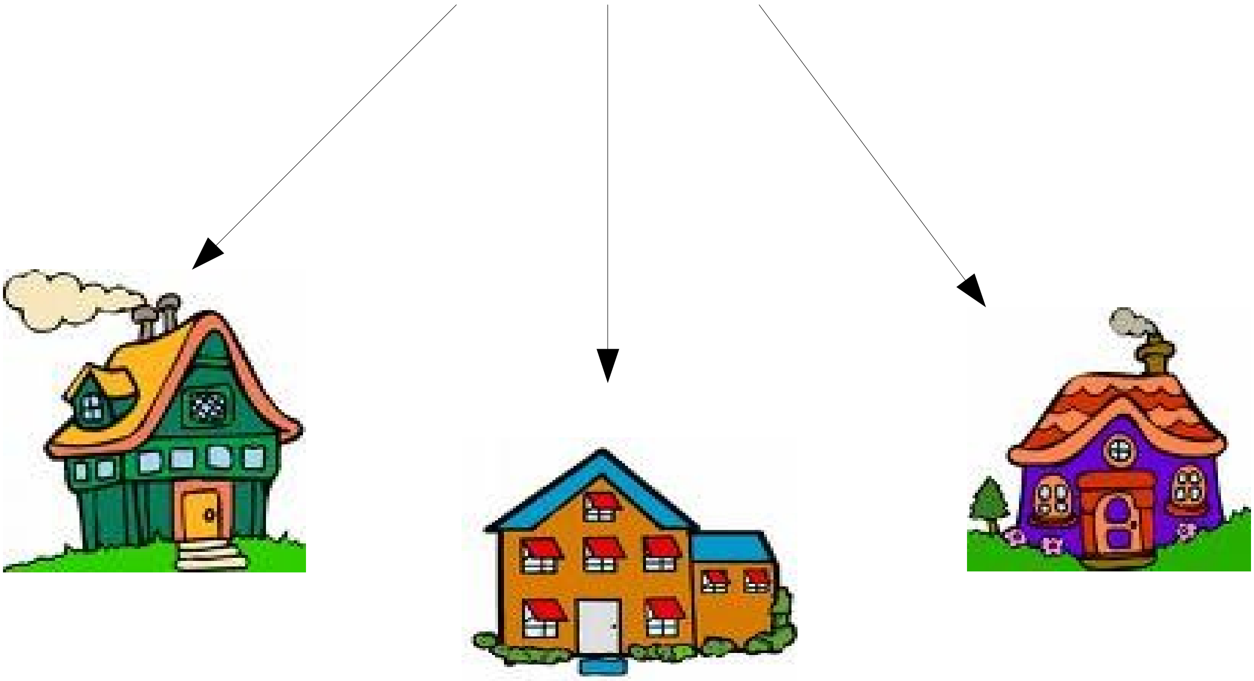
## INHERITANCE

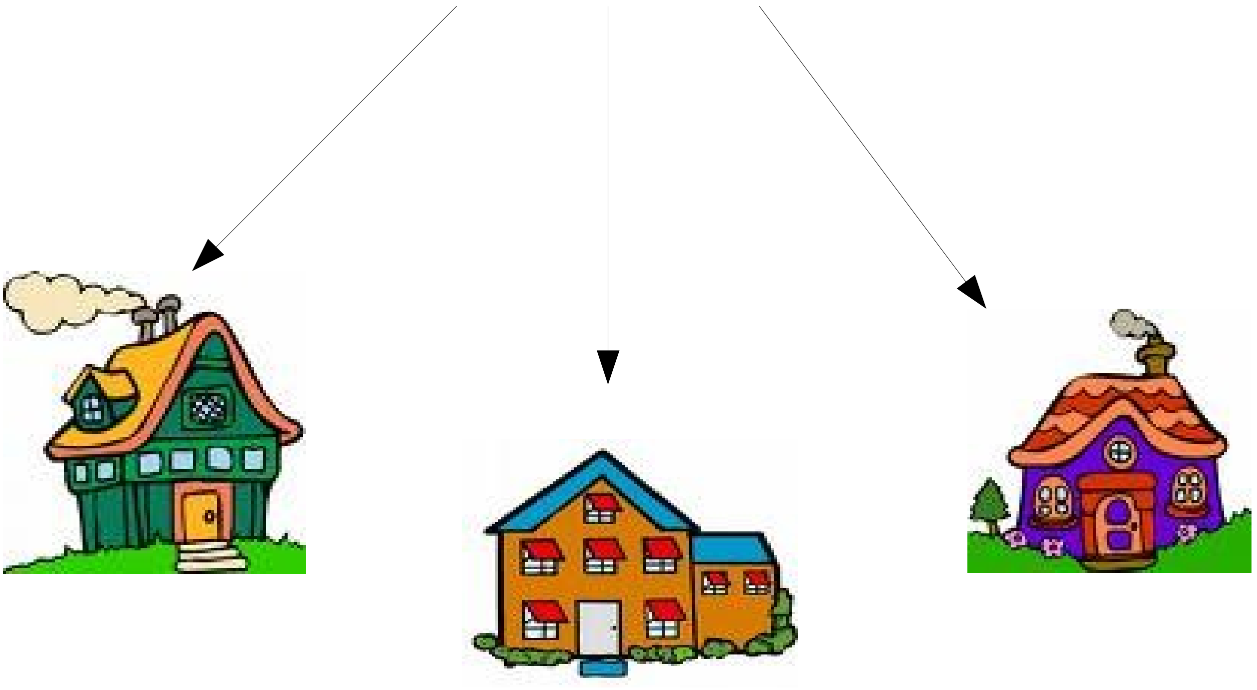
A class is like a house blue print



These 3 houses are derived from the original blueprint, so in object oriented programming terms can be thought of a derived classes, or sub-class of the base class/original blueprint.

