OOPS

Programming tech --> set of rules

Classes and Objects

classes --> its a blueprint, it defines behav and charac of a object

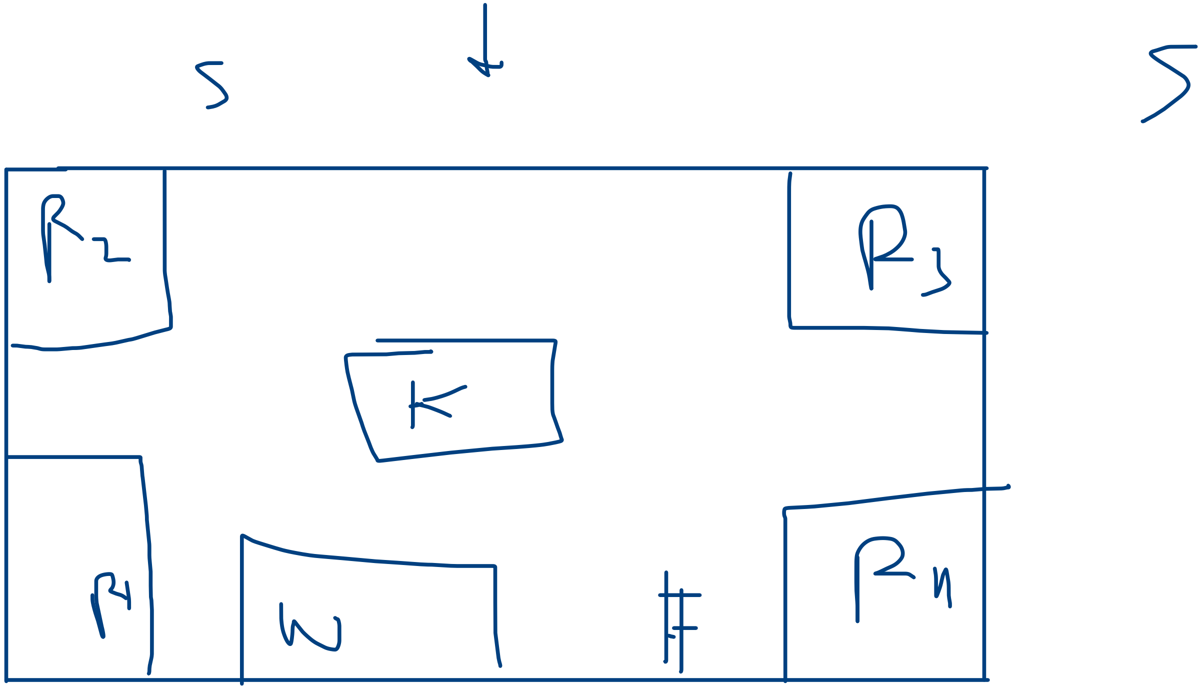
Syntax --> class

Objects --> are real world entity which is defined from a class.

