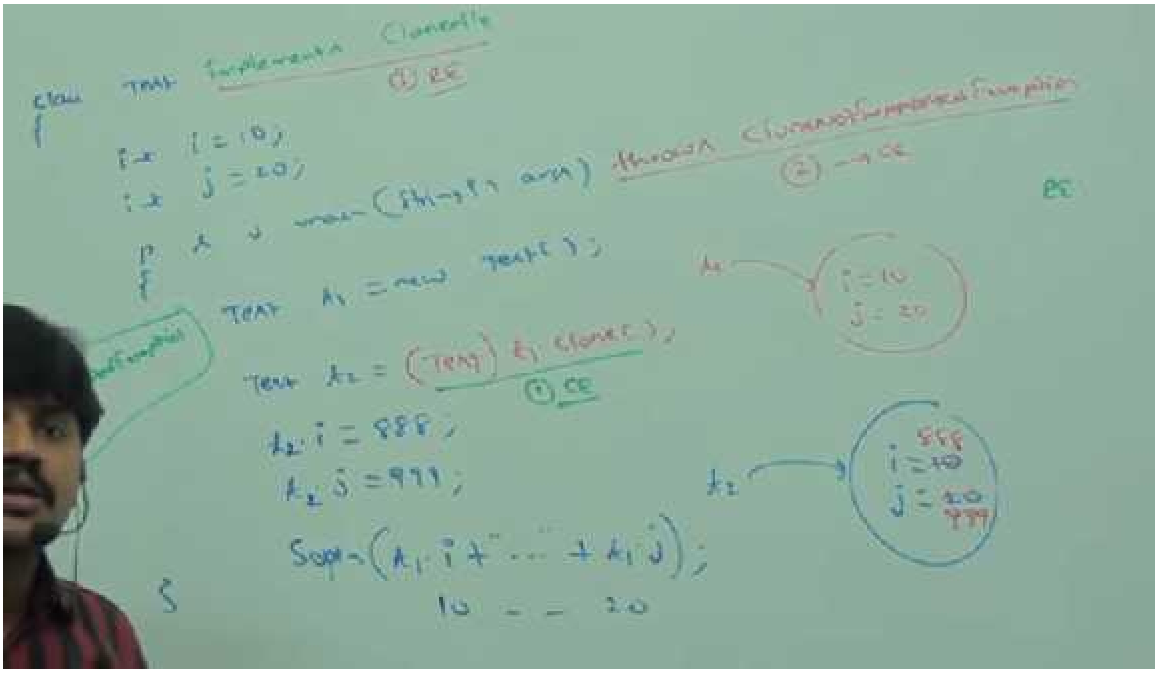
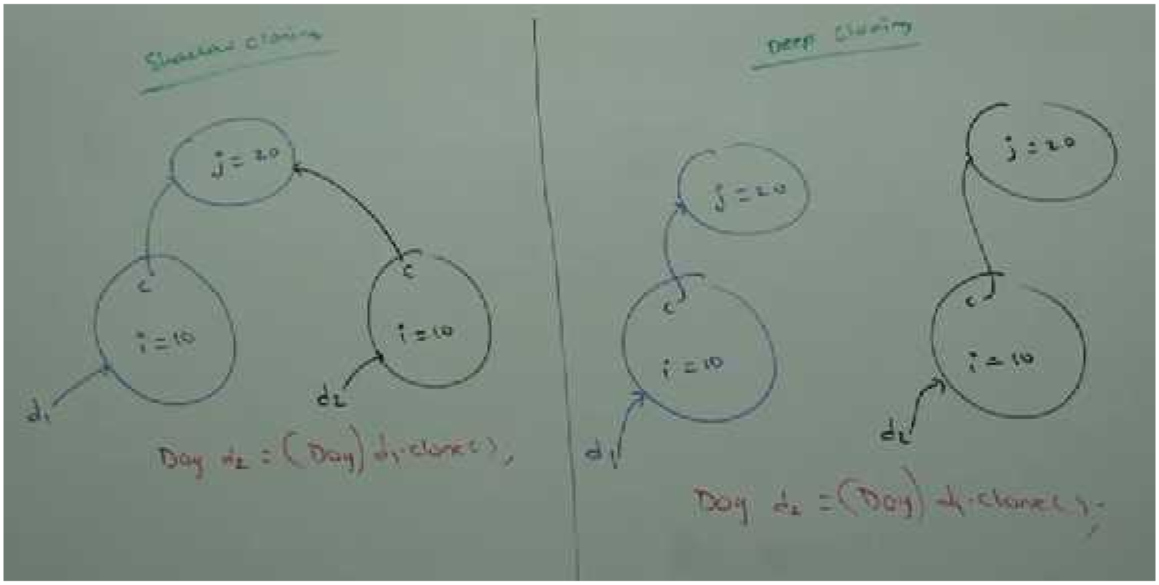
In Object class we have following method,

