Object Oriented Programming (OOP)

Mohamed Ezz

Lecture 6

Review

- What is the classes of email System?
  - Write the classes & member variables utilizing Hashmap & ArrayList classes
- What is the classes of facebook system ?
  - Write the classes & member variables utilizing Hashmap & ArrayList classes
- What is the classes of Library system ?
  - Write the classes & member variables utilizing Hashmap & ArrayList classes

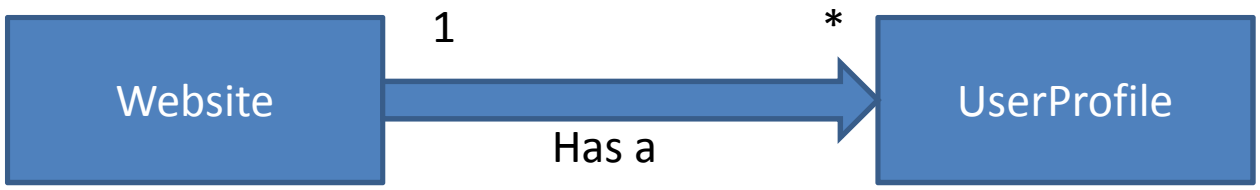
Lecture Objectives

- ✓ Understand Inheritance & software reusability
- ✓ Exercise Inheritance by example
- ✓ Understand Protected specifer and super keyword.
- ✓ Understand java class hierarchy Adam class.
- ✓ Practice inheritance on Website example

Login & Registration

Most of Web applications such as Facebook, yahoo, gmail, hotmail, amazon,… have two common functions:

1. **Register**: to register customer into their website
2. **Login** : to allow customer to access his/her protected data in the site

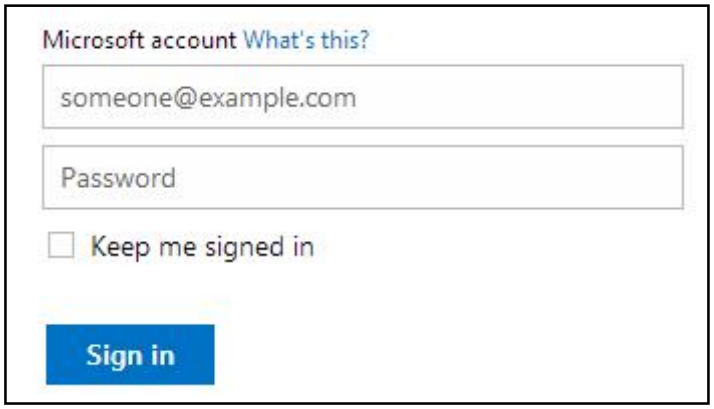


Registration

- Registration function
  - Create new UserProfile, and store it with e-mail/ID as key

Login

- **Login function**
  - Looking for UserProfile with ID or e-mail
  - validate password entered matches password in UserProfile



UserProfile Class

```
class UserProfile {
    private String firstName;
    private String lastName;
    private Date dateOfBirth; // use Date class created in lecture 4
    private String email;
    private String password;
    private String gender;
    private String mobile;
    private String country;
//setter & getter for all attributes except password

   public boolean setPassword (String password, String confirmPassword){
            if(passsword.equals(confirmPassword)){
                        this.password= password; return true;}
            else{
                        System.out.println("Password & confirm mismatch");
                        return false;}
    }
}
```

Website (facebook, mail,…) Class-1

```
class WebSite{
  protected HashMap <String, UserProfile > userList= new HashMap <String, UserProfile >();
  public boolean registerUser(String firstName, String lastName,
                                        Date dateOfBirth, String email, String gender,
                                        String mobile, String country, String password,
                                        String confirmPassword){
            if (userList.get(email) ==null){ // this mean email not registered before
                        UserProfile user= new UserProfile();
                        user.setfirstName(firstName);
                        user.setlastName(lastName);
                        //complete other setter  ……
                        if(user.setPaqsswprd(password,confirmPassword)){
                                    userList.put(email, user); return true;
                        } else{
                                    System.out.println("password & confirm mismatch"); return false;

                        }
            } else {
                        System.out.println("email already used, try another email"); return false;
            }
```