# INHERITANCE

The base class (or blue print) has predefined data types also called attributes

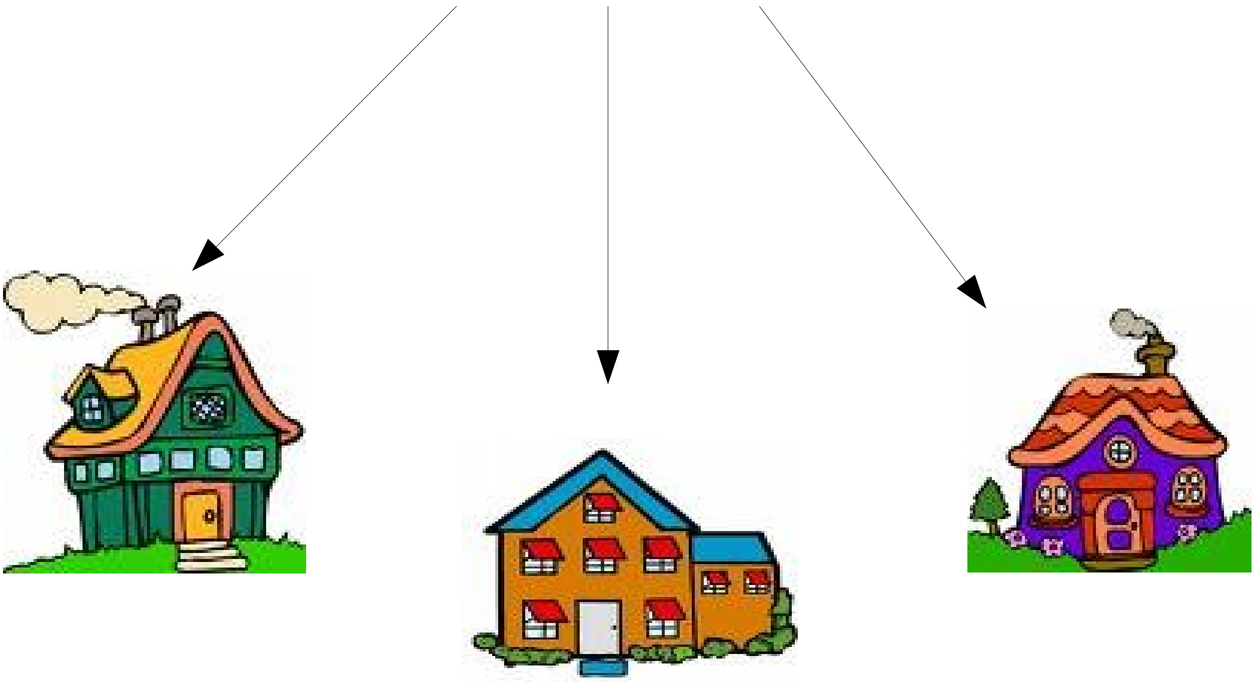


Each of these house (or sub-classes/child classes) inherit the attributes from the base class (blue print)

For example the blue print will have width and height attributes. These are float data types. Each derived house/sub-class can then change the value of width and height to suit the new house/object.

