Práctica final: Vacunación COVID19

Miguel Ángel Conde González
Antonio Gómez García
Mario Enrique Casado García

17 de diciembre de 2020

Definición del problema

El objetivo de la práctica es realizar un programa que simule la vacunación en un consultorio médico de la COVID a partir de 2021, para ello se establecerá un sistema de vacunación por edades, y se llevará a cabo un estudio de los resultados posteriores. Los pacientes, los médicos y los que atienden al estudio se representarán mediante threads. La práctica tiene una parte básica donde se define el funcionamiento básico del sistema y que es obligatorio implementar (supondrá como mucho el 80 % de la nota) y, además, se proponen una serie de mejoras opcionales para aumentar la nota (como mucho supondrá el 20 % de la nota).

Para aprobar la práctica es necesario que la parte básica funcione correctamente. La nota se asignará en base a la calidad del código entregado, valorándose:

- Política de nombres (coherencia en el nombrado de variables y funciones).
- Sangrado (indentación).
- Comentarios (cantidad, calidad y presentación).
- Legibilidad: Nombre de variables, funciones, etc.
- Reusabilidad y mantenibilidad: Uso de parámetros, etc.

La práctica es en grupo y se evaluará según la metodología CTMTC explicada en clase.

Parte Básica (80 % de la nota)

La aplicación funciona de la siguiente manera:

- Los pacientes van a entrar en el consultorio y se van a separar por edades a la espera de que se les comience a vacunar. En concreto se distribuyen los pacientes en junior (hasta los 16 años), medios (de 16 a 60 años) y senior (de 60 en adelante).

- Al consultorio solamente van a poder entrar 15 personas a la vez, si se supera esa cifra la persona no podrá vacunarse y su solicitud de vacunación será rechazada. Para atender a los pacientes se cuenta con 3 enfermer@s y un médico. Cada uno de los enfermer@s atenderá pacientes de un rango de edad, por lo tanto habrá uno asignado para los juniors, otro para los medios y otro para los senior. El médico, a su vez, auxiliará a aquellas colas con más solicitudes y podrá por tanto vacunar en cualquier rango de edad. Esto supone que de los 15 pacientes 4 puedan ser atendidos y 11 estarían esperando.

- Un 20 % de los pacientes, se cansa de esperar y se va y un 10 % al final se lo piensa mejor y también abandona el consultorio. Del 70 % restante un 5 % pierde el turno por ir al baño mientras espera.

- A cada paciente se le asignará un identificador único y secuencial (paciente_1, paciente_2,... paciente_N) a medida que vayan entrando en el consultorio.

- Una vez un paciente ha sido vacunado, aún debe esperar en el consultorio para ver si la vacuna le da reacción y en caso de que así sea, tendrá que ser atendido por el médico durante 5 segundos antes de abandonar el consultorio. Solo da reacción a un 10 % de los pacientes.

- Un 25 % de los pacientes vacunados va a participar en un estudio sobre seroprevalencia, para ello se cuenta con un estadístico (otro thread). Este comprobará quien está implicado y le realizará la encuesta y tomará los datos (empleará 4 segundos para ello). Hasta que no acabe el estudio los encuestados no podrán abandonar el consultorio.

Aspectos a tener en cuenta:

- Los enfermer@s tienen un un identificador único (enfermer@_1, enfermer@_2, enfermer@_3).

- Solo cuando se haya vacunado a un paciente se pasará a comprobar si tiene reacción, al estudio serológico o se marchará.

- Los pacientes comprueban cada 3 segundos si se marchan por alguna de las posibles razones (se cansan, se lo piensan mejor o van al baño y pierden turno) y en caso de ser así abandonan la cola.

- Cada vez que se atienden 5 pacientes, el enfermer@ descansa 5 segundos, con lo que o los atiende otro compañero o tendría que atender a los de su rango de edad el médico.

- Los pacientes de cada tipo deben atenderse por orden de llegada.

- Si algún enfermer@ no tiene pacientes que atender de su rango de edad puede vacunar a otros de otro tipo, pero una vez termine de hacerlo deberá comprobar de nuevo los pacientes de su tipo.

- El médico atiende a quien le da reacción y si no hay nadie vacuna a los pacientes del rango edad con mayor número, dentro de estos siempre sigue el criterio de antigüedad.