Website (facebook, mail,…) Class-2

```
public boolean login (String email, String password){
        UserProfile user= userList.get(email) ;
        if (user ==null){
                System.out.println("User not exist");
                return false;
        }
        if(password.equals(user.getPassword() ){
                return true;
        }
        else
                System.out.println("Incorrect password");
                retrun false;
        }
}
}
```

OOP Concept

- Encapsulation
- Information Hiding (Data Hiding)

Object Based

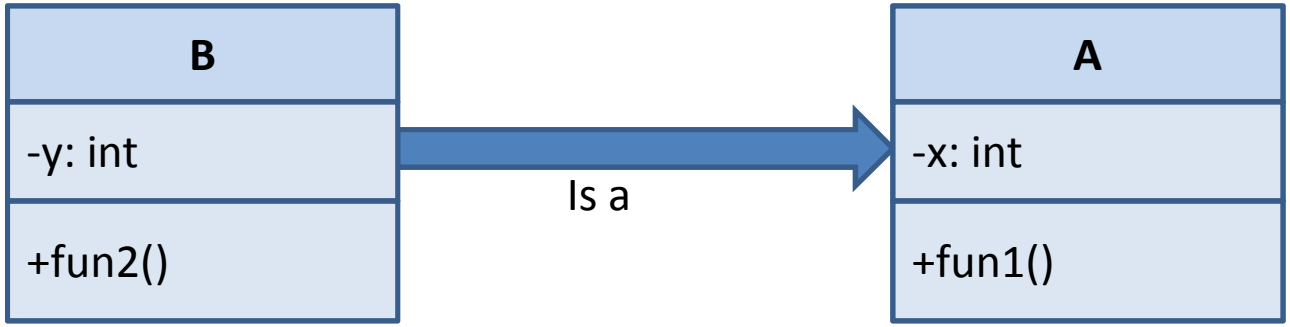
- **Inheritance**
- **Polymorphism**

For Software Reuse

Definition of inheritance

- A form of **software reuse** in which a new class is created by absorbing an existing class's members and embellishing them with new or modified capabilities.
- With inheritance, programmers save time during program development by reusing proven and debugged high-quality software.
- Class should inherit the members of an existing class. The existing class is called the superclass, and the new class is the subclass

