# Pale INF422 – Composants d'un système informatique
# *INF422 Final Exam – Components of a Computing System*

2010–2011

- L'examen dure 3 heures.
  *The exam lasts 3 hours.*
- Tous documents autorisés.
  *All documents are authorized.*
- Il est impératif de commenter les programmes et de justifier vos réponses.
  *Programs must be commented and every answer must be justified.*

## English Text

### 1 Miscellaneous questions

It is a multiple choice questionnaire. Write down the number of the chosen answer. No need to justify your answers for this exercise.

Grading: **1** point for each good answer, **0** points if no answer, $-\mathbf{0.5}$ points for each false answer. The resulting grade between $-\mathbf{3}$ and **6** is normalized to the number of points for the exercise in the global exam.

#### Question 1.1

Three of the following commands list the processes whose name contains the `virus` substring but do not contain the `magic` substring. Which one does not display such a list?

(1) `ps -ef | grep virus | grep -v magic`
(2) `ps -ef > fic && grep virus < fic | grep -v magic`
(3) `ps -ef > fic || grep virus < fic | grep -v magic`
(4) `ps -ef > fic ; grep virus < fic | grep -v magic`

#### Question 1.2

What is the purpose of the `PATH` variable?

(1) indicate to the systen where to look for executable files;
(2) tell the home directory of the user;
(3) indicate to the systen where to look for text files;
(4) indicate to the systen where to look for data files.

#### Question 1.3

What is the result of executing the following script with argument `'ls*'` and the `PATH` variable is set to `/bin:/sbin`?

```
#!/bin/sh

for program in "$@"; do
  list=`echo $PATH | sed -e 's/:/ /g'`
  for directory in $list; do
    for file in $directory/$program; do
      [ -x "$file" ] && echo $file
    done
  done
done
```

(1) it fails with a syntax error;
(2) it prints, in this order, `/bin/ls`, `/bin/lspci`, `/sbin/lsmod` among other things;
(3) it prints, in this order, `/sbin/lsmod`, `/bin/ls`, `/bin/lspci` among other things;
(4) it does not print anything.

### Question 1.4

What is memory-mapped I/O?

(1) a mechanism to separe code from data in processes;
(2) a mechanism to communicate with devices;
(3) a mechanism to translate virtual addresses into physical ones;
(4) an implementation of input-output in the Java virtual machine.

### Question 1.5

What is `resUpdate` in the following Java code?

```
final Runnable resUpdate = new Runnable() {
    public void run() {
        textViewResult.setText("Number of primes: " + nbPrimes);
    }
};
```

(1) an instance of the `Runnable` interface, whose `run()` has been implemented;
(2) a thread, instance of the `Runnable` interface, whose method `run()` has been implemented;
(3) a thread, instance de the `Thread` class, whose method `run()` has been overloaded;
(4) a method.

### Question 1.6

We assume the decimal number 1000 is represented as a *little-endian* word of 16 bits in memory at address 42. What are the hexadecimal values of the bytes at address 42 and 43?

(1) 42: `0x00`; 43: `0x10`;
(2) 42: `0x10`; 43: `0x00`;
(3) 42: `0xE8`; 43: `0x03`;
(4) 42: `0x03`; 43: `0xE8`.

## 2 File system

The command `ls -al pale_inf422` prints the following output:

```
drwxr-----  3 acohen profs  4096 Nov  6 22:11 .
drwxr-xr-x 17 acohen profs  4096 Nov  6 21:58 ..
-rw-r--r--  1 acohen profs 16224 Nov  6 22:11 pale.tex
drwxr-xr-x  6 acohen profs  4096 Nov  6 22:01 .svn
```

### Question 2.1

List the sub-directories in `pale_inf422`.

### Question 2.2

How many sub-directories does the current directory contain (i.e., the parent of `pale_inf422`)? Justify your answer.

### Question 2.3

Can an other user from the `profs` group list the contents of the `pale_inf422` directory? Enter this directory? Read the `pale.tex` file? Justify your answer.

### Question 2.4

Can a user from the `eleves` group list the contents of the `pale_inf422` directory? Enter this directory? Read the `pale.tex` file? Justify your answer.

## 3 Scalar product

We wish to accelate the computation of the scalar product of 2 vectors of integers `x` and `y` of 10000 éléments. The target processor holds 2 cores and can execute 2 threads in parallel.

### Question 3.1

Implement a Java method creating 2 threads. Each thread is responsible of the first and second half of the elements of the 2 vectors, respectively, and computes the scalar product of two half-vectors of 5000 elements. The `x` and `y` vectors will be represented as arrays of 10000 elements, and each half-vector will be identified through an interval of disjoint indices (0 to 4999 and 5000 to 9999 respectively). The main thread waits for the termination of the 2 computational threads, then adds the 2 partial results and returns the final result.

