Abstract class

we can provide both Abstract methods & non abstract methods.

↓

Not providig Implementation

* Subclass will provide the implementation

Abstract class (Why?)

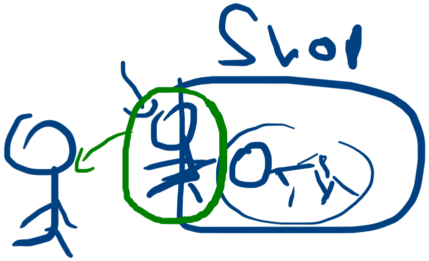
- * We can not Create Instances of the class.

- * Sub class will provide the implemention of abstract Methods

- * partially Implemented class.

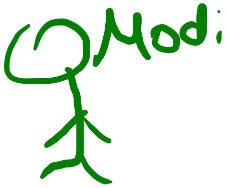
- * Abstract class can contain abstract methods or non Abstract method.

1. give food.
2. walk
3. bathing.

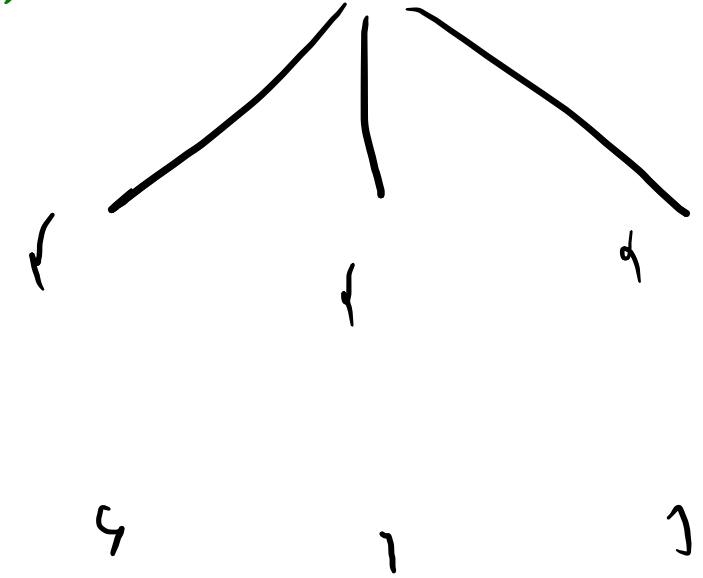


1. food. __
2. thing -

Abstract class



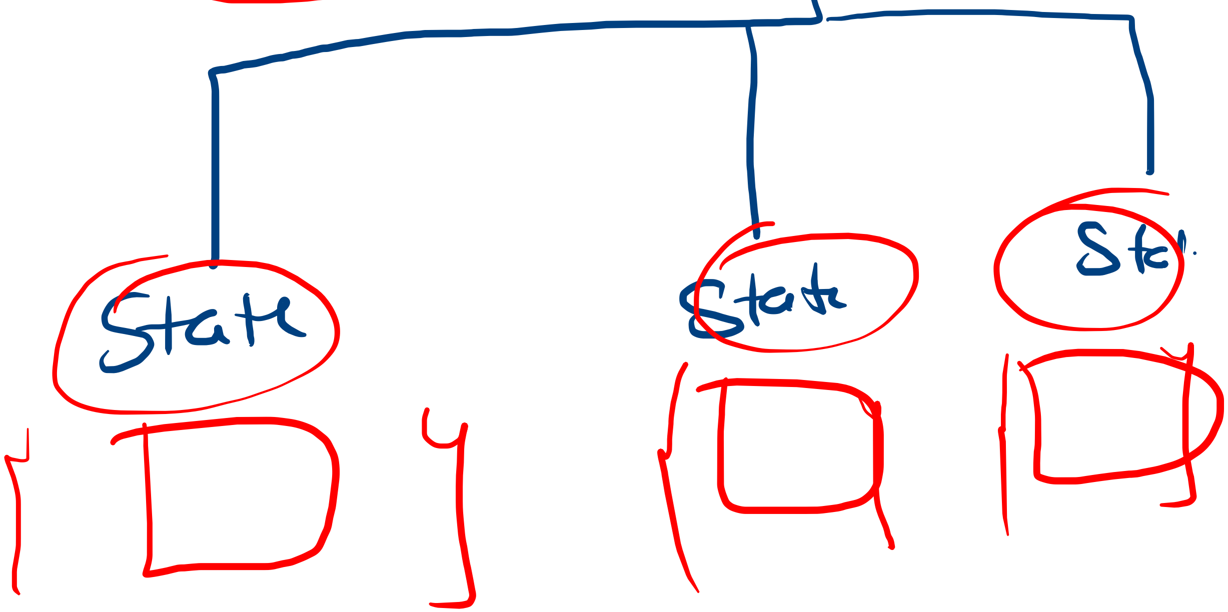
abstract Stand for NationalAtt( )