Inheritance

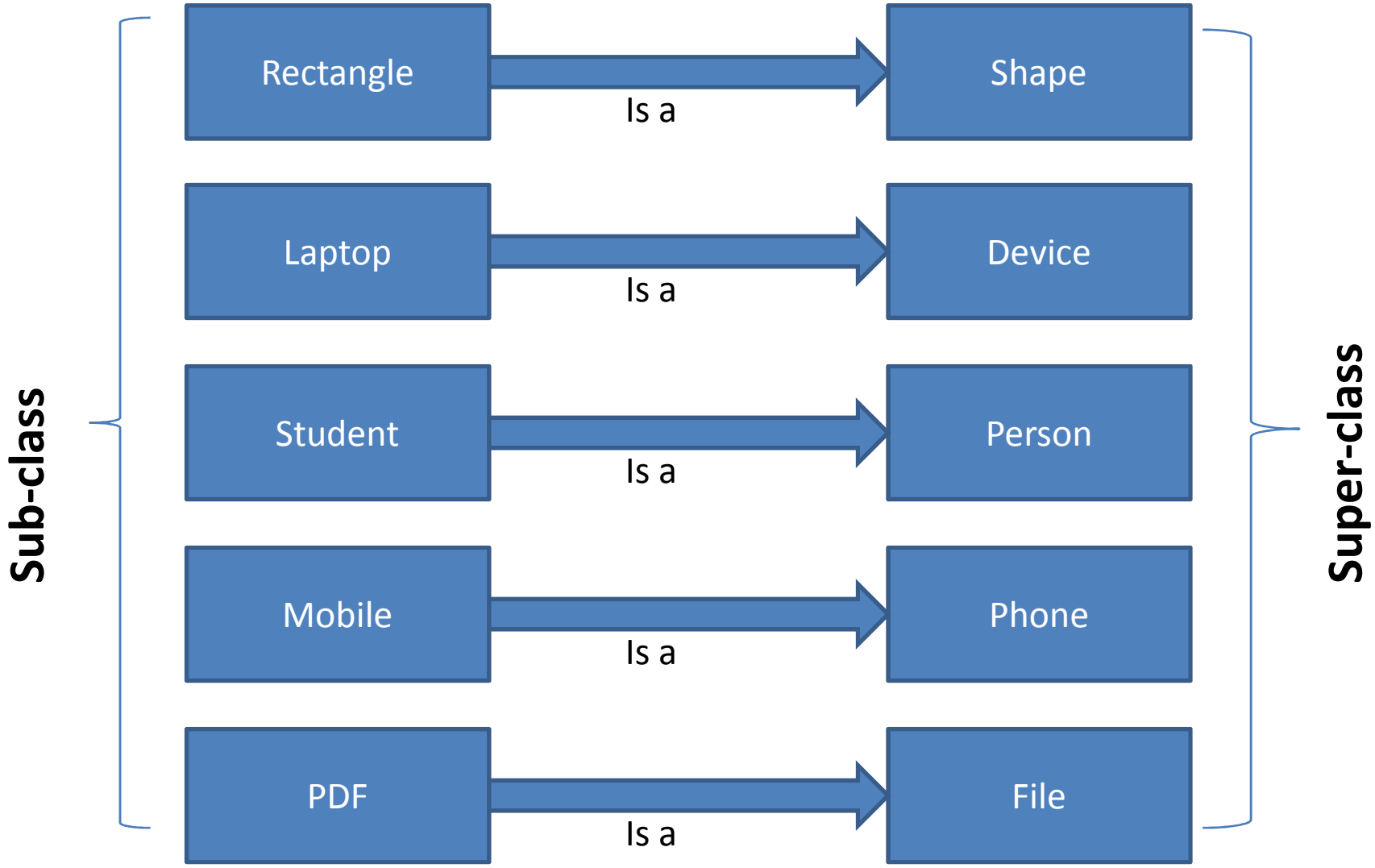



```
Class B inherit class A
```

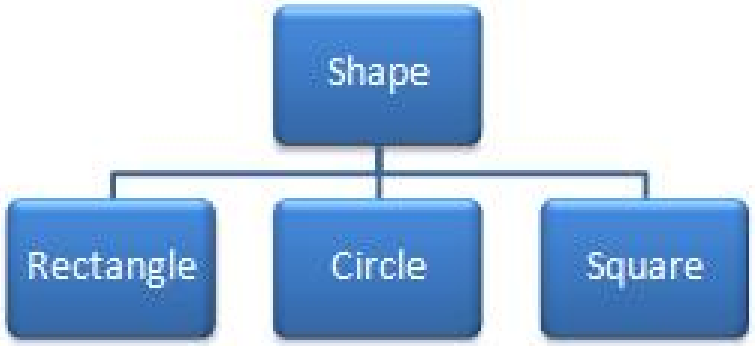
```
Class B: is the child class
Has → attribute: y
Has → attribute: x (from parent)
Has → method : fun2
Has → method : fun1(from parent)
```

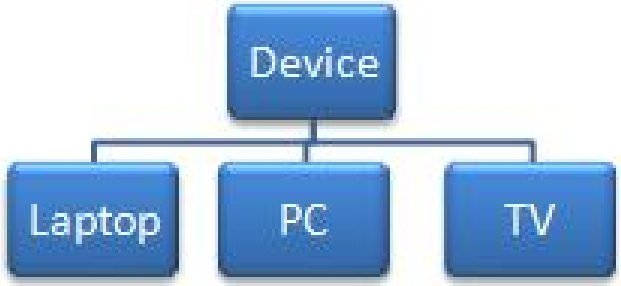
```
Class A: is the parent class
Has → attribute: x
Has → method : fun1
```