### Question 3.2

Let us now assume that `x` and `y` are vectors of floating point numbers of type `float`. The previous parallelisation technique generally returns a different result from the sequential computation of the scalar product of 10000 elements. Explain why, and give an example where the sum of 3 floating point numbers produces very different results depending on the order of the additions.

## 4 Mouse pointer

We study the design of a driver for a mouse connected to a serial port, the update of the mouse pointer on the screen, and the interpretation of clicks on the button (we assume there is only one).

The mouse driver is generally a module of the system's kernel, written in the C and/or assembly language. We will simulate its behavior in Java.

Each state change (button press or release, mouse movement), the dedicated mouse microcontroller emits two bytes on the serial port towards the computer. These bytes represent the two's complement of the value between $-127$ and 127 of the horizontal and vertical displacement of the mouse since the last recorded position; when the first byte is $-128$ (binary representation $10000000_2$), the second byte can only take one of the following three 3 values:

- 0: periodic confirmation of the connection;
- 1: button pressed;
- 2: button released.

On the computer side, the serial port is managed by a dedicated microcontroller called UART. It manages a buffer of 16 bytes to store data coming from the serial port. Read and write accesses to the buffer and UART control are implemented through memory-mapped I/O, and exposed to Java programs through the following variables:

- `byte buf[16]`: buffer to store up to 16 bytes from the serial port;
- `byte cnt`: store the number of valid bytes in the buffer;
- `byte ack`: assigning this variable empties the buffer and authorizes the reception of new data starting at `buf[0]`;
- `int millis`: millisecond counter internal to the UART; can be reset to any value when assigning the variable.

As soon as a first byte is received in the buffer (in `buf[0]`), the UART informs the main processor by issuing an interrupt. It is simulated in Java through the asynchronous call to the `onReceive()` method in a dedicated thread (mechanism comparable to event management in the graphical interface of Android). Between the interrupt has been issued and its dedicated routine starts executing on the main processor, several bytes may accumulate in the buffer without issuing an additional interrupt. However, no additional data is stored in the buffer while the interrupt is running. Beyond 16 bytes, data is lost and the sender is informed; in the present case, the mouse simply ignores the message about data loss.

### Question 4.1

The pair of bytes $(-128, 0)$ allows the mouse to signal its presence to the computer. It is sent periodically every $10\,ms$ if the mouse state changed in this time interval.

What happens if the connection is interrupted between the emission of the first and the second byte on the serial link? What is the effect on the computer's interpretation of further mouse movements? Propose a fix based on the periodic connection code $(128, 0)$.

### Question 4.2

The current position of the mouse pointer is stored in the graphical interface in variables `mx` et `my`. Ces variables must be updated, adding the values received on the serial port when these are in the $-127$ to 127 range. Implement the `onReceive()` method to read the `cnt` bytes received and update the mouse position (no need to worry about bounding these variables to simulate the screen borders). Do not forget to inform the UART of the completion of the reception and to empty the buffer, writing (any value) in the `ack` variable. Assume the value of `cnt` is even and there is no transmission error.

### Question 4.3

When an odd number of bytes is received or when the transmission is interrupted, a shift may occur. Explain the different possible scenarios and modify your program to handle all the cases.

### Question 4.4

The `updateMouseState()` method of the graphical interface is responsible for displaying the mouse pointer at its new position in the framebuffer before each screen refresh cycle, i.e., about 50 times per second. What could happen if `onReceive()` is called at the same time the graphical interface reads variables `mx` and `my`? How to avoid this undesired behavior?

### Question 4.5

Button press, button release, click and double-click are respectively associated to four methods of the graphical interface: `onPress()`, `onRelease()`, `onClick`, `onDoubleClick()`. Give three reasons why it is out of the question to let the mouse driver call these methods directly.

### Question 4.6

To signal button events to the graphical interface, the driver must use a callback handler of the graphical interface called `buttonStateCallback`. Il sends it an instance of the `Runnable` interface whose methode `run()` calls one of the four methods to react to button events.

Write a Java program fragment illustrating this mechanism in the special case of a button press event.

### Question 4.7

Modify `onReceive()` to identify the type of the event (it is useless to complete the callback implementation for the 3 other cases). You will assume that a click is defined by the succession of a press and release event in less than 500ms, and a double-click is defined by the succession of two clicks in less than 500ms. If a click is detected, this event is reported instead of the succession of press and a release events. A click should not be reported before being certain that it is not a double-click.