1. @ sign immediately precedes a constant **NSString** object, if no @ is specified, you are writing a constanct C-style string.
2. How to call methods defined in objective c.

myCar = [Car new];

[myCar getName];

[myCar setName:@”hello world”];

1. How to define one class

The @interface section describes the class and its methods, and the @implementation section describes the data (the instance variables that objects from the class will store) and contains the actual code that implements the methods declared in the interface section.

**@interface Fraction: NSObject //Parent class NSObject**

**-(void) print;**

**-(void) setNumerator: (int) n;**

**-(void) setDenominator: (int) m;**

**@end**

**// here – means it’s an instance method, + means it’s a class method**

**@implementation Fraction**

**{**

**int numerator;**

**int denominator;**

**}**

**-(void) print**

**{**

**NSLog(@”%i / %i ”, numerator, denominator);**

**}**

**-(void) setNumerator: (int) n**

**{**

**numerator = n;**

**}**

**-(void) setDenominator: (int) m**

**{**

**denominator = m;**

**}**

**@end**

When a method takes an argument, you also append a colon to the method name when referring to the method. Therefore, setNumerator: and setDenominator: is the correct way to identify these two methods, each of which takes a single argument.

1. Some datatype and qualifiers

Type: int float double char

Qualifiers: long, long long, short , signed, unsigned

With these, you can get long int, long long int, long double etc which will have greater range.

With these qualifiers, the output format will change also:

char %c

short char %sc

int %i, %x, %o

long int %li

long long int %lli

id %p

1. Type id is used to store an object of any type, so it’s a generic object type, which is also the **basis** for very important features in Objective-C known as **polymorphism** and **dynamic binding**.
2. Dd
3. Dd
4. Dd
5. D
6. D
7. D
8. D
9. Dd
10. D
11. D
12. D
13. D
14. d