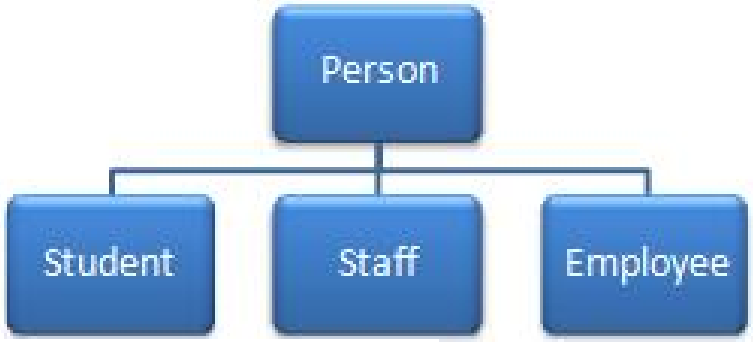
Inheritance Example

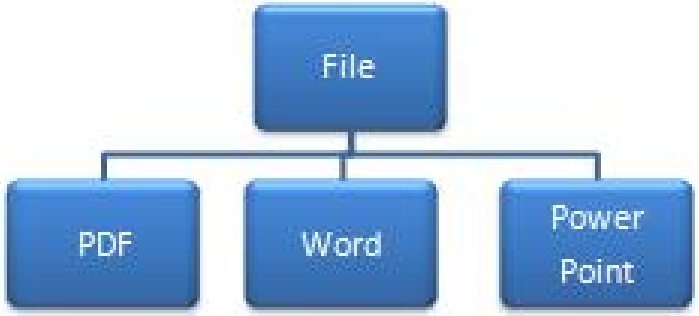


Class hierarchy Example









What is the common methods in super class?

Syntax of Inheritance

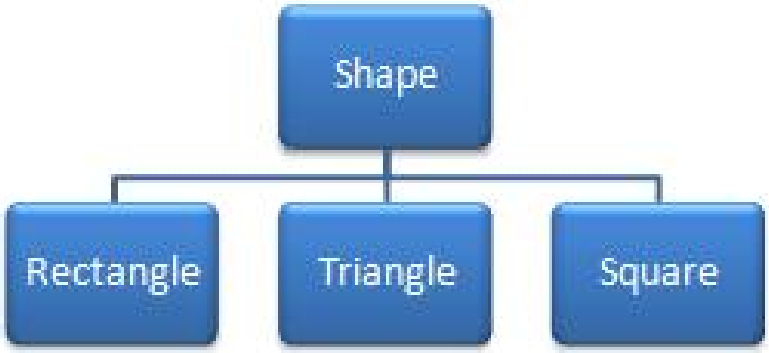
```
class subClass extends superClass {
        declarations

        constructor definition(s)

        method definitions
}
```

```
Example:
class Rectangle extends Shape{

}
```

Shape Inheritance Example

Shape.java

```
class Shape{
    private int color = 0;
    public void setColor(int color){
            this.color=color;
    }
    public int getColor(){
            return color;
    }
}
```



Rectangle.java

```
class Rectangle extends Shape{
    private int width = 0;
    private int height = 0;

    public Rectangle(int h, int w){
            width=w;
            height=h;
    }

    public void print(){
            System.out.println(width);
            System.out.println(height);
            System.out.println(getColor());
    }
}
```

Shape Inheritance Example

Triangle.java

```
class Triangle extends Shape{
    private int base = 0;
    private int height = 0;

   public Triangle (int h, int b){
            base=b;
            height=h;
    }

   public void print(){
            System.out.println(base);
            System.out.println(height);
            System.out.println(getColor());
    }
}
```