## INHERITANCE

The base class / blue print also has embedded functionality called behaviours.

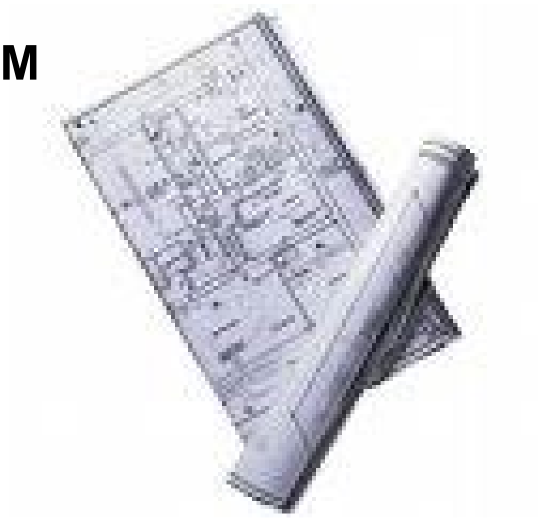


The derived houses / sub-classes also inherit these behaviours.  An example of a behaviour analogeous to the blue print could be internal heating.

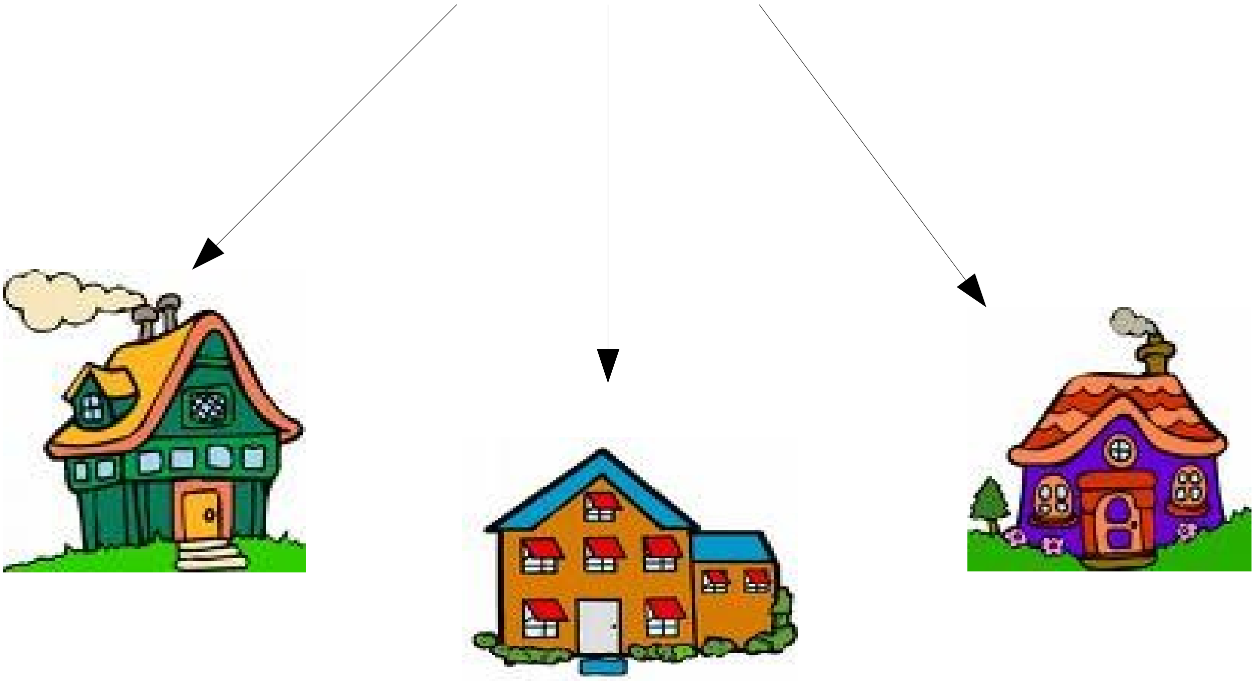
All houses have it but each one can heat the house in individual ways such as ducted, solar and insulation.