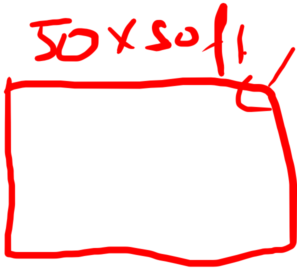


Modi decided.

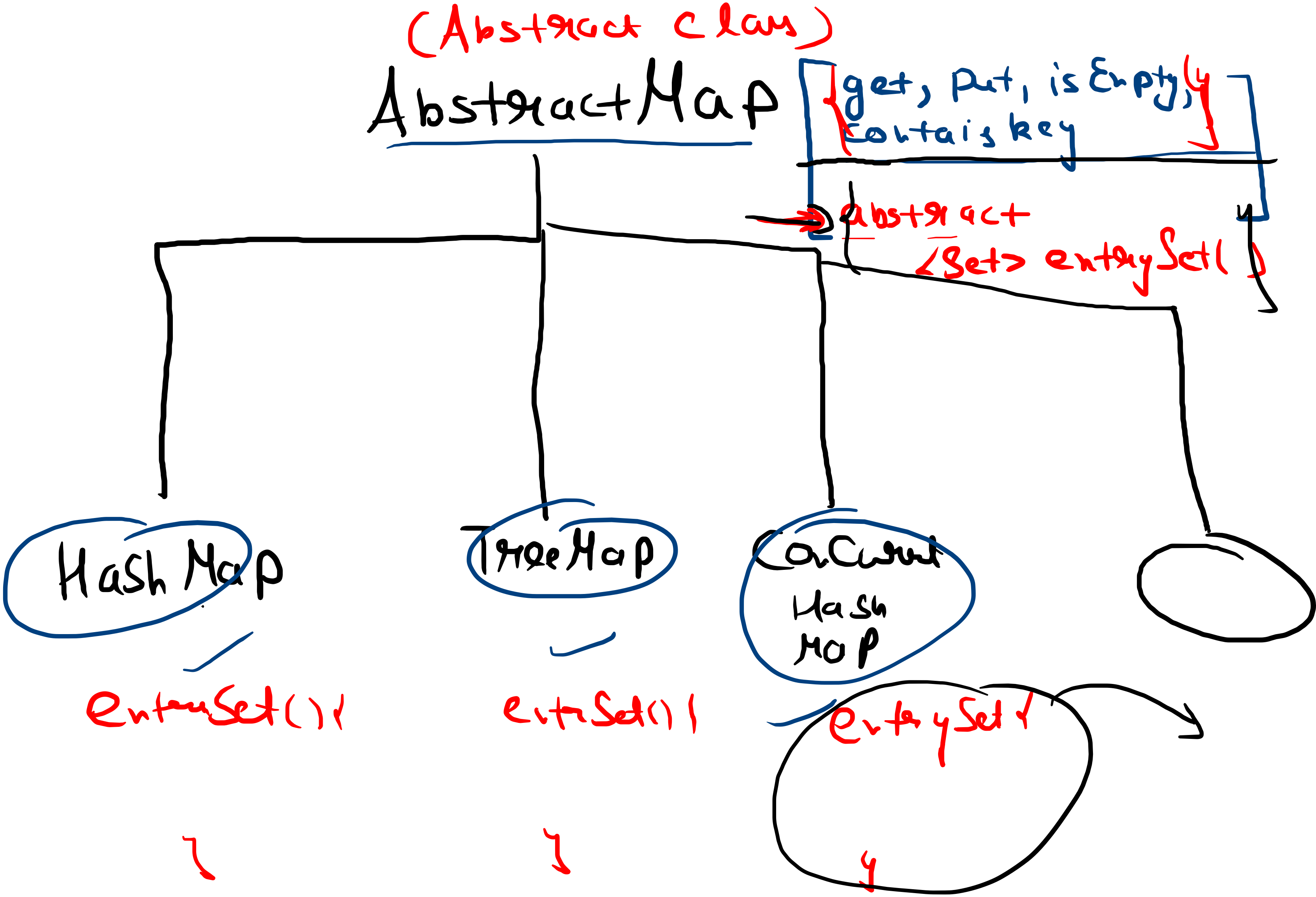
(swach bharat)