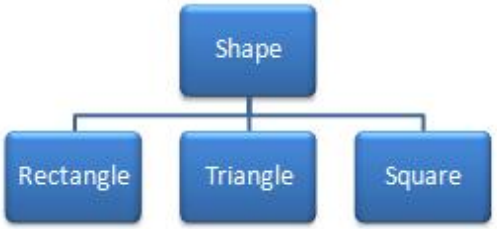
Square.java

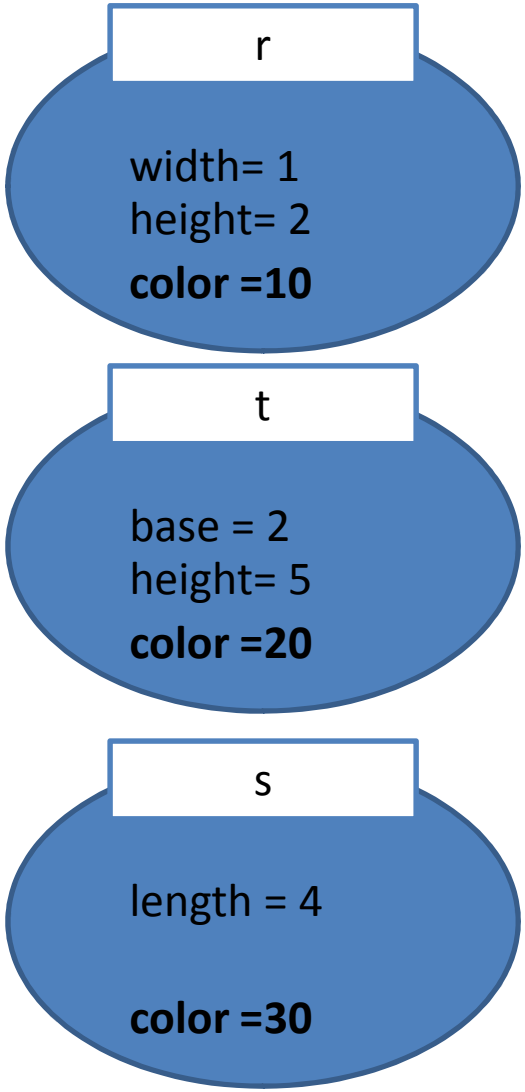
```
class Square extends Shape{
    private int length = 0;

   public Square (int l){
            length =l;
    }

   public void print(){
            System.out.println(l);
            System.out.println(getColor());
    }
}
```

Shape Inheritance Example

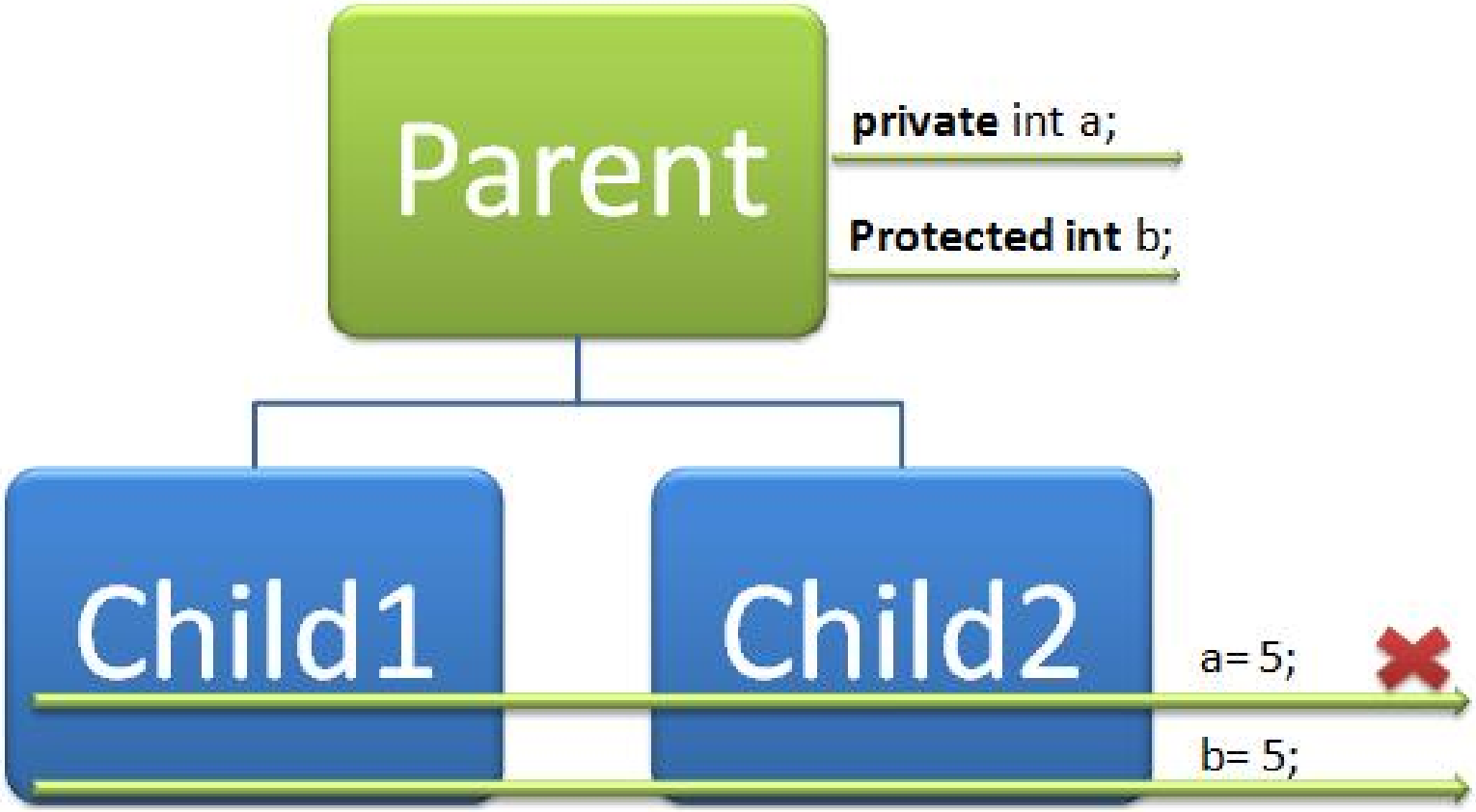
TestShape.java

```
class TestShape {
   public static void main(String arg[]){
            Rectangle r= new Rectangle (1,2);
            r.setColor(10);
            Triangle t= new Triangle(2,5);
            t.setColor(20);
            Square s= new Square (4);
            s.setColor(30);
            r.print();
            t.print();
            s.print();
   }
}
```





