1151044 - Programación Orientada a Objetos
Tarea 1 - 2026-I

Implementa un tipo de dato `fraccion` que modele una fracción o número racional. Tanto el numerador como el denominador de la fracción deben representarse por variables `int` como sigue:

```
struct fraccion {
   int num, den;
};

fraccion a;
a.num = 1;
a.den = 2;                              // a vale ½
fraccion b = { 1, 2 };                  // b vale ½ también
fraccion c = fraccion(1, 2);            // c vale ½ también
fraccion d(1, 2);                       // d vale ½ también
fraccion e = { .num = 1, .den = 2 };    // e vale ½ también
```

Sean `f1` y `f2` fracciones. Las operaciones disponibles deben ser las siguientes.

- Suma de fracciones.

```
fraccion f3 = f1 + f2;
```

- Resta de fracciones.

```
fraccion f3 = f1 - f2;
```

- Multiplicación de fracciones.

```
fraccion f3 = f1 * f2;
```

- División de fracciones.

```
fraccion f3 = f1 / f2;
```

- Identidad aritmética.

```
fraccion f2 = +f1;
```

- Negación aritmética.

```
fraccion f2 = -f1;
```

- Recíproco de una fracción[1].

```
fraccion f2 = ~f1;
```

- Comparación de fracciones (nota que la fracción $\frac{1}{2}$ **es igual** a $\frac{2}{4}$ aún si sus representaciones internas difieren).

```
bool b1 = (f1 < f2);
bool b2 = (f1 <= f2);
bool b3 = (f1 > f2);
bool b4 = (f1 >= f2);
bool b5 = (f1 == f2);
bool b6 = (f1 != f2);
```

- Conversión a número en punto flotante, específicamente a `double`, mediante llamada a función denominada `flotante`.

```
double d = flotante(f1);
```

[1] Es bastante debatible si definir este operador en la vida real es buena idea.

- Lectura de fracciones (entero seguido del símbolo / seguido de entero, sin espacios entre ellos).

```
std::cin >> f1;        // 1/2 al leerse vale ½
```

- Impresión de fracciones (entero seguido del símbolo / seguido de entero, sin espacios entre ellos).

```
std::cout << f1;       // el valor ½ se imprime 1/2
```

Sólo puedes usar los archivos de biblioteca `iostream` y `numeric`. Tu código no debe declarar variables estáticas ni globales. Puedes consultar una página de prueba en https://racc.mx/uam/trimestre-actual/2026-i/poo/tarea1.html. El código que envíes no debe contener `main` y no debe leer ni imprimir nada directamente.

Envía tu código fuente desde tu cuenta institucional al formulario en https://forms.gle/hKTWWHzwj873dGN38. Tu código será evaluado con varios casos de prueba y se espera que cumpla la semántica descrita. Puedes suponer que los casos de evaluación no desencadenarán divisiones entre cero ni sobreflujo (*overflow*) al manipularlas usando los algoritmos comunes de matemáticas, aún si las fracciones nunca se simplifican.

Ejemplo de uso	**Ejemplo de salida**

`int main( ) {`
`  fraccion f1 = { 1, 2 };`
`  fraccion f2 = { 3, 4 };`
`  std::cout << f1 + f2;`
`}`
	`5/4`