Builder construction

House class --> BluePrint

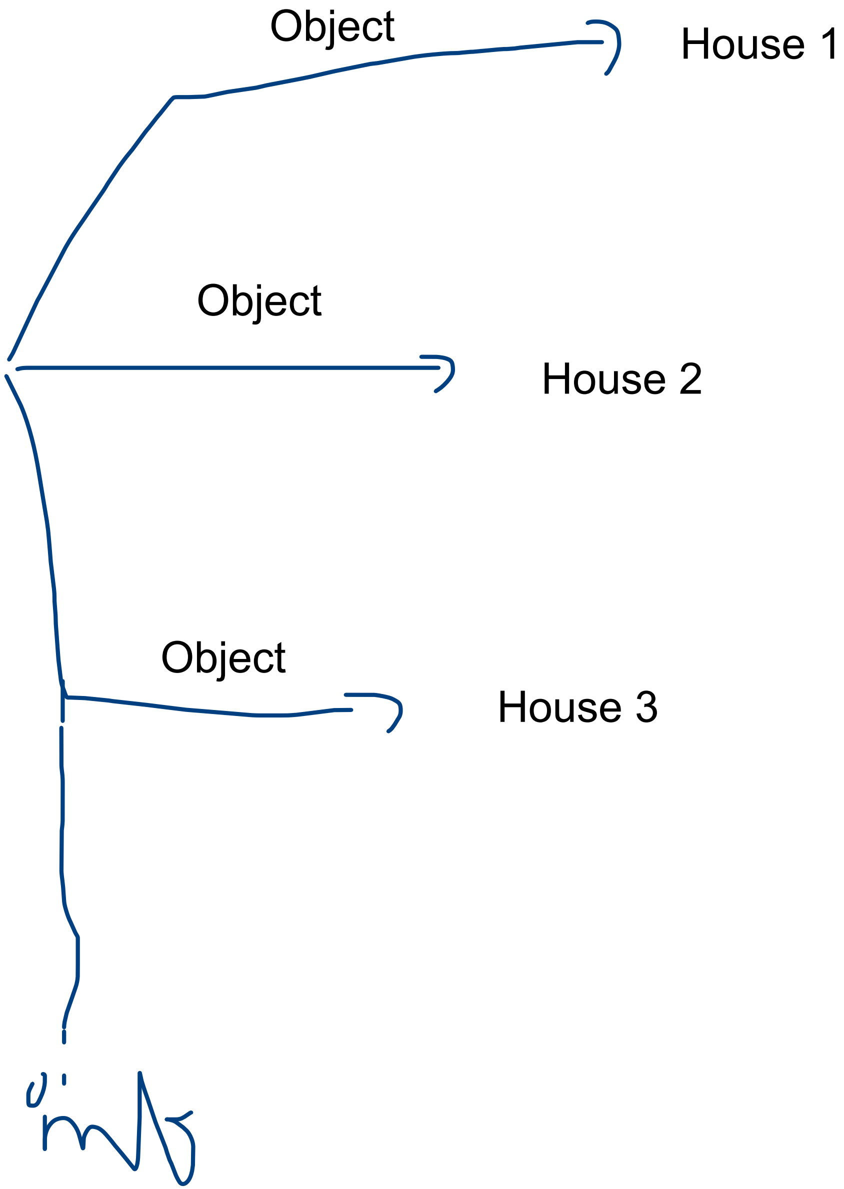
1. Num of rooms
2. Num of Kitchens
3. isStair
4. Num Of washrooms
5. Address