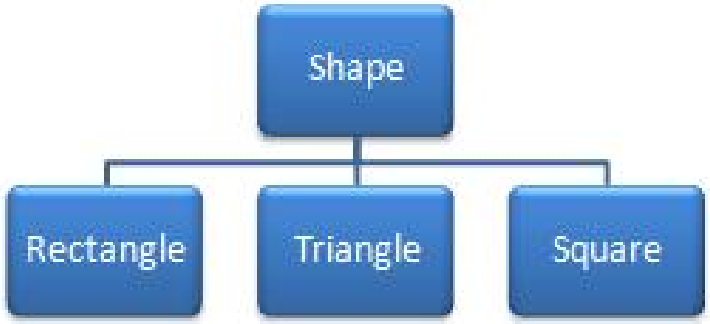
Inheritance – with Protect



Protected same as private, but accessed only by sub-classes (Childs only)

Shape Inheritance – with Protect

```
class Shape{
    protected int color = 0;
    public void setColor(int color){
            this.color=color;
    }
    public int getColor(){
            return color;
    }
}
```




```
class Rectangle extends Shape{
    private int width = 0;
    private int height = 0;

    public Rectangle(int h, int w){
            width=w;
            height=h;
    }

    public void print(){
            System.out.println(width);
            System.out.println(height);
            System.out.println(color);
    }
}
```

More about field modifiers

- Access control modifiers
  - *private:* private members are accessible only in the class itself
  - *package (Default):* package members are accessible in classes in the same package and the class itself
  - *protected:* protected members are accessible in classes in the same package, in subclasses of the class, and in the class itself
  - *public:* public members are accessible anywhere the class is accessible

Protect Access Specifier

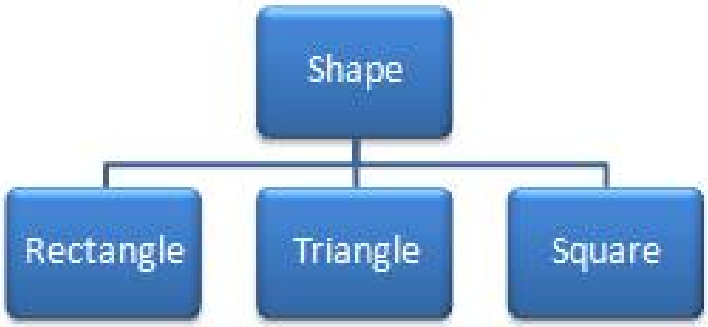
Super Class	Sub Class in other package

package aaaa;
Class A{
  private int x;
  int y;
  public int z;
  protected k;
}
	package bbbb;
	Class B extend A{
	  public void test(){
	  x=5;
	  y=3;
	  z=2;
	  k=4;
	
	  }
	}
	
	Which assignment correct?

