EECS2030 Advanced Object-Oriented Programming (Fall 2021)

Q&A - Lecture 6, 7a

Wednesday, November 24

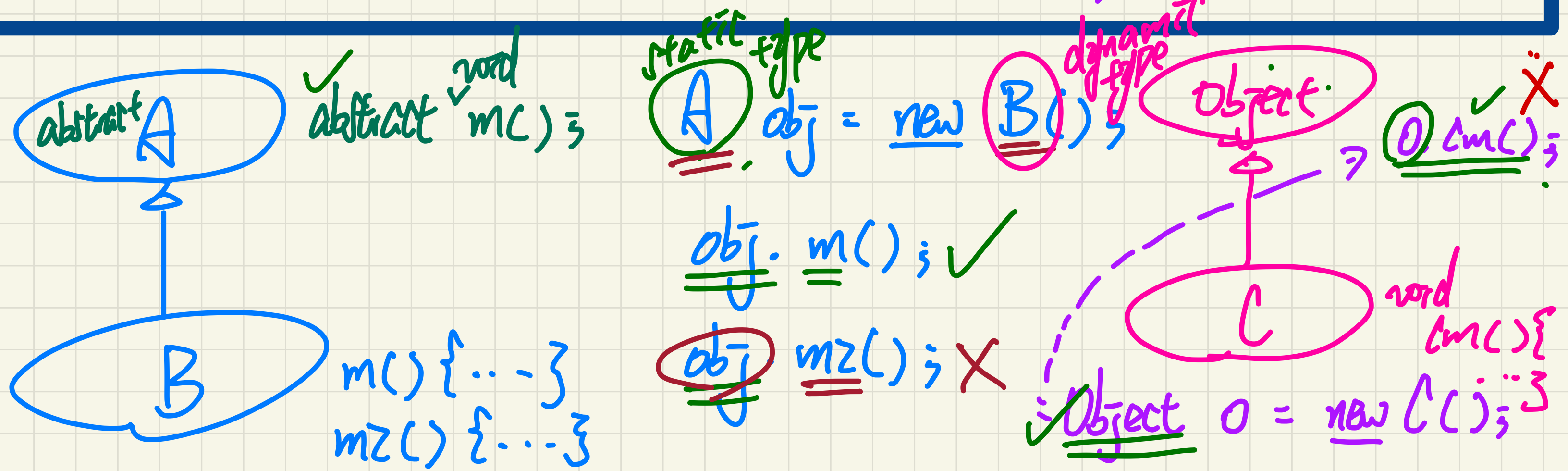
Announcement

- Lecture W11 (released: Nov. 23)
  1. Generics
  2. Programming Exercise
  3. ~~ArrayList vs. Hashtable~~
     ↳ videos (generics)
- Lab5 due Monday, December 6
- Written Test 3
- Programming Test 3
  ↳ 1. Simple example test
  2. Lab4

(Part A2, Timestamp: 9:45) Professor, as you said in this video,
methods that are not abstract and are not explicitly written in a parent class
can cause problems if they are defined only in sub classes and then called.

As we know that Object class is the parent of every class in Java, then when we declare some method in our newly created class that inherits from the Object class, how will calling this method not cause a compilation error because not every method that we declare are in Object class and they are not abstractly defined.

v1 = v2 ;

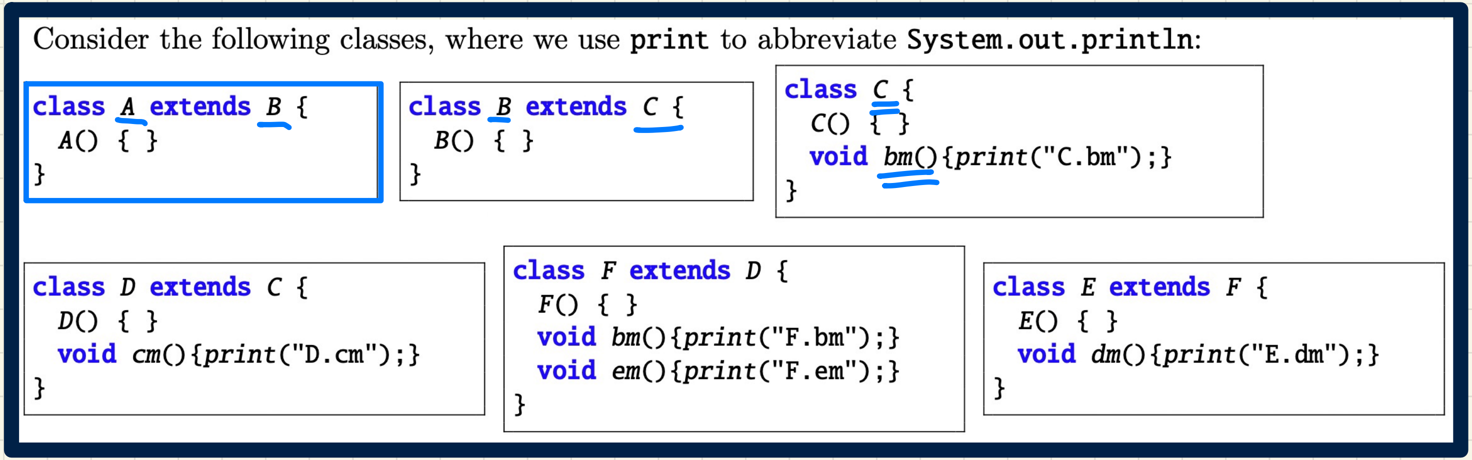


I've heard some programmers say that
certain books about code have been very influential to them.
Do you think this is still a relevant learning method
given our access to sources like StackExchange/youtube?
Are there any books that made an impact on you?

