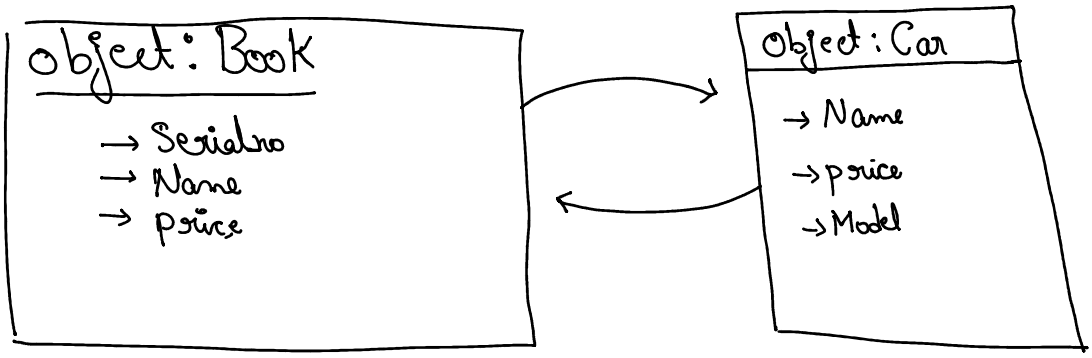
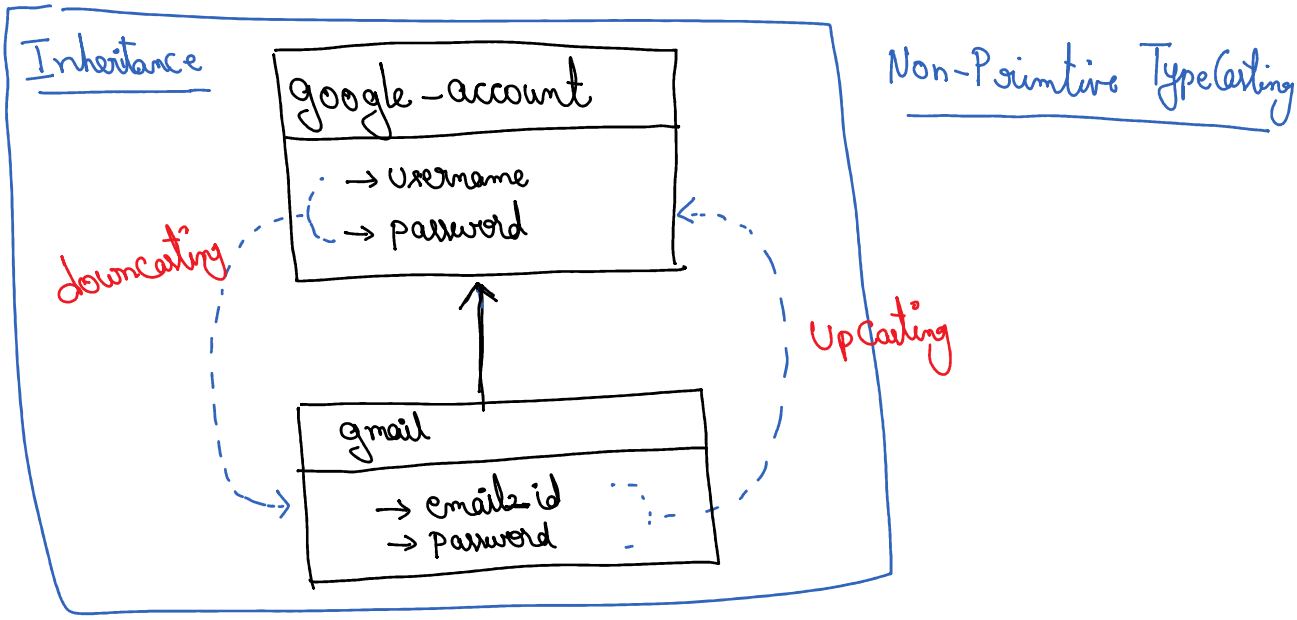
Non primitive type casting

In java, classes are called non primitive datatypes but we can't convert every class to another class type.