abstract (harghar sochla)



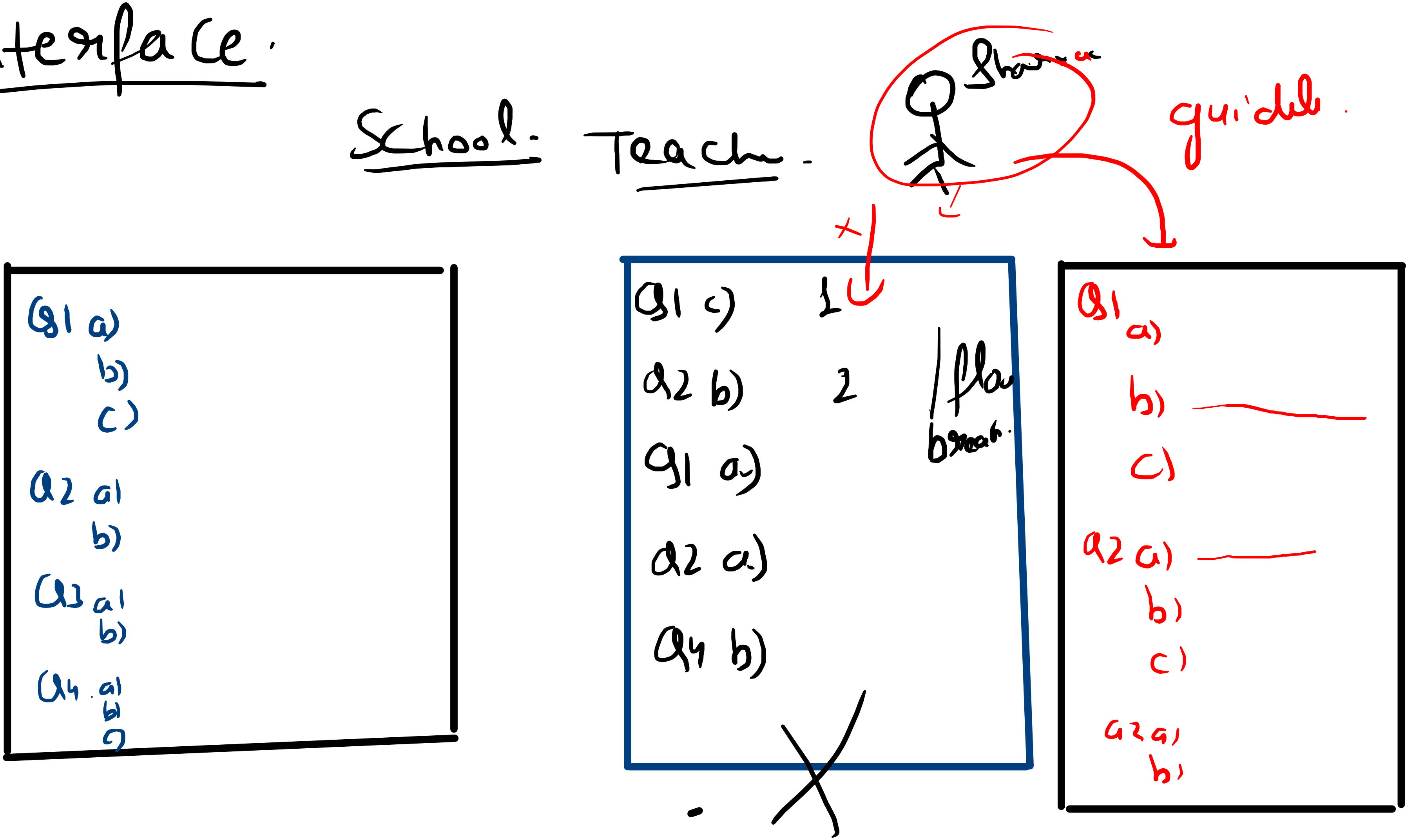


Jav.

