POO LAB 1

Write the class **Fractie (**whose objects are common fractions representing rational numbers**)** so that the following code

```
int main()
{
    Fractie f1(1, 4);
    Fractie f2(8, 16);
    Fractie f3 = f1 + f2;
    f1.print(); printf(" + "); f2.print(); printf(" = "); f3.print(); printf("\n");
    Fractie f4 = f2 - f1;
    f2.print(); printf(" - "); f1.print(); printf(" = "); f4.print(); printf("\n");
    Fractie f5 = f3 * f4;
    f3.print(); printf(" x "); f4.print(); printf(" = "); f5.print(); printf(" [");
    Fractie f6 = f5.simplify();
    f6.simplify(); f6.print(); printf("]\n");

    Fractie f7 = { 2,4 };
    Fractie f8 = "10/20";
    if (f7 == f8) { printf("Egale\n"); }
    Fractie f9 = "123";
    f9.print();
    Fractie f10 = { 123,321 };
    printf("\n[%d/%d]", f10["numerator"], f10["denominator"]);
}
```

compiles and upon execution prints the following to the screen:

```
1/4 + 8/16 = 12/16
8/16 - 1/4 = 4/16
12/16 x 4/16 = 48/256 [3/16]
Egale
123/1
[123/321]
```

Observations:

- You are not allowed to use **STL** at all (for vectors, strings, maps or any **template/object** defined in STL). The only exception is the usage of "**std::cout**" from the main function
- You are not allowed to use string manipulation functions defined in "string.h" such as **strlen**, **strcpy**, **strdup, strtok, strcmp**, etc, or string to number conversions.
- If you don't respect the previous conditions (e.g. use strlen, or strcpy, etc) → we will compute the correctness of the code, but the final grade will be half of the computed score for each particula code that uses those functions.
- For the purpose of this example, we will consider that all numbers are positive and the denominator is never 0 (you don't need to check for this). We will also consider that the string format is always correct (either "number/number" or just "number") → in the last case the denominator will be considered 1.

Grading (informative):

G1	Fractie constructor with two integer values	1p
G2	Fractie constructor with const char * value	5p
G3	Organize your project in 3 files: **main.cpp**, **Fractie.h** and **Fractie.cpp**	1p
G4	Organize your class **Fractie** to include private and public members, the definition of a constructor, and at least one method.	2p
G5	Operator- to subtract two fractions	3p
G6	Operator+ to add two fractions	3p
G7	Operator* to multiply two fractions	2p
G8	Operator== to check if two fraction are equal (in their simplified form)	2p
G9	Method: print that will print the fraction	2p
G10	Method: simplify that returns a simplified fraction	2p
G11	Operator[] to get the numerator or denominator	4p
G12	Program compiles and upon execution produces the expected results	3p