- De los pacientes a atender, el 80 % tiene todo en regla, el 10 % no se ha identificado correctamente y el 10 % tiene catarro o gripe. Esto tiene una implicación en los tiempos de atención:
  - 80 % todo en regla – En estos casos, el tiempo de espera está entre 1 y 4 segundos y después se comprobará reacción y si se participa en el estudio.
  - 10 % mal identificados – En estos casos, el tiempo de espera está entre 2 y 6 segundos y después se comprobará reacción y si se participa en el estudio.
  - 10 % tiene catarro o gripe – En estos casos, el tiempo de espera está entre 6 y 10 segundos y no pueden vacunarse, así que no esperarán a la reacción ni participarán en el estudio, abandonando el consultorio.
- El los dos primeros casos se vacuna y se participa en el estudio serológico según los porcentajes previamente comentados. Transcurrido el tiempo considerado se abandonará el consultorio.

Toda la actividad quedará registrada en un fichero plano de texto llamado `registroTiempos.log`. En concreto, es necesario registrar al menos:

- Cada vez que un paciente accede al sistema
- Cada vez que una paciente deja el consultorio por el motivo que sea
- Cada vez que un paciente se vacuna correctamente.
- Cuando a un paciente le da reacción la vacuna.
- Cuando un paciente participa en el estudio serológico.
- Se registra el inicio y final del descanso de cada enfermer@.
- Al finalizar el programa se debe terminar de vacunar a todos los pacientes en cola, pero ya no podrán participar en el estudio serológico.

Consideraciones prácticas:

- Simularemos el inicio de funcionamiento del sistema mediante señales. En caso de que un paciente junior que quiera acceder al consultorio se usará la señal SIGUSR1, en el caso de un paciente medio SIGUSR2 y en el caso de un paciente senior SIGPIPE. Cada vez que se le envíe la señal, supone que un nuevo paciente trata de acceder al consultorio.
- Es obligatorio el uso de mensajes que se escribirán en un log y se mostrarán por pantalla. El formato de tales mensajes será:

  `[YYYY-MM-DD HH:MI:SS] identificador: mensaje`

  Donde `identificador` puede ser el identificador del paciente, el identificador del enfermero o el médico y `mensaje` es una breve descripción del evento ocurrido.
- Las entradas del log deben quedar escritas en orden cronológico.
- El programa finaliza cuando recibe la señal SIGINT y deberá hacerlo correctamente.

Partes opcionales (20 % de la nota)

- Asignación estática de recursos (10 %):
  - Modifica el programa para que el número de pacientes que pueden tratarse en el sistema sea un parámetro que reciba el programa al ser ejecutado desde la línea de comandos.
  - Modifica el programa para que el número de enfermer@s que atienden pacientes senior sea un parámetro que reciba el programa al ser ejecutado desde la línea de comandos.
- Asignación dinámica de recursos I (5 %):
  - Modifica el programa para que el número de pacientes que pueden acceder al consultorio pueda modificarse en tiempo de ejecución.
  - Solamente es necesario contemplar un incremento en los pacientes. No es necesario contemplar la reducción.
  - Cada vez que se cambie el número de pacientes tiene que reflejarse en el log.
- Asignación dinámica de recursos II (5 %):
  - Modifica el programa para que el número de enfermer@s senior se pueda modificar en tiempo de ejecución.
  - Solamente es necesario contemplar un incremento en el número de enfermer@s senior. No es necesario contemplar la reducción.
  - Cada vez que se produce un cambio en este sentido debe quedar reflejado en el log.

Escritura de mensajes en log

Es recomendable utilizar una función parecida a esta para evitar repetir líneas de código. Recibe como parámetros dos cadenas de caracteres, una para el identificador y otra para el mensaje (la fecha la calcula la propia función):

```c
void writeLogMessage(char *id, char *msg) {
  // Calculamos la hora actual
  time_t now = time(0);
  struct tm *tlocal = localtime(&now);
  char stnow[25];
  strftime(stnow, 25, "%d/%m/%y %H:%M:%S", tlocal);

  // Escribimos en el log
  logFile = fopen(logFileName, "a");
  fprintf(logFile, "[%s] %s: %s\n", stnow, id, msg);
  fclose(logFile);
}
```

Ejemplo 1: Diseño de la parte básica