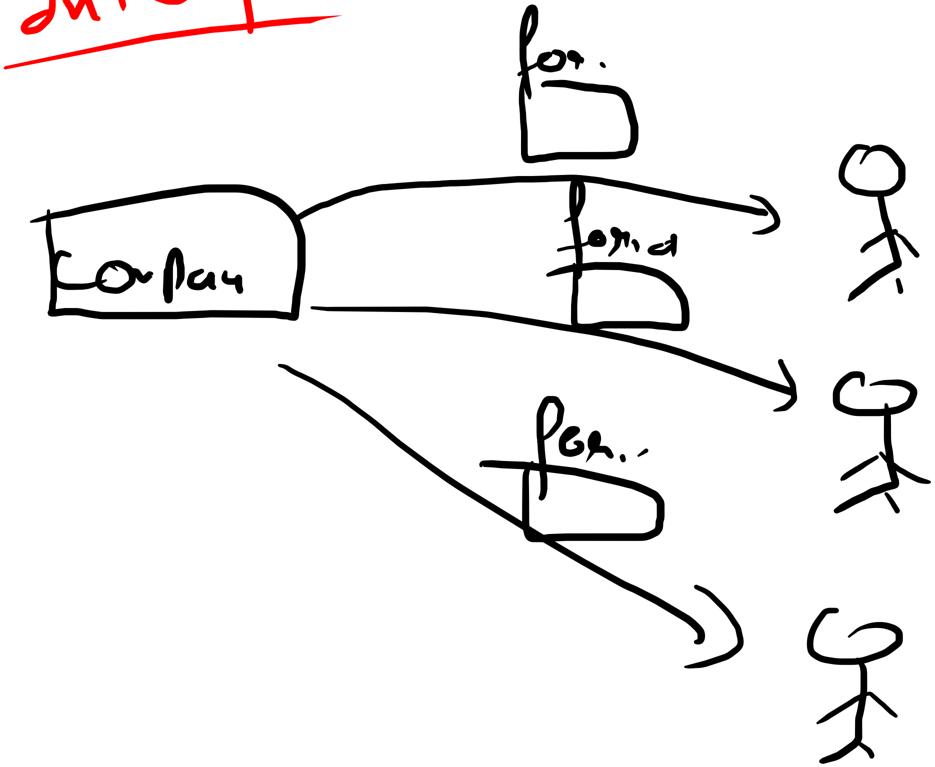

(Abstract class)
AbstractMap
{get, put, isEmpty(){ containskey }
abstract <Set> entrySet( ){ }
HashMap
entrySet(){ }
TreeMap
entrySet(){ }
ConCurrent HashMap
entrySet{ }

Interface.

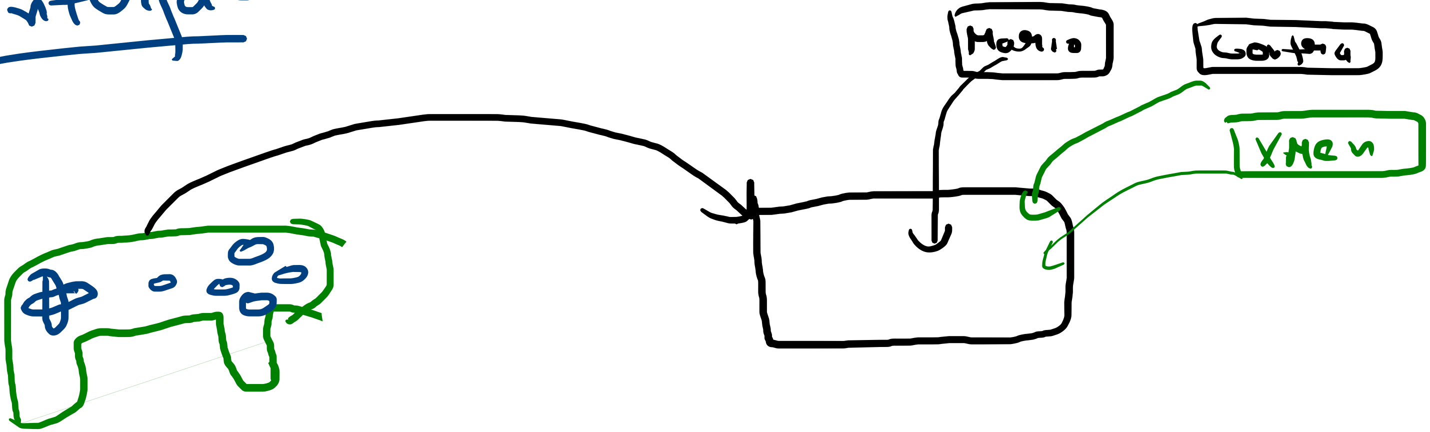
School: Teacher.