House
Layout

House class --> BluePrint

1. Num of rooms
2. Num of Kitchens
3. isStair
4. Num Of washrooms
5. Address



Object

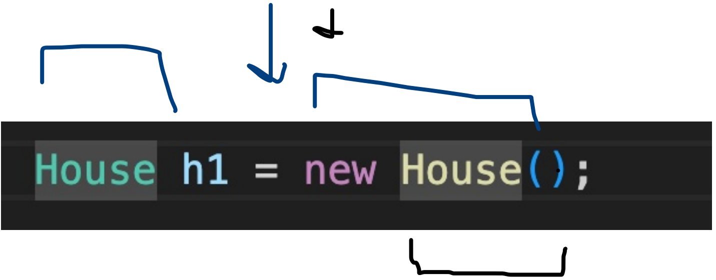
House 1

Object

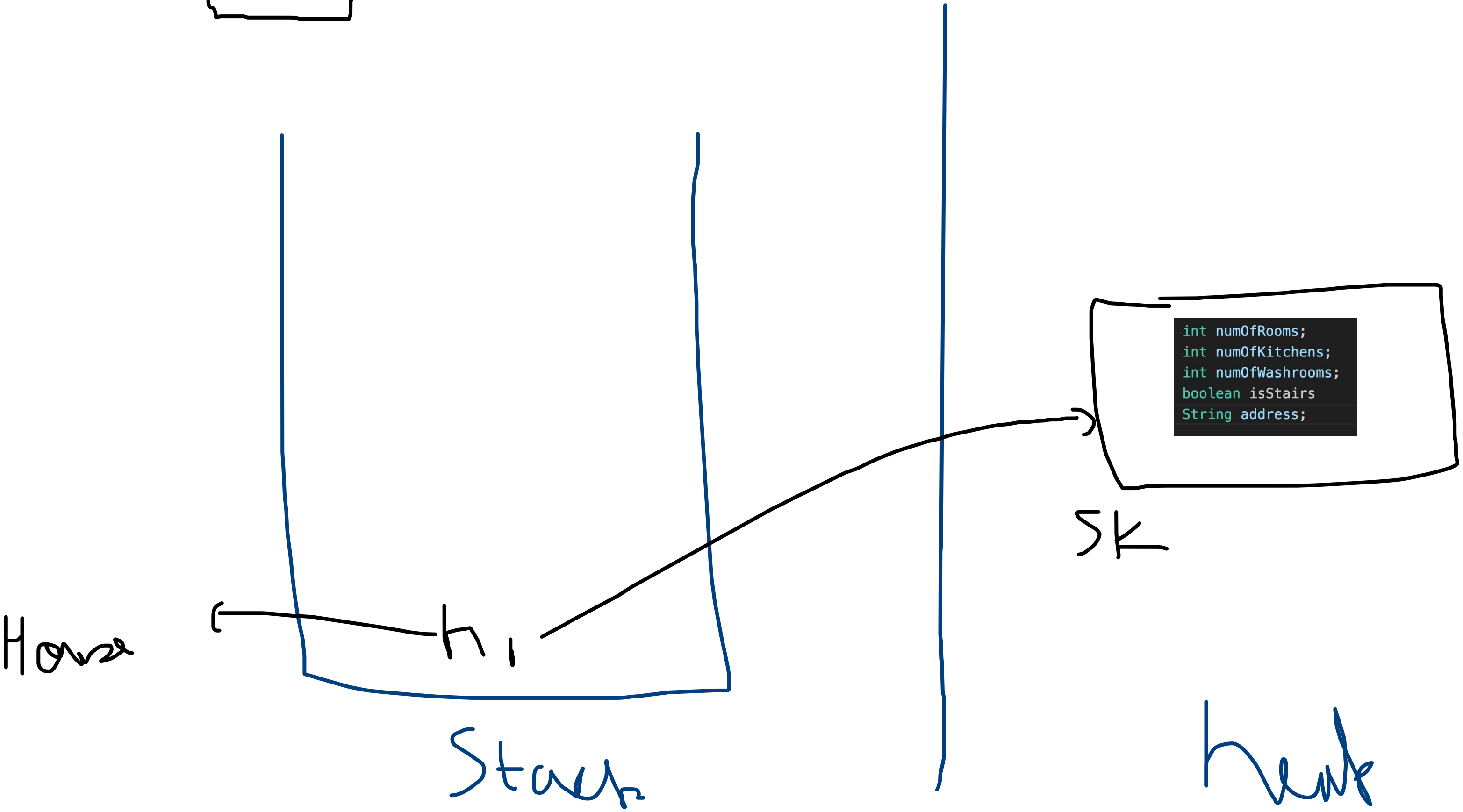
House 2

Object

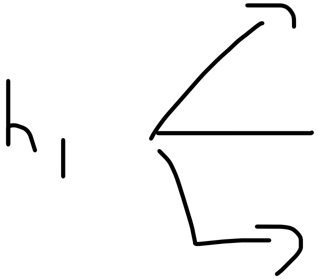
House 3

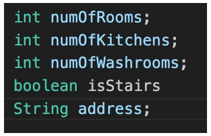


```java
House h1 = new House();
```

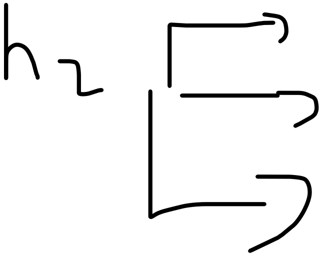


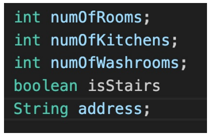
```java
int numOfRooms;
int numOfKitchens;
int numOfWashrooms;
boolean isStairs
String address;
```