This is called inheritanceand is one of the features of object oriented programming to allow for code re-use with little or no modification.

## POLYMORPHISM



The blue print base class has a behaviour or method called MailBox()

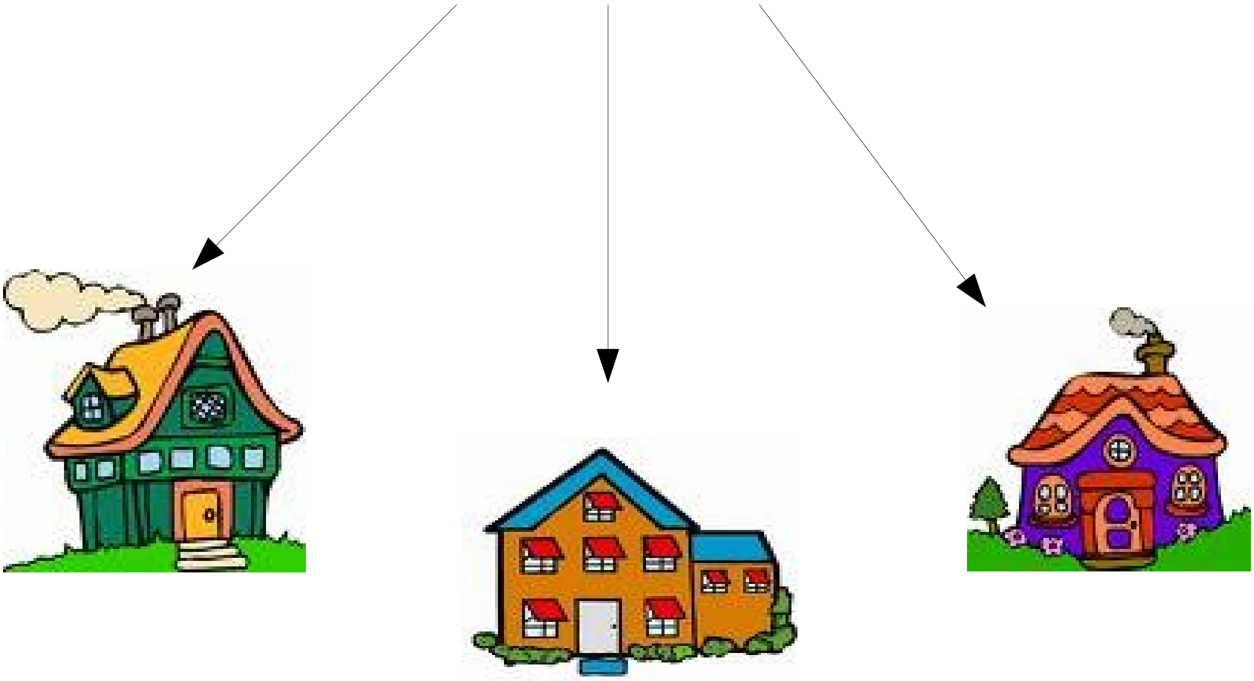


Each of the houses inherits the MailBox() behaviour but in house number one it refers to an email account, house number two is a vaccuum tube mail system and house number three uses a traditional mail box.

