Ejercitación Clases

La ejercitación consiste en completar y codificar desde 0 funcionalidades de una biblioteca de manejo de fechas. Todos los ejercicios pueden resolverse en src/Fecha.cpp. En los directorios src y tests se pueden encontrar los archivos Horario.cpp, test_horario.cpp y test_horario_main.cpp de referencia.

Se recomienda resolver los ejercicios en orden. En CLion se encuentran disponibles los siguientes targets:

- horarios_main: compila los tests de horarios que no usan gtest
- horarios_gtest: compila los tests de horarios que usan gtest
- ejN: compila todos los tests hasta el ejercicio N inclusive ($1 \leq N \leq 8$).

Los targets también pueden compilarse y ejecutarse sin usar CLion. Para ello:

1. En una consola pararse en el directorio raíz del projecto. En este debería haber un archivo CMakeLists.txt.
2. Ejecutar el comando $> cmake . Esto generará el archivo Makefile
3. Ejecutar el comando $> make TTT donde TTT es uno de los targets mencionados anteriormente. Esto creará un ejecutable con el nombre del target en el directorio actual.
4. Ejecutar el comando $> ./TTT siendo TTT el nombre del target utilizado anteriormente. Esto correrá el ejecutable.

Ejercicio 1

Completar la función `bool esBisiesto(Anio anio)` que devuelva `true` sii el anio es bisiesto.

Ejercicio 2

Escribir la función `int diasEnMes(Anio anio, Mes mes)` que devuelva la cantidad de días en el mes de ese año.

Ejercicio 3

Implementar el constructor y los métodos para obtener los atributos de la clase Fecha.

Ejercicio 4

Implementar los operadores de comparación (==, <) de la clase Fecha.

Ejercicio 5

Implementar el operador de desigualdad de la clase Fecha.

Ejercicio 6

Completar la declaración de la `class Periodo`. La clase Periodo representa un período de tiempo en años, meses y días. Notar que no se puede representar solamente con días porque cuantos días representa un mes depende de a partir de que año y mes se cuenta. La clase debe tener los siguientes métodos públicos:

- `Periodo(int anios, int meses, int dias)`: Constructor de la clase.
- `int anios() const`: los años guardados en la instancia
- `int meses() const`: los meses guardados en la instancia
- `int dias() const`: los dias guardados en la instancia

Ejercicio 7

Implementar los métodos privados de la clase Fecha

```
void sumar_anios(int anios);
void sumar_meses(int meses);
void sumar_dias(int dias);
```

Luego, agregar e implementar el método público `void Fecha::sumar_periodo(Periodo p)` en Fecha. El mismo debe modificar la fecha sumándole el período pasado por parámetro.

Estos métodos agregan días/meses/años y modifican el estado interno de la Fecha. Tener en cuenta que es necesario redondear los días y los meses. Por ejemplo:

```
Fecha f(1981, FEBRERO, 20);
f.sumar_periodo(Periodo(0, 0, 10));
EXPECT_EQ(f.mes(), MARZO);
EXPECT_EQ(f.dia(), 2);

f.sumar_periodo(Periodo(0, 10, 0));
EXPECT_EQ(f.anio(), 1982);
EXPECT_EQ(f.mes(), ENERO);
```

Ejercicio 8

Crear una clase `class Intervalo`. La clase Intervalo representa un intervalo entre dos fecha. Debe tener los siguientes métodos públicos.

- `Intervalo(Fecha desde, Fecha hasta)`: el constructor donde desde es anterior a hasta.
- `Fecha desde() const`: devuelve la fecha de inicio del intervalo
- `Fecha hasta() const`: devuelve la fecha de fin del intervalo
- `int enDias() const`: devuelve la cantidad de días del intervalo.