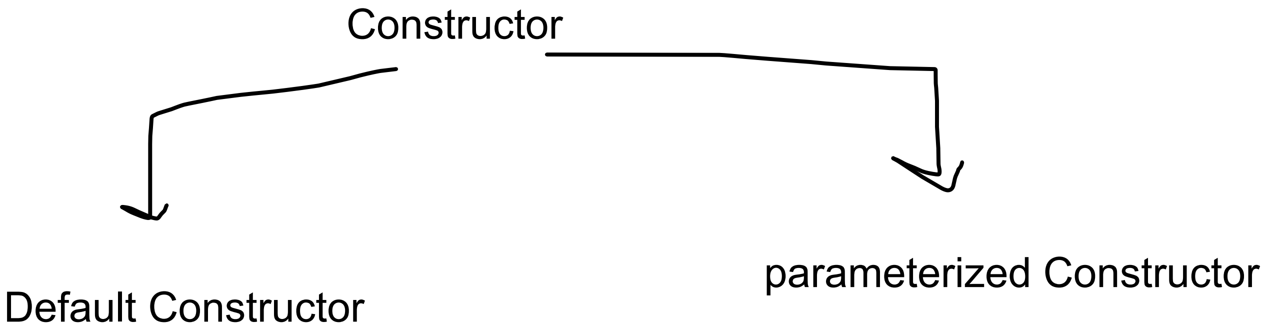


```
int numOfRooms;
int numOfKitchens;
int numOfWashrooms;
boolean isStairs
String address;
```





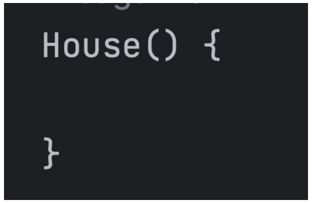
```
int numOfRooms;
int numOfKitchens;
int numOfWashrooms;
boolean isStairs
String address;
```



Constructor

Default Constructor

parameterized Constructor



```
House() {

}
```

Getters and Setters Methods

getters -- > get the values of instance variables.

setters --> to set the values of instance variables

namingConvention

instance variable = numOfRooms

getter method --> getNumOfRooms()
setter method --> setNumOfRooms(int numOfRooms)

Access Modifiers --> applied on Methods, Variables and class

Default

Public —

Private —

Protected

✓ Full Syntax of Variable --> AccessModifier Static/NonStatic DataType nameOfVar

✓ Full Syntax of Method --> AccessModifier Static/NonStatic ReturnType nameOfMethod()

✓ Full Syntax of class --> AccessModifier Static/NonStatic class Class public --> Full Access to internal or external of present class

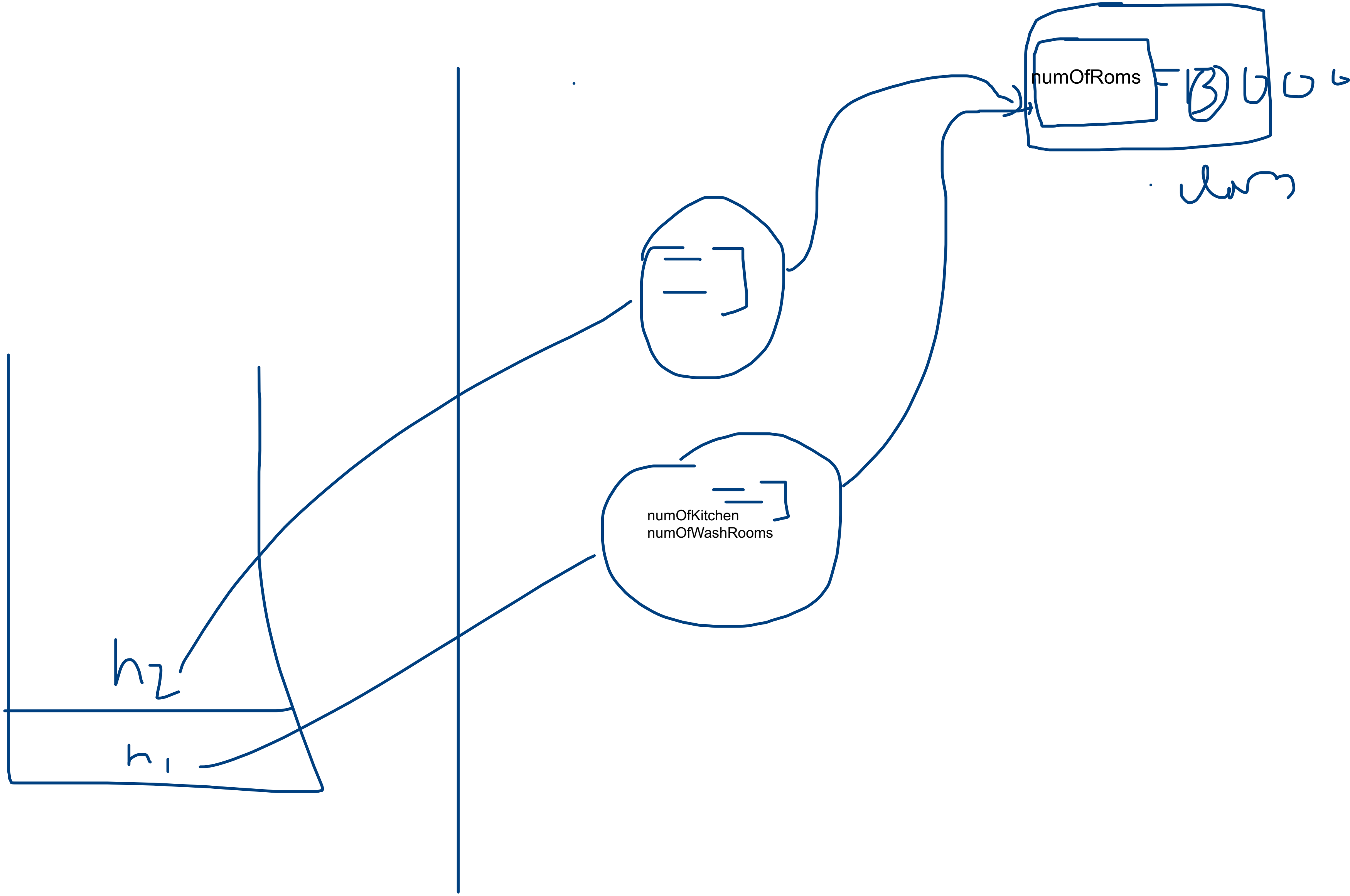
private --> Only access in internal of class, not external