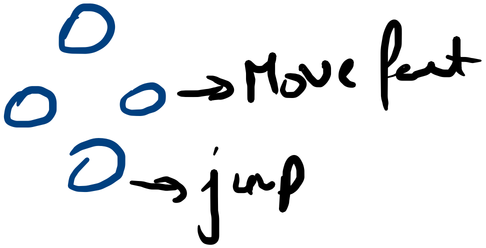
Interface



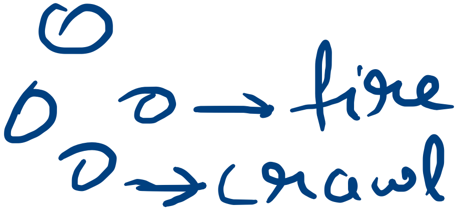
Interface



Mario

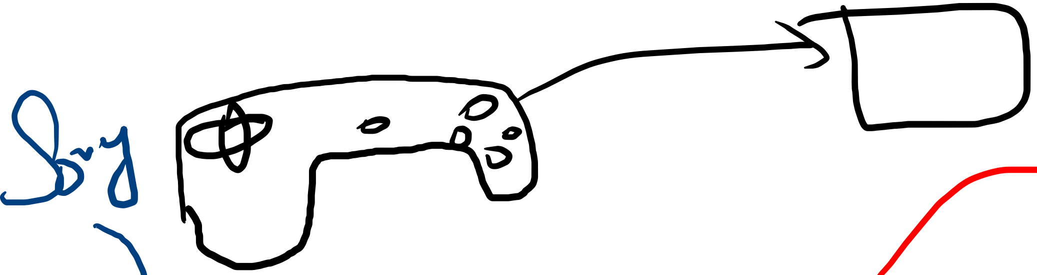


Contra



Interface

↓



```
public interface controller{

    public void up();
    public void down();
    public void right();
      "          "    left();

}
```

Game company

```
public class Mario implements
                      controller

{

  @override
    public void up(){
        -
         -

      }
  @override
    public void down(){

      }

}
```

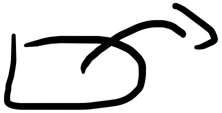
```
Controller game1 = new Mario( );
{
    game1.up( );
    game1.down( );
    gam 1. left( );
    ga- 1. right( );
}

Controller game1 = new Contra( );
{
    gan1. up( );
    gan 1. down ( );
    gan 1. left ( );
    ga 1 . right ( );
}
```

new Mario()	new Contra()	new spiderMan()

```
Controller c ;
    for i in arr ;
        c = i
        c.up()
        c.down()
        c.get()
        c.right...
```

```
Interface Test {
    void binarySearch(arr[], k)
}

class fw15_20 implements Test
{
    void binarySearch(arr[], k)
        {
            -  -
        }
}
```

