if the processes are serviced according to the first-come, first-Served (FCFS) scheduling? Mark the **correct** alternative.

| | |
|---|---|
| | 10.6 |
| | 12 |
| | 13 |
| | 14 |
| | 20 |

1.6) Consider the following processes, their CPU burst times given in milliseconds and priorities given in the table below.

| Process | CPU Burst | Priority |
|---|---|---|
| P1 | 10 | 3 |
| P2 | 1 | 1 |
| P3 | 2 | 3 |
| P4 | 1 | 4 |
| P5 | 5 | 2 |

Also, consider that the processes have arrived in the order P1, P2, P3, P4, P5, all at time 0.

Answer the questions (a, b and c) for each scheduling algorithm below. Finally, based on this scenario, answer which algorithm presents the lowest average waiting time (d).

**Round-Robin (RR) - Quantum = 1**
**First Come First Served (FCFS)**
**Shortest Job First (SJF)**

Hint: draw the Gantt chart for each algorithm (check slides #8, #9, etc from the Scheduling module).

a) What is the turnaround time of each process?

b) What is the waiting time of each process?

c) What is the average waiting time?

d) Which of these algorithms presents the lowest average waiting time?

## 2. Practical Exercises

```
#include <pthread.h>
gcc yourcode.c -lpthread
```

a) Matrix multiplication is one of the most famous examples for deploying threads. Based on the code of the previous exercise, construct a code to perform multiplication between two matrices A and B, storing the result in a matrix C. **Also, compare the execution time with a sequencial version of your code**.

Input: matrices A and B should be 3x3 (3 lines and 3 columns) and its content defined as I/O input.
Output: result in matrix C and ethe time elapsed in the multiplication.

## Submission Guidelines

- Submitted code must compile without errors.
- Zip your code using the name pattern: ssds_ex+**exercise+number_initials**. For instance, the file of Exercise 01 for Bruno Rodrigues would be named: sssds_ex01_BR.zip

Submit the exercise to the reseach assistants e-mail addresses.