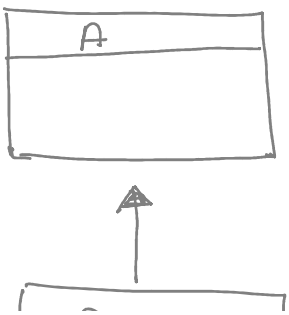
To perform non primitive type casting we must and should have inheritance being performed between classes





i) UpCasting (the process of saving a child instance in parent instance variable object)

```
class A{
  ---
}
class B extends A{
  ---
}
```



B obj = new B(); // child instance → obj ⇒ B

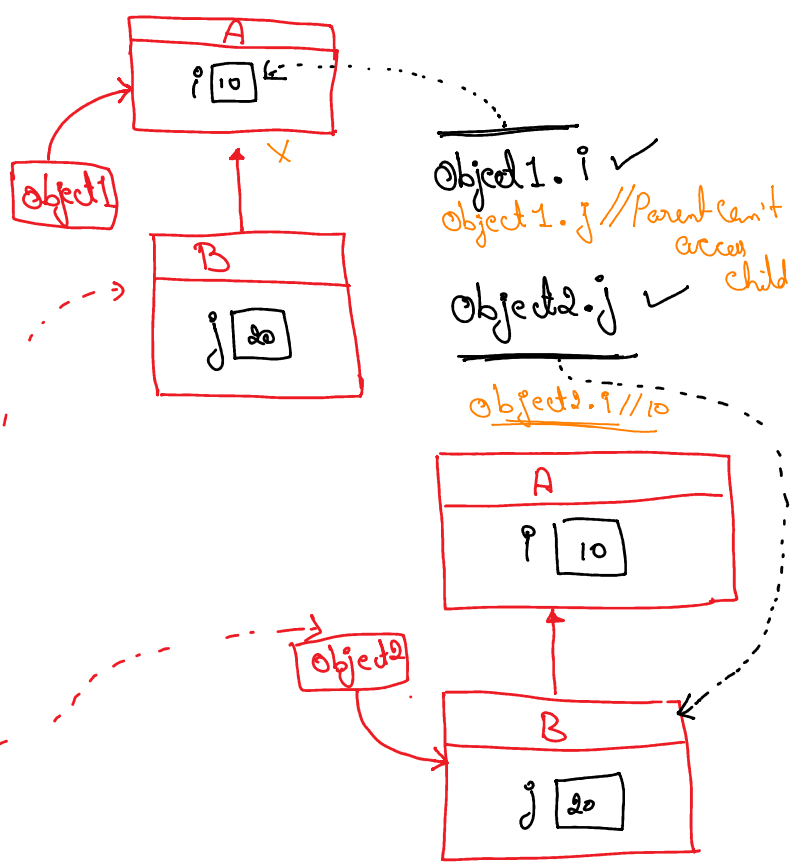
↳ instance variable of child having

A obj2 = new B(); // child instance → obj2 ⇒ A (Parent) / B (child)

↳ instance variable of parent

```
class A {
    int i=10;
}
class B extends A {
    int j=20;
}
main() {
    A object1; //Parent object
    // use a parent object to print i
    object1 = new B();
    // use a child object to print j
    B object2; //child object
    object2 = new B();
}
```

Upcasting



Down Casting [explicit]

↳ convert a parent type object into a child type

```
class A {
    ...
```

→ i] B obj = new A()

child object — Parent instance

```
class A {
    : : :
}
class B extends A {
    : : :
}
```