- Youtube Videos (quality? false info?)
- books

Written Test 3 Example Questions

A → B → C bm



Consider the following classes, where we use print to abbreviate System.out.println:

```
class A extends B {
  A() { }
}
```

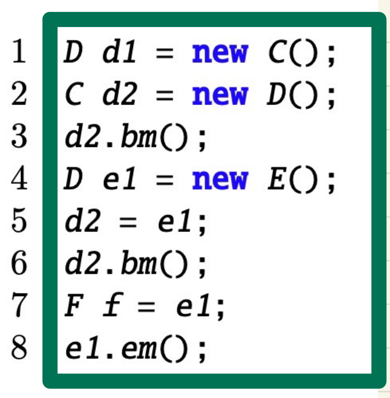
```
class B extends C {
  B() { }
}
```

```
class C {
  C() { }
  void bm(){print("C.bm");}
}
```

```
class D extends C {
  D() { }
  void cm(){print("D.cm");}
}
```

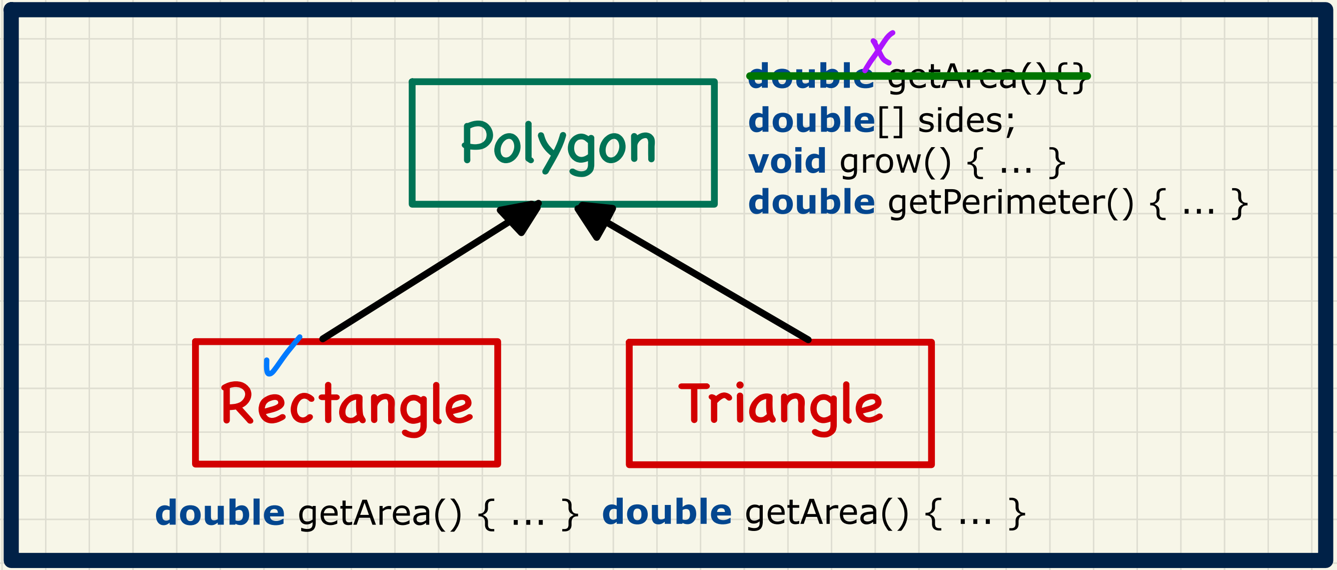
```
class F extends D {
  F() { }
  void bm(){print("F.bm");}
  void em(){print("F.em");}
}
```

```
class E extends F {
  E() { }
  void dm(){print("E.dm");}
}
```



```
D d1 = new C();
C d2 = new D();
d2.bm();
D e1 = new E();
d2 = e1;
d2.bm();
F f = e1;
e1.em();
```

- sketch paper ready
- draw the corresponding inheritance hierarchy diagram.



Polygon

~~double getArea(){}~~

double[] sides;

void grow() { … }

double getPerimeter() { … }

Rectangle

Triangle

double getArea() { … } double getArea() { … }

```
Polygon[]  ps = . . . . ;
for( int  i = 0 ;  i < nop ;  i++ ){
    if( ps[i]  instanceof  Rectangle ){  ps[i].getArea(); }
                                      ((Rec.)  ps[i]).getArea();
}
```

ST: Polygon