System and device programming
28 / 02 / 2012

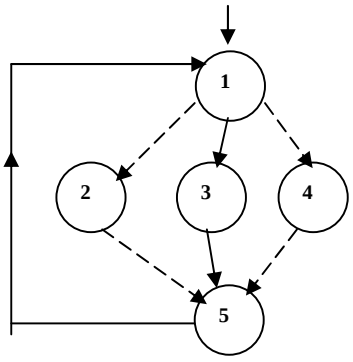
(Theory: no textbooks and/or course material allowed)

(15 marks) The final mark is the sum of the 1st > 8 and the 2nd part > 10
The final mark cannot be refused, it will be registered (no retry for marks >= 18)

1. (3.0 marks) Write the sequence of system calls that allows a process to catch the first received **`SIGURS1`** signal without aborting**,** ignore the second **`SIGUSR1`** signal that is received after the first one, and continue with the same behavior (catch – ignore) for the successive **`SIGUSR1`** signals.

2. (3.0 marks) Illustrate the syntax used to mounts a file system **`/dev/fsx`** on a directory **`dir`**. Every user is enabled to mount a file-system? Which kernel data structures are affected by the mounting operation? How?

3. (3.0 marks) Implement this precedence graph **with the minimum number of semaphores.** The processes are 5, and they are **cyclic**.
In every cycle **either** P2 **or** P4**, not both,** is executed in concurrency with P3.



4. (3.0 marks) Describe why the C library is not thread safe, and outline the main difference between the _beginthreadex and the CreateThread functions. Show why a program using CreateThread could be erroneous.

5. (3.0 marks) Which are the roles of files pointers and of the overlapped structures in direct file access on WIN32 systems. Briefly describe common aspects and differences. Provide a brief example of application for both of them.

System and device programming
28 / 02 / 2012

(C program: textbooks and/or course material allowed)

(18 marks) The final mark is the sum of the 1st > 8 and the 2nd part > 10
The final mark cannot be refused, it will be registered (no retry for marks >= 18)

Write a C program in the **Win32** environment which implements a shared array of Q FIFO queues, between a sets of M servers and a set of N clients.

The **main** program receives, as arguments on the command line, an input file name. The input file includes ***fixed length records***, with the following format:

- The first record contains three integer numbers, indicating the values of M, N and Q, respectively.
- The second record includes 2Q float numbers, describing the FIFO queues. For each queue the two numbers represent the minimum and the maximum predicted duration time for the tasks to be en-queued.
- All other records contain the description of a task. For each task the file specifies:
    - The name of the task: A character string of 40 bytes.
    - The number C of a client (clients are numbered from 0 to N-1).
    - An identifier: An unique number (an integer) for client C.
    - The predicted duration: A floating point number representing the number of milliseconds.

The main program starts M server threads, and N client threads. The FIFO queues are managed by one or more threads, depending on the student's choice. Each client thread reads the task assigned to it from the input file, i.e., client thread C is assigned tasks with client number C. The task is en-queued to a FIFO queue compatible with the predicted task duration (i.e., the task duration is included between the minimum and the maximum times for that queue) until it is extracted by a server thread.

Clients produce tasks, characterized by a name, an integer identifier (unique for the client), and a predicted duration. Tasks are en-queued to one of the Q FIFOs. FIFOs are numbered from 0 to Q-1. A task is queued to the FIFO matching the predicted time.

FIFOs are implemented as circular buffers of maximum size equal to 10. A client is blocked when trying to en-queue to a full FIFO.

Servers repeatedly extract tasks from the FIFO queues and simulate their execution. FIFOs with lower index have a larger priority. Simulation is performed by making the thread sleeping for a time equal to the predicted duration time.

Access to the common input file must be done concurrently using file locking. All operations must be done avoiding read/write conflicts on the shared FIFO queues, and maximizing concurrent processing.