child object — Parent instance

// Incompatible type C.T.E

ii) `B obj = (B) new A();`

child object — parent instance

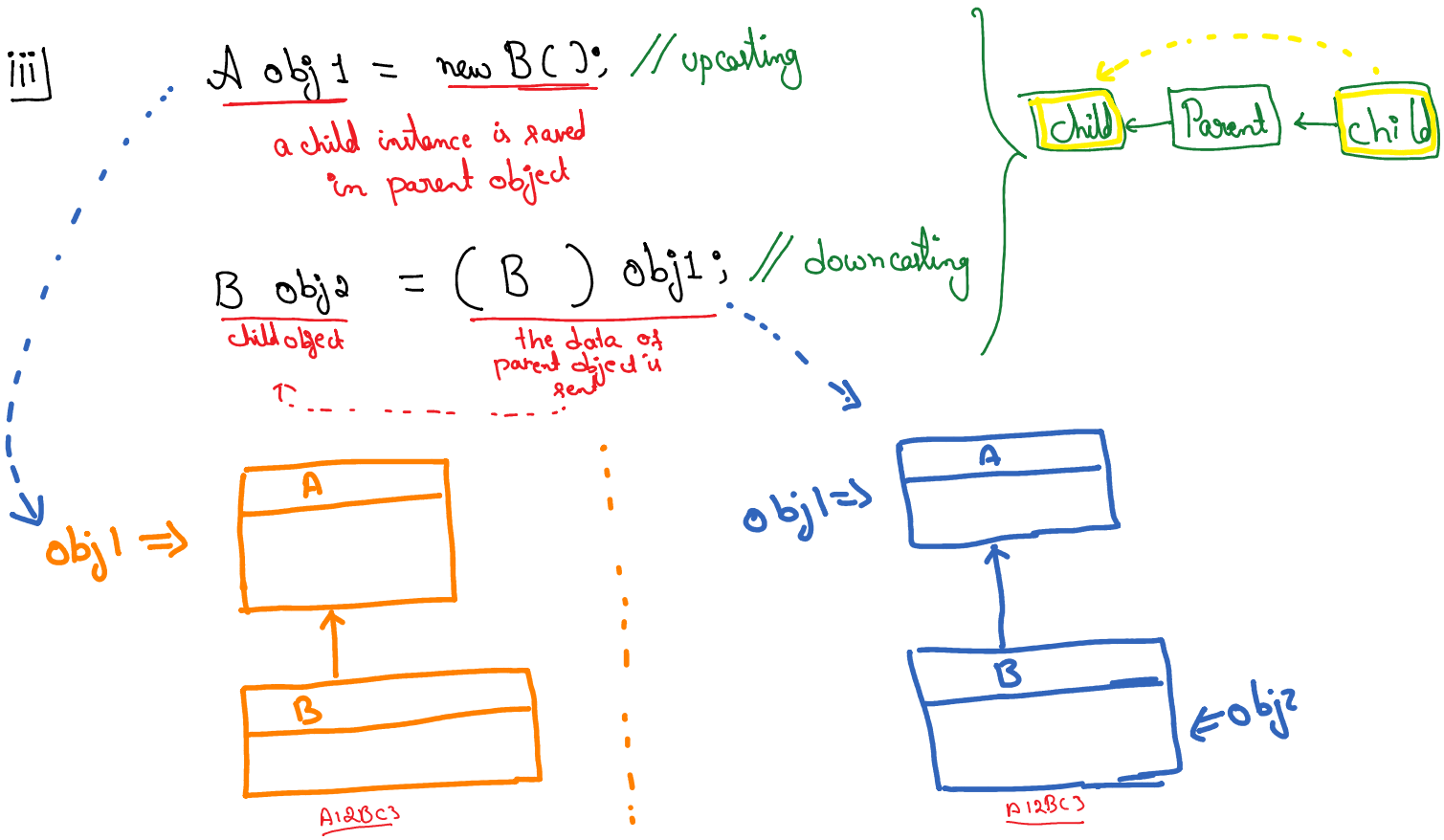
// classCastException R.T.E

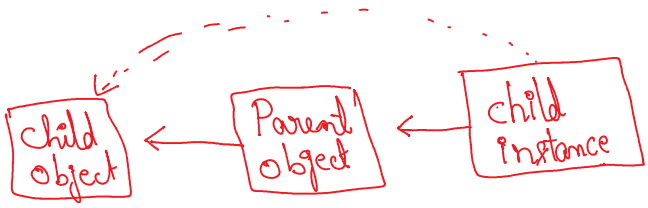
iii) `A obj1 = new B(); // upcasting`

a child instance is saved in parent object

`B obj2 = (B) obj1; // downcasting`

child object — the data of parent object is sent






```java
// upcasting is an implicit operation
// the child type is converted into parent type
class A10 {
    int i = 10;
}
```

```java
class B10 extends A10 {
    int j = 20;
}
public class Upcast { // driver class
    public static void main(String[] args) {
        A10 obj1; // parent object
        B10 obj2; // child object
        obj1 = new B10();
        // parent object holding a child instance
        obj2 = new B10();
        // child object holding a child instance
        /*
         * Even after upcasting the parent will only be able to access the parent data
         */
        System.out.println("Using parent obj to access i :" + obj1.i);
        // ! System.out.println("Using parent obj to access j :" + obj1.j);
        System.out.println("-----------------------------------------");
        System.out.println("Using child obj to access i :" + obj2.i);
        System.out.println("Using child obj to access j :" + obj2.j);
    }
}



class abcd {
}
class xyz extends abcd {
}
public class Downcast { // driver class
    public static void main(String[] args) {
        // 1. xyz obj1 = new abcd();
        // 2. xyz obj2 = (xyz) new abcd();
        // 3. by merging upcasting and downcasting together
        // parent object with upcast
        abcd obj1 = new xyz();
        System.out.println(obj1);
        // downcast using the cast operator
        xyz obj2 = (xyz) obj1;// copying data from parent
        System.out.println(obj2);
    }
}
```