**ASSIGNMENT 1 : PROTOCOLS AND SUBCLASSES**

**NAME : BSB MOUNICA**

**USN : 1PI10IS022**

**CLASS : 7th Semester, ‘A’ sec**

**PROTOCOLS**

Protocols declare methods that can be implemented by any class. They are useful in 3 main situations:

1. To declare methods that others are expected to implement
2. To declare the interface to an object while concealing its class
3. To capture similarities among classes that are not hierarchically related

* Protocols work like interfaces in Java.
* We can define protocol with some methods and then implement it in many classes.
* Each of these classes could have different implementation of methods provided by protocol, but the interface would be the same.

**Eg**: If we want to calculate area of different shapes, like square and circle. We can put calculatearea method definition in protocol and provide different implementations in these two classes. Then you can cast each class instance to protocol type and invoke given method receiving appropriate results.

**Syntax:**

@protocol MyProtocol

- (void)myProtocolMethod;

@end

**SUBCLASSES**

* Every object we create in your Cocoa application descends from the 'NSObject' foundation class.
* The NSObject class identifies properties and methods which apply to all objects.
* The NSObject class is divided into smaller groups of objects, called subclasses.
* Objects in these subclasses not only conform to the protocol of NSObject, they are also defined more precisely by the methods that govern their subclass.

**Syntax:**

*Say we have a class:*

@interface MyClass: NSObject

@property (nonatomic, strong) NSArray \* myarray;

@end

*And we want to create a subClass of MyClass,*

@interface Mysubclass : MyClass {

// instance variables that Mysubclass has but Myclass lacks go here

}

* A subclass can override a method it inherits from the class it is based on.
* The class it is based on is called its superclass.
* Subclassing is just a subclass of another class and it inherits behaviour and members of its parent class.

***Basic definition of a class with protocols and subclasses***

struct objc\_class

{

    Class isa;

    Class super\_class; //Classes contain pointers to their superclass but contain no pointer to subclasses

    const char \*name;

    long version;

    long info;

    long instance\_size;

    struct objc\_ivar\_list \*ivars;

    struct objc\_method\_list \*\*methodLists;

    struct objc\_cache \*cache;

    struct objc\_protocol\_list \*protocols;

};

Hence the difference.