protected native Object Clone() throws CloneNotSupportedException



```
class Test implements Cloneable
{
  int i = 10;
  int j = 20;
  p s v main(String[] args) throws CloneNotSupportedException
  {
    Test t1 = new Test();
    Test t2 = (Test) t1.clone();
    t2.i = 888;
    t2.j = 999;
    Sopln(t1.i + "...." + t1.j);
         10 -- 20
  }
```



```
Dog d2 = (Dog) d1.clone();
```

```
Dog d2 = (Dog) d1.clone();
```

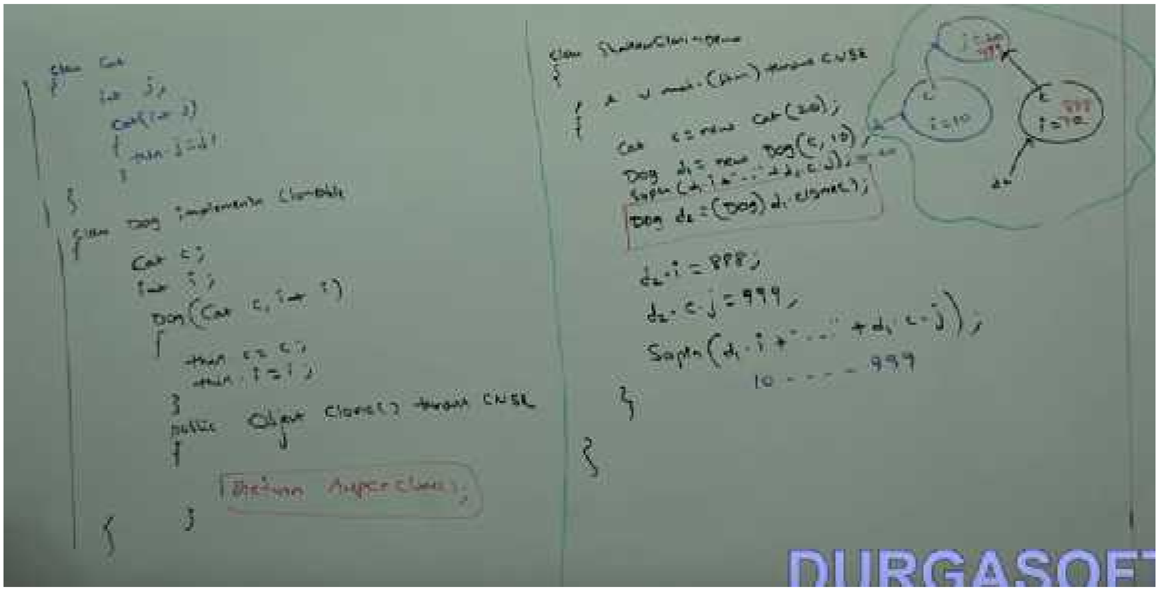
Shallow Clonning:

The process of creating bitwise copy of an Object is called shallow clonning.