default --> access is internal as well as in same package in external class

Static Keyword

Access --> is always happens at same level

static keyword --> making class, method or variable as class level


numOfRoms
numOfKitchen
numOfWashRooms
h2
h1

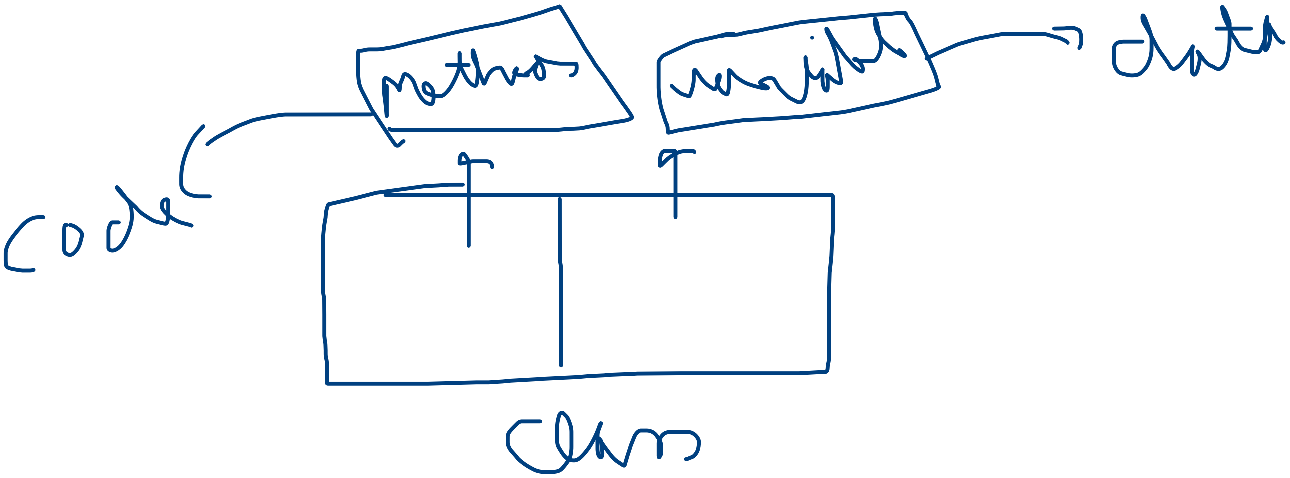
Pillars of OOPS (Rules)

1. Encapsulation
2. Inheritance
3. Polymorphism
4. Abstraction

Advance of Abstraction --> interface

Encapsulation

Encapsulation says, binding your data and code together. So that your data is completly hidden from external classes and can be access to external class only via methods ie, code



✓ Your data is protected.

✓ It is a restriction to access the data of one object in another class

Default →

Public

Private

Protected