Exercises on Polymorphism

COMP-202B, Summer 2011, All Sections

Constructors

Suppose I have the following class and interface definitions. In cases where it is written:

```
{ .. }
```

it means that the method is defined but for the purpose of this question it does not matter what the method does.

```
public interface A {
int methodOne();
double methodTwo(int x);
}
```

```
public class B implements A {
int methodOne() { ... }
double methodTwo(int x) { ... }
double methodTwo(double d) { ... }
double methodTwo(String s) { ...}
}
```

```
public class C implements A {
int methodOne() {...}
double methodTwo(int x) { ...}

}
```

```
public class D extends C {
double methodTwo(int x) { ... }
double methodTwo(String input);
double methodThree();
}
```

Using constructors

Which of the following will compile and which will not:

- A a = new A();
- A a = new B();
- A a = new C();
- A a = new D();
- B b = new A();

- B b = new B();
- B b = new C();
- B b = new D();
- C c = new A();
- C c = new B();
- C c = new C();
- C c = new D();
- D d = new A();
- D d = new B();
- D d = new C();
- D d = new D();

Using polymorphism

For each of the following, figure out a)whether the code compiles, b)whether the code has a run time error and c) which method executes on the last line of the block:

-
  ```
  A a = new D();
  a.methodThree();
  ```

-
  ```
  A a = new D();
  ((D)a).methodThree();
  ```

-
  ```
  A a = new C();
  ((D)a).methodThree();
  ```

-
  ```
  A a = new B();
  a.methodTwo(3.0);
  ```

-
  ```
  A a = new B();
  a.methodTwo(3);
  ```

-
  ```
  C c = new D();
  c.methodTwo(3);
  ```

-
  ```
  C c = new D();
  c.methodOne();
  ```

-
  ```
  C c = new D();
  c.methodTwo("hello");
  ```

-
  ```
  C c = new D();
  ((D)c).methodTwo("hello");
  ```