If the main object contins premitive variables then exactly duplicate copies will be created in the cloned object.

If the main object contains any reference variable then corrosponding object won't be created just duplicate reference variable will be created pointing to old contained object.

Object class clone method ment for shallow clonning.



```
class Cat
{
    int j;
    Cat(int j)
    { this.j=j;
    }
}
class Dog implements Cloneable
{
    Cat c;
    int i;
    Dog(Cat c, int i)
    {
        this.c=c;
        this.i=i;
    }
    public Object clone() throws CNSE
    {
        return super.clone();
    }
}
```

```
class ShallowCloneDemo
{
    p s v main(S[] a) throws CNSE
    {
        Cat c=new Cat(20);
        Dog d1=new Dog(c,10);
        Sopln(d1.i+"...."+d1.c.j);
        Dog d2=(Dog)d1.clone();
        d2.i=888;
        d2.c.j=999;
        Sopln(d1.i+"...."+d1.c.j);
            10 ---- 999
    }
}
```

In shallow clonning by using clonned object reference if we perform any change to the contained object then those changes will be reflected to the main object.

The overcome this problem we should go for deep clonning.

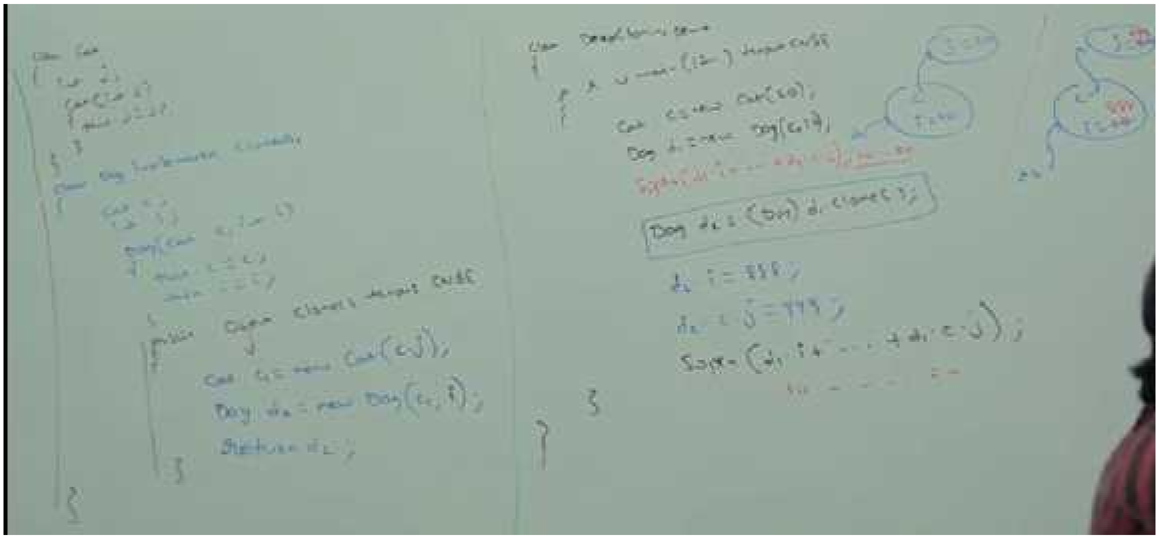
Deep Clonning:

The process of creating exactly duplicate independent copy including contained object is called deep clonning.

In deep clonning if the main object contain any premitive variables then in the clonned object duplicate copies will be created.

If the main object contains any reference variable then the corrosponding contained objects also will be crated in the cloned copy.

By default object class clone method ment for shallow clonning but we can implement deep clonning expicitely by overriding clone(), method in our class.



By using clonned Object reference if we perform any change to the contained object then those changes won't be reflected to the main object.

Which clonning is best ?

if object contains only premitive variables then shallow clonning is the best choice.

If object contains reference variables then deep clonning is the best choice.