CSE
341

Class — 07
26-01-2026

#Abstract class

If a normal method inherits an abstract class it also becomes abstract

↳ So, we only create a reference of an abstract class

↳

abstract class → shape

↳ perimeter()

↳ area()

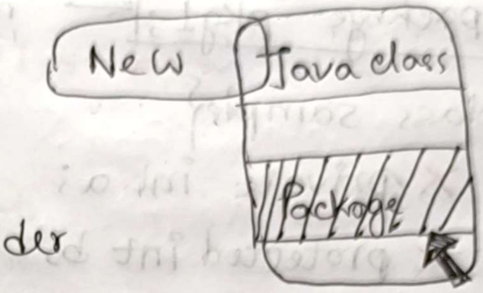
CSE 342

26-01-2026

```
abstract class Shape{
private double d1, d2;
    abstract void area();
      ~~System.out.println(~~
          ~~"Area: " + (d1 * d2)~~
      ~~);~~
   ~~}~~

Shape(double d1, double d2){
    this.d1 = d1;
    this.d2 = d2;
}
}

class Triangle extends Shape{
   ~~void area~~
   Triangle(double d1, double d2){
       ~~this.d1 = d1;~~  super(d1, d2);
       ~~this.d2 = d2;~~
   }
     void area(){
     System.out.println(
        "Area: " + (0.5 * d1 * d2)
     );
   }
}
```

```
class Rectangle extends Shape{
    Rectangle(double d1, double d2){
        ~~this.d1 = d1;~~ super(d1,d2);
        ~~this.d2 = d2;~~

    }

    void area(){
        System.out.println(
            "Area: " + (d1 * d2)
        );

    }

}

public class Main{
    public static void main(String[] args){
  Shape ~~Triangle~~ t1 = new Triangle(10,15);
        t1.area();

  Shape ~~Rectangle~~ r1 = new Rectangle(5,2);
        r1.area();

    }

}
```

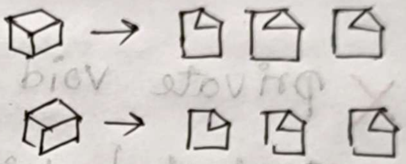
#Package



It like a folder

- to organize classes under a file.

- one package contains multiple files



- one project can have multiple packages

Project → Multiple packages → Multiple files

package 1 file 1

package 2 file 2

....

#Access Modifiers

same

	private	public	protected	default
same class	✓	✓	✓	✓
same pkg sub class	✗	✓	✓	✓
same pkg non-subclass	✗	✓	✓	✓
diff pkg sub class	✗	✓	✓	✗
diff pkg non-subclass	✗	✓	✗	✗

Keyword variable

```
package pkg1;   ← must be first line

(public) class sample{
    ✗ private int a;
    ✗ protected int b;
    ✓ public int c;
    ✗ default int d;

    public Sample(
        int a, int b, int c, int d
    ){
        this.a = a; this.b=b;
        this.c=c; this.d=d;
    }

    ✗ private void show1(){—("Private method.");}
    ✗ protected void show2(){—("Protected method.");}
    ✓ public void show3(){—("Public method.");}
    ✗ default void show4(){—("Default method.");}
}
```

```
package pkg2;
import pk1. sample;
public class Main{
    public static void main(String[] arg){
        Sample ob1 = new Sample();
        ↳ what if, { ob1.a
                     ob1.b
                     ob1.c
                     ob1.d }
    }
}
```

```
package pkg2;
import pkg1.Sample     or     pkg1.sample
                                  ↑
class Sample1 extends Sample{
    public void show5(){
          ...
    }
}
```

```
package pkg1;
public class Box{
        int height, width, depth;
    public Box(int h, int w, int d){
            ~~System.out.println(h*w~~
              height = h;
              width = w;
              depth = d;
        }
    public void show(){
            System.out.println (h*w*d);
        }
}
```

```
package  pkg2;
import  pkg1.Box;
class  Box1 extends Box{
   Box1(int h, int w,int d){
      super(h,w,d);
   }
}

public class Main{
  public static void main(String[] args){
     Box1 b1 = new Box1(5,10,15);
     b1.show();
  }
}
```