LAB2: Tipos genéricos

Objetivo

El objetivo de este laboratorio es ejercitar los siguientes conceptos:

- Programación genérica

Tarea a realizar

Diseño e implementación de una matriz genérica

Se desea implementar una clase que represente una matriz bidimensional de números reales que permita realizar operaciones de álgebra lineal básica. Las matrices deben poderse construir con cualquier tipo aritmético (enteros, reales, números complejos,…).

```
matriz<double> a{filas, columnas};
matriz<float> b{filas, columnas};
matriz<std::complex<float> > c{filas, columnas};
```

Deben ofrecerse como mínimo las siguientes funcionalidades:

- Constructor a partir de número de filas y columnas. Todos los valores de la matriz se inician a al valor por defecto para el tipo base de la matriz. La matriz almacena el número de filas y de columnas y un búfer en los que almacenan todos los elementos de la matriz.
- Constructor por defecto. Se crea una matriz de cero filas y cero columnas que no consume memoria dinámica, pero sobre la que posteriormente se puede copiar otra matriz.
- Debe soportar operaciones de copia y movimiento.
- El operador paréntesis se sobrecargará con dos argumentos para acceder a una posición de la matriz.
- Los operadores + y * se sobrecargarán para implementar la suma y el producto de matrices.

Diseño e implementación de números racionales

Un problema frecuente con el uso de números en coma flotante tiene que ver con la representación aproximada de valores. Otro problema tiene que ver con la asociatividad de las operaciones.

Diseñe un tipo para representar números racionales. Debe representar un número racional como una fracción con un numerador y denominador de un mismo tipo entero. El tipo entero se pasará como argumento de plantilla.

```
racional<int> r{4,5}; // 4/5
racional<long> r{1,2}; // 1/2
```

Busque una manera de evitar que sea posible instanciar los números racionales con tipos que:

- No sean tipos enteros.
- Sean enteros sin signo.

La representación interna de un número racional siempre debe contener la fracción más simple posible.

Implemente suficientes propiedades de los tipos racionales como para que sea posible utilizar todas las operaciones de matrices de valores racionales.

```
matriz<racional<int>> m{100,100}; //100 filas, 100 columnas
```

Evaluación

Escriba un programa que genere tres matrices cuadradas (A, B, y C) de un tamaño constante N y las rellene con números aleatorios. Los números aleatorios se generarán de manera que sigan una distribución normal con media 2.5 y desviación típica 5.0.

Posteriormente el programa debe realizar las siguientes operaciones:

1. Calcular la matriz D como el resultado de la expresión A*B+C.
2. Calcular el valor S como la suma de todos los valores de la diagonal principal.
3. Imprimir el valor S por la salida estándar (cout).

Evalúe el tiempo que tarda en ejecutarse el programa para distintos valores de N (100 y 1000). Compare el comportamiento para matrices de float, doublé, long double y racional<int>.

Deberá analizar de forma independiente el tiempo total de la aplicación y el tiempo requerido solamente por la expresión D=A*B+C.

Información adicional

- Para la generación de números aleatorios consulte la información del componente <random> de la biblioteca estándar. Ver: https://en.cppreference.com/w/cpp/numeric/random/uniform_real_distribution.
- Para medir el tiempo total de ejecución de la aplicación puede usar los siguientes mandatos desde Linux:
    - Mandato: perf stat (ver man perf-stat).
    - Mandato: time (ver man time).
- Para medir el tiempo que tarda en ejecutarse una sección de código, puede hacer uso del componente chrono de la bilbioteca estándar. Ver: https://en.cppreference.com/w/cpp/chrono/high_resolution_clock/now.

Material a entregar

Se entregará:

1. Código fuente.

2. Pequeña memoria explicativa que incluirá un análisis de rendimiento. La memoria será un breve documento en formato PDF.

Para entregar el código fuente se entregarán en un archivo comprimido:

- matriz.h: Archivo de cabecera para matriz.
- racional.h: Archivo de cabecera para números racionales.
- main.cpp: programa principal que realiza la evaluación.