

Workshop 1: Objective C basics

The aim of this exercise is to get familiar with a new language called Objective C, which is an extension to the C programming language.

Task 1: Compile and run the Fractions program

Download [Fractions.zip](#) and unzip to your desktop. This file contains the source code for an example program called Fractions.

Open the “Terminal” and change to the directory containing the Fractions source code:

```
cd Desktop/Fractions
ls
```

To compile an Objective C program from the command line:

```
gcc -framework Cocoa -o frac Fraction.m main.m
```

To run the program:

```
./frac
```

Now look at the files. Right click main.m and ‘Open With’, ‘TextEdit’.

Task 2: Text output

If you want to calculate the decimal value of the fraction then you can use the float datatype:

```
float result = 1.0*topNumber / bottomNumber;
```

Edit the Fractions program by writing code that will change the output from:

```
The fraction is: 1/3
```

to:

```
The fraction is: 1/3 = 0.33333
```

Re-compile and run the program.

Task 3: Text input

Edit the Fractions program such that it prompts the user to enter two numbers (hint: use the scanf function). For example:

```
Enter the top number:
1
Enter the bottom number:
4
The fraction is: 1/4 = 0.25
```

Task 4: Creating methods

Add a new method to your Fraction class, called 'square'. First, you need to add the method to the header file (Fraction.h):

```
- (void) square;
```

Second, you need to implement the method in the implementation file (Fraction.m). The method should 'square' the fraction ($\text{fraction} = \text{fraction}^2$).

To test it works, add a line *to the main method* to square the fraction and then print the result.

Task 5: Methods with parameters

Now do the same for another method called 'powerOf':

```
- (void) powerOf: (int) power;
```

This method is the same as 'square' except that it has a parameter called 'power'. It should calculate the fraction to any power ($\text{fraction} = \text{fraction}^{\text{power}}$).

Test it!

Task 6: Creating classes

Create a new class called Calculator (you will need a header file and an implementation file: Calculator.h and Calculator.m). The Calculator class should contain:

a) Two Fraction variables (fraction1 and fraction2);

```
Fraction *fraction1;  
Fraction *fraction2;
```

b) Getter and setter methods (for fraction1 and fraction2);

```
- (Fraction *) fraction1;  
- (Fraction *) fraction2;  
- (void) setFraction1: (Fraction *) frac;  
- (void) setFraction2: (Fraction *) frac;
```

c) A method called 'add' that calculates the addition of fraction1 and fraction2, and returns the result as a new Fraction object.

```
- (Fraction *) add;
```

Write your header (.h) first, and then implement each method.

Does your new class compile?

Task 7: Instantiating objects

Now you have a new class, you must use it!

In the main method:

a) instantiate a new Calculator object

```
Calculator *calc = ...
```

b) instantiate two Fraction objects, and set them on the Calculator object

```
Fraction *frac1 = [[Fraction alloc] init];  
[frac1 setTopNumber:1];  
[frac1 setBottomNumber:2];  
[calc setFraction1:frac1];  
...
```

c) use the 'add' method on the Calculator object to print out the result of adding the two fractions.

```
Fraction *result = ...  
[result print];
```

Task 8: More practice

Can you add a multiply method to the Calculator class?