So You Think You Know Java

Adam Berger
15-212 Spring 1998
Feb 18, 1998

1. What is a *strongly-typed language*? Name an advantage and disadvantage of strong typing in a programming language.

2. How is Java *architecture-neutral*? Why is this a good thing? Why is this a bad thing?

3. If we look in the `java.lang.Math` class, we'll find a line which looks like:

   ```
   public static final double PI = 3.14159265;
   ```

   What purpose is each of the keywords `public, static, final` serving here? Explain how together, these three lines obviate the need for a *preprocessor*. (And just what is a preprocessor, anyway?)

4. What does *overloading* mean? Give an example of how Java uses overloading. Explain why overloading is good, and also how it can be bad. Overloading isn't just confined to programming languages; give an example of overloading in English.

5. The two ways modern programming languages like Java handle the memory requirements of running programs is with the *stack* and *heap.*

   (a) Define these two terms.

   (b) Which is being used for `a` in each of the following program fragments?

   ```
   public void foo() {
      static String a = "Hello, world";
     }

   public void bar() {
      String a;
      a = new String("Hello, world");
    }
   ```

   (c) Which, if either, does garbage collection handle?

6. What's wrong with the following Java program? (There are three bugs.)

   ```
   class broken
   {
     public static main (String[] args) {
       System.out.println("Hello, world\n");
       foo();
     }

     private void foo() {
       System.out.println("Hello, again\n");
     }
   }
   ```

7. What is *multithreading*? Give an example of a situation where multithreading would be useful.