But we don't care about this. All we want to do is post a letter so we can use this ability of classes called Polymorphism to call House.Post() and each house will deal with the letter how it sees fit to do so.

## ENCAPSULATION

The blue print base class has a behaviour or method called MailBox()



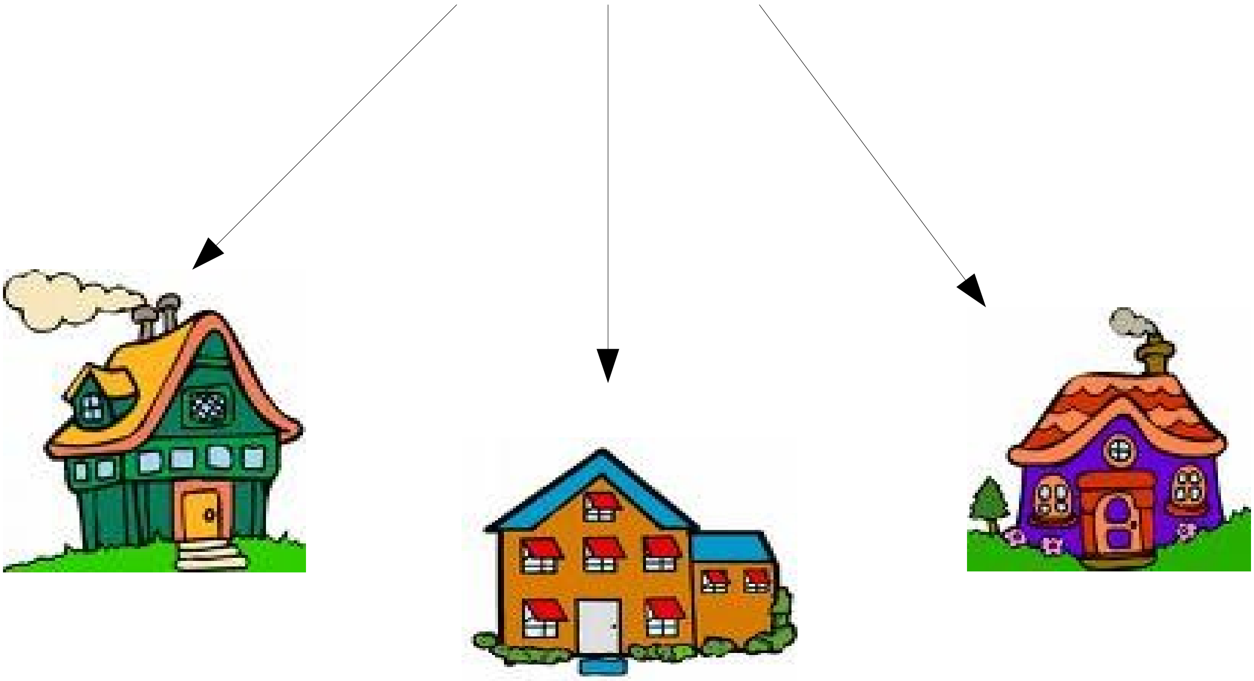
Because we don't care how the blue print or derived house sub-class does it, we can just call House.Post() we call this encapsulation.

It means we don't need to know the internal workings of the class to still be able to use it.

It also means in future if the Post() behaviour changes, such as using Holographic Mail our calling code doesn't break when the original programmers add the Holographic Mail functionality.

# ABSTRACTION

The blue print base class is a great example of abstracting a problem to a higher level.



The program could call for houses to be built in a virtual city.  We don't care how they work under the hood, only that we can create as many houses as memory will allow.

Abstraction is converting the problem into a human intelligable format that helps solve problems as the appropriate level.