Práctica 1

David Gutiérrez Villar

2ªDAM ESICA

Ejercicio 1:

La programación concurrente es la ejecución simultánea de múltiples tareas interactivamente. Estas tareas pueden ser un conjunto de procesos o hilos de ejecución creados por un único programa. Las tareas se pueden ejecutar en una sola CPU (multiprogramación), en varios procesadores, o en una red de computadores distribuidos.

Ejercicio 2:

La programación concurrente es un conjunto de acciones que pueden ser ejecutadas simultáneamente, en cambio, la programación paralela es un tipo de programación concurrente diseñado para ejecutarse en un sistema multiprocesador.

Ejercicio 3

1. El contexto de un proceso es la información en la CPU sobre un proceso que se está ejecutando.
2. Multiprogramación se refiere a que los procesos parecen que se están ejecutando a la vez sin embargo no es así si no que se van intercalando para que parezca que se ejecutan a la vez.
3. Son programas no interactivos con el usuario que se ejecutan en segundo plano.
4. En UNIX con el fork un proceso que solicita crear otro proceso crea un proceso exactamente igual a el (hijo). El padre y el hijo tienen identificadores de programa distintos. La jerarquía en UNIX es que un proceso solo puede tener un padre y 0 o más hijos. En Windows no hay jerarquía.
5. Un proceso en UNIX solo puede tener un padre.
6. Un proceso puede tener infinitos hijos.
7. Estados:
    - Nuevo: El proceso se acaba de crear y entra en ejecución pero no se ha cargado en memoria.

- Listo: Ya está cargado en memoria y listo para ejecutarse pero no se está ejecutando.
- Bloqueado: Proceso esperando a un evento.
- Saliente: Proceso terminado.
- Ejecutando: El proceso se está ejecutando.

8 Es cuando en la CPU se cambia la información de un proceso que se estaba ejecutando por la nueva información del nuevo proceso que se va a ejecutar.

Ejercicio 4

Es el bloqueo permanente de un conjunto de procesos que compiten por los recursos del sistema o bien se comunican unos con otros. A diferencia de otros problemas de la gestión concurrente de procesos, no existe una solución eficiente.

Condiciones:

- Condición de exclusión mutua: existencia de al menos un recurso compartido por los procesos, al cual solo puede acceder uno simultáneamente.
- Condición de retención y espera: Un proceso que ha recibido un recurso esta esperando a otro a la vez que retiene el que ya tiene.
- Condición de no expropiación: Los recursos solo son liberados cuando los procesos que los están utilizando acaben con ellos o ya no los necesiten.
- Condición de espera circular: Un proceso está esperando un recurso que tiene otro proceso que a su vez está esperando un recurso que tiene otro proceso y así sucesivamente hasta que el último proceso quiere un recurso que tiene el primero de ahí lo de circular.

Ejercicio 5

La clase processBuilder. Ejemplo para hacer ping:

```
ProcessBuilder processBuilder = new ProcessBuilder();
        processBuilder.command("cmd.exe", "/c", "ping -n 3 google.com");
        try {
            Process process = processBuilder.start();
            BufferedReader reader =
                    new BufferedReader(new
              InputStreamReader(process.getInputStream()));
            String line;
            while ((line = reader.readLine()) != null) {
                System.out.println(line);
            }
            int exitCode = process.waitFor();
            System.out.println("\nExited with error code : " + exitCode);
        } catch (IOException e) {
            e.printStackTrace();
        } catch (InterruptedException e) {
            e.printStackTrace();
          }
```

Ejercicio 6

Es un algoritmo que se encarga de gestionar el orden en que cada uno de los procesos se ejecuta o dicho con más precisión tiene el control de la CPU. Para tomar esa decisión se usan diferentes algoritmos.

Tipos:

- FCFS (First-Come, First-Served): En este algoritmo el primer proceso que llega es el primer proceso que se ejecuta. Es el algoritmo con menos rendimiento. Desfavorece a los procesos cortos y a los que tienen muchas interrupciones.
- SJF (Shortest Job First): Este algoritmo siempre prioriza a los procesos más cortos. Si los procesos son igual de cortos se utiliza el metodo FCFS anteriormente explicado. Este algoritmo tiene un problema y es que puede pasar que al poner los procesos mas largos siempre alfinal de la cola nunca se ejecuten, esto se conoce como inanición.
- SRTF (Short Remaining Time Next): Este algoritmo es una variante del anterior (SJF) en la cual se añade la expulsión de procesos que consiste en la expulsión de procesos largos en ejecución para ejecutar otros más cortos. Este algoritmo también tiene un problema y es que un proceso largo puede ser expulsado tantas veces que no llegue a terminar.
- Round Robin: Este algoritmo asigna a cada proceso un tiempo equitativo tratando a todos los procesos por igual. Siempre vuelve al primer proceso una vez terminado con el último intervalo.

Ejercicio 7

Programa:

```
public class Main {

    private static final String Ruta = "helloBatch.bat";

    public static void batch(String nombre) {

        Process processBuilder;

        try {
```

```
                processBuilder = new ProcessBuilder(new
                File(Ruta).getAbsolutePath(), nombre).start();
                BufferedReader br = new BufferedReader(new
                InputStreamReader(processBuilder.getInputStream()));
                String line;
                while ((line = br.readLine()) != null) {
                        System.out.println(line);
                }
            } catch (IOException e) {
                e.printStackTrace();
            }
        }
        public static void main(String[] args) throws IOException {
            batch(JOptionPane.showInputDialog(null, "Introduzca su nombre
            para hacer el echo:"));
        }
}
```