new fwo_ls()	new fwo_ls()	_ _ _

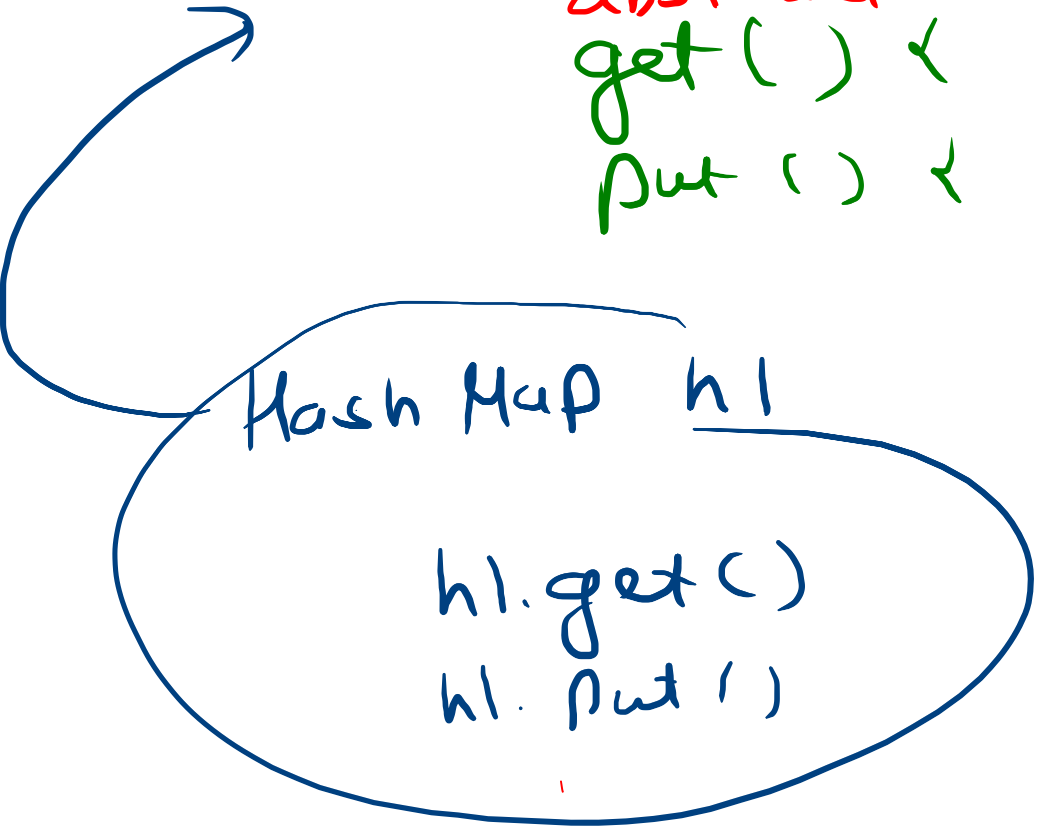


```
Test t ;

    for i in arr ;

        t = i ;

    output = t.binarySearch()
        if (output == 4)
            { print ("Passed");
            }
```

```
Mario m = new Mario();

   m.up()
```

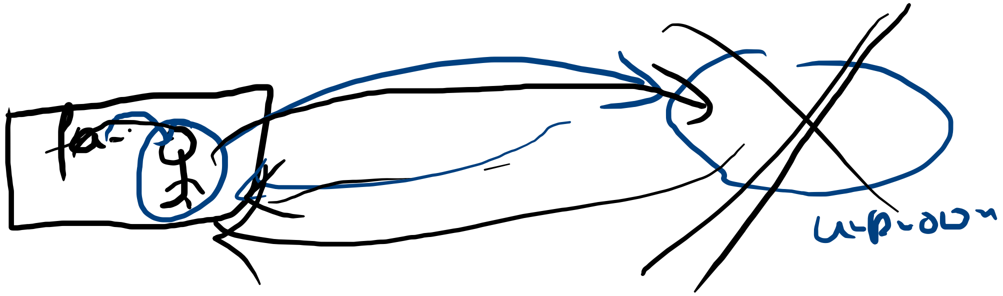
Abstract class x not create Insta

```
AbstractMap
    abstract dai.k() ;
    get ( ) {        }
    put ( ) {        }
```



```
HashMap h1

    h1.get()
    h1.put()
```

Coupling. — How much a class is dependent on other classes.



Coupling

- * Measure How much a class is dependent on other class

- * There should be Minimal dependency b/w classes.

- * Aim for low coupling

Cohesion

Strong your relation in the family.

how related the responsibilities of class are.

* High Cohesion