Inheritance and Super keyword

Sub-class can access super-class member method or attribute using keyword **super**

- Inside Rectangle class:
  super.color;



Sub-class can call the super-class constructor using keyword **super**

super() → this will call shape constructor from inside rectangle class

Inheritance & super example

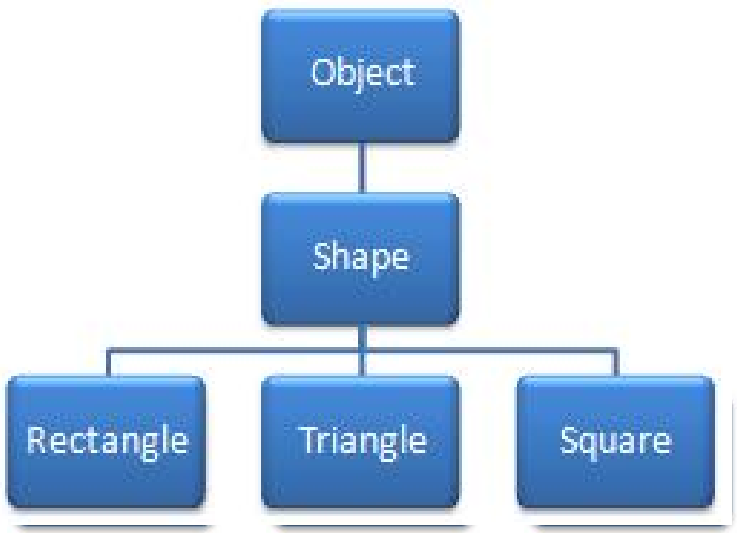
```
class Shape{
    protected int color = 0;
    public Shape(int c){
            color=c;
    }
    public void setColor(int color){
            this.color=color;
    }
    public int getColor(){
            return color;
    }
    public void print(){
            System.out.println(color);
    }
}
```

```
class Rectangle extends Shape{
    private int width = 0;
    private int height = 0;

    public Rectangle(int h, int w, int c){
            super(c);
            width=w;
            height=h;
    }

    public void print(){
        System.out.println(width);
        System.out.println(height);
        //System.out.println(super.color);
         super.print();
    }
}
```

Object is the Super Class of java?

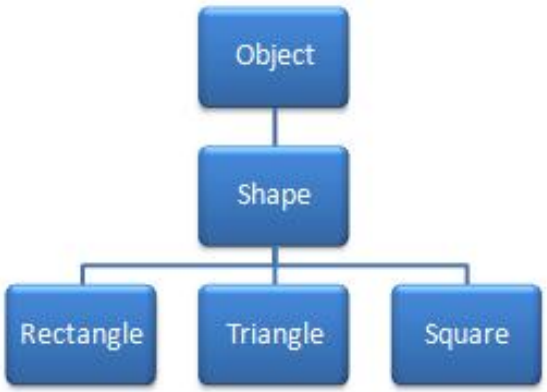
Java has class called **Object** which consider the parent class for all java classes, even user defined classed

This is **Adam** of java classes



Object Class

Object class has methods & attribute inherited for all java classes such as:



Methods	Description	Issues
equals()	compares two objects of same type for equality and returns true if equals and false otherwise	shallow compare which mean primitive attributes compared while object not compared(ref only)
toString()	returns a String representation of an object	
clone()	takes no arguments and returns a copy of the object on which it is called	shallow copy which mean primitive attributes copied while object not copied (ref only)

Object – toString method

```
class Rectangle {
    private int width = 0;
    private int height = 0;
    private Point p= new Point(0,0);

   public Rectangle(int h, int w){
            width=w;
            height=h;
    }
   public String toString(){
      return “width =” + width+ “height =” +
height+ “ x=”+p.x + “ y=”+p.y ;

   }
}
```

```
class TestRectangle {

   public static void main(string ar[]){
       Rectangle r1= new Rectangle (1,2);
       Rectangle r2= r1.clone();
        r2..p.setX(5);
        r2..p.setY(5);
       System.out.println(r1);
       System.out.println(r2);

   }
}
```

Inherit Website - Facebook

```
class Facebook  extends WebSite{


// methods of facebook such as
// addPost, share, like, addfreind,.....









}
```

Inherit Website - hotmail

```
class Hotmail extends WebSite{

// methods of hotmail such as
// send messge, view messge, ….

}
```