Tarea 4

Programación orientada al objeto

$1^{er.}$ Semestre 2018

Introducción

Esta tarea está diseñada para que Ud. diseñe una clase simple, realice sobrecarga de operadores y conversión entre objetos y tipos de datos primitivos.

La clase `MFD`

Un número *minifloat decimal* (MFD) es un objeto que representa un número de punto flotante utilizando un valor booleano y 3 datos enteros que representan el signo, la parte entera, la parte fraccionaria y la potencia de 10 del número de punto flotante correspondiente. En otras palabras, un objeto *MFD* con miembros *true*, 3, 14159265359, 0, respectivamente, representa el número π.

En el formato MFD, siempre intente almacenar números de punto flotante de manera que la parte fraccionaria comience con un dígito diferente a 0, para así evitar errores de representación. Esto quiere decir que el número 1.0203 debe almacenarse como 10.203×10^{-1}.

```
class MFD {
    public:
        // AGREGUE AQUI SU CODIGO

    private:
        bool signo;          // Toma valores true o false
        long int entero;     // Parte entera del MFD
        long int fraccion;   // Parte fraccional del MFD
        long int exponente;  // Se utiliza base 10
}
```

Escriba un constructor para esta clase, que reciba 4 argumentos: los valores iniciales de las variables miembro. Si el primer argumento está ausente, el número representado es positivo. Si cualquiera de los otros 3 argumentos está ausente, entonces la variable miembro correspondiente toma el valor 0.

Agregue además otros dos constructores a la clase, que reciban como argumentos un entero de tipo `int` y un número de punto flotante de doble precisión, respectivamente.

Escriba funciones de acceso llamadas `bool signo()`, `long int entero()`, `long int fraccion()` y `long int exponente()` que retornen los valores de las variables miembro correspondientes, y funciones de acceso `setSigno(bool s)`, `setEntero(long int e)`, `setFraccion(long int f)` y `setPotencia(long int p)` que permitan modificar estas variables miembro.

Agregue ahora una función de impresión `void imprime()` que imprime este objeto `MFD` en la consola, utilizando notación exponencial. En el ejemplo del caso anterior, su función debe imprimir `3.14159265359E0`.

Sobrecargue las 4 operaciones aritméticas básicas de manera que retornen un nuevo objeto `MFD` con los resultados correctos correspondientes. Los resultados deben ser normalizados de manera que la parte fraccionaria no comience con un 0. Para la división de objetos `MFD`, si el numerador es igual a 0, entonces se debe generar una excepción del tipo `domain_error` cuyo argumento debe ser un mensaje de explicación.

Asimismo, sobrecargue los operadores `+=`, `-=`,`*=`, `/=`, de manera que modifiquen el objeto y lo retornen. Sobrecargue además los operadores de comparación `==` y `!=`. Tenga en cuenta que los números `MFD(true, 5, 0, 1)` y `MFD(true, 50, 0, 0)` son iguales.

Sobrecargue los operadores `>>` y `<<` para poder leer objetos `MFD` desde un flujo de entrada y escribirlos a un flujo de salida.

Escriba una función `double toDouble()` que retorne un número de punto flotante de doble precisión de valor equivalente al objeto `MFD`. Si el valor no cabe en el número de punto flotante, entonces se debe generar una excepción de tipo `domain_error` cuyo argumento debe ser un mensaje de explicación.

Escriba también una función `String toString()` que retorne un objeto `String` equivalente al objeto `MFD`.

Finalmente, utilice estos objetos `MFD` para calcular 1000!

Para el desarrollo de esta tarea, Ud. debe utilizar principalmente los iteradores, contenedores y algoritmos de la biblioteca estándar. Su código será evaluado no sólo de acuerdo a su efectividad, sino también su eficiencia.

Esta tarea puede ser realizada en grupos de máximo 3 personas. Su código fuente deberá ser subido a InfoAlumno antes de las 6 pm del día 6 de julio. Además, deberá entregar un informe escrito en secretaría de Electrónica antes de esa fecha, que *debe* incluir un listado de su programa, además de una descripción de las dificultades encontradas y cómo fueron solucionadas.