Urgencias

El objetivo de este control es desarrollar un TAD específico para una aplicación.

1) Trabajo a realizar

Nos han encargado implementar un sistema para la gestión de la admisión en el Servicio de Urgencias de un Hospital. Cuando un paciente llega al servicio, se le toman los datos y se le asigna un código de identificación único (un número entero no negativo). A partir de ahí, el paciente espera a ser atendido. El orden de atención da prioridad a los pacientes por orden de llegada. Una vez atendido un paciente, sus datos se eliminan del sistema. Así mismo, en cualquier momento un paciente puede desistir de ser atendido. En este caso, en el control de salida se le solicita su número de identificación, y se elimina todo rastro de él del sistema.

La implementación del sistema se deberá realizar como un TAD `GestionAdmisiones` con las siguientes operaciones:

- `GestionAdmisiones()`: Operación constructora que crea un sistema de gestión de admisiones vacío.
- `an_paciente(codigo, nombre, edad, sintomas)`: añade al sistema un nuevo paciente con código de identificación `codigo`, con nombre `nombre`, con edad `edad` y con una descripción de síntomas `sintomas`. En caso de que el código esté duplicado, la operación señala un error `EPacienteDuplicado`.
- `info_paciente(codigo, nombre, edad, sintomas)`: en `nombre`, `edad` y `sintomas`, que son parámetros de salida, se devuelve la información correspondiente al paciente con código `codigo`. En caso de que el código no exista, se levanta un error `EPacienteNoExiste`.
- `siguiente(codigo)`: almacena en `codigo`, que es un parámetro de salida, el código del siguiente paciente a ser atendido. En caso de que no haya más pacientes, esta operación levanta un error `ENoHayPacientes`.
- `hay_pacientes()`: devuelve `true` si hay más pacientes en espera, y `false` en otro caso.
- `elimina(codigo)`: elimina del sistema todo el rastro del paciente con código `codigo`. Si no existe tal paciente, la operación no tiene efecto.

Dado que este es un sistema crítico, la implementación de las operaciones debe ser lo más eficiente posible. Por tanto, debes elegir una representación adecuada para el TAD, implementar las operaciones y justificar la complejidad resultante.

2) Código de apoyo

El material proporcionado es el siguiente:

- En el archivo `main.cpp` está el programa de prueba. **Este programa no debe modificarse**.
- En el archivo `GestionAdmisiones.h` deben añadirse los campos necesarios para soportar las estructuras de datos requeridas por el TAD, así como todas las clases y definiciones adicionales que se consideren oportunas.
- En el archivo `GestionAdmisiones.cpp` deben implementarse todos los métodos declarados en `GestionAdmisiones.h`.

El programa principal lee y ejecuta una serie de acciones a realizar sobre un sistema de gestión de admisiones. Las posibles órdenes son las siguientes:

- `admite` *código nombre edad síntomas*: Admite un paciente con código *código*, nombre *nombre*, edad *edad*, y síntomas *síntomas*. Tanto *nombre* como *síntomas* no deben contener espacios en blanco. Como respuesta imprime `OK` si el código no está duplicado, o `CODIGO_PACIENTE_DUPLICADO` en otro caso.
- `atiende`: Simula la atención de un paciente (obtiene la información del siguiente paciente a atender, y lo elimina del sistema). Como respuesta imprime `OK`, o `NO_HAY_PACIENTES` en caso de que en el sistema no haya pacientes.
- `siguiente`: Muestra toda la información del siguiente paciente a atender. Como respuesta muestra una línea del tipo *código nombre edad síntomas*, o bien `NO_HAY_PACIENTES` en caso de que en el sistema no haya pacientes.
- `datos` *c*: Visualiza los datos (*nombre*, *edad* y *síntomas*) del paciente con código *c*.
- `desiste` *c*: Elimina el paciente con código *c* del sistema (si es que existe). Siempre imprime `OK` como respuesta.
- `en_espera`: Imprime `SI` si hay pacientes en espera, y `NO` en otro caso.

A continuación, se muestra un ejemplo de entrada / salida:

Entrada	Salida
admite 1 jose 22 dolor_muelas admite 2 pepe 25 dolor_barriga en_espera siguiente admite 2 juan 23 dolor_cabeza admite 3 juan 23 dolor_cabeza siguiente atiende datos 3 datos 4 datos 1 siguiente atiende siguiente atiende siguiente atiende en_espera	OK OK SI 1 jose 22 dolor_muelas CODIGO_PACIENTE_DUPLICADO OK 1 jose 22 dolor_muelas OK juan 23 dolor_cabeza PACIENTE_NO_EXISTE PACIENTE_NO_EXISTE 2 pepe 25 dolor_barriga OK 3 juan 23 dolor_cabeza OK NO_HAY_PACIENTES NO_HAY_PACIENTES NO

IMPORTANTE: **Debe incluirse nombres y apellidos al comienzo de** `GestionAdmisiones.h` **(si no, el control no se corregirá)**.