1. Suppose we have a class `A` which has a constructor that takes a single integer.

   (a) After the following statements have been executed, how many `A` objects will exist (not counting garbage objects) and which objects are they? Explain your answer and include in your explanation a picture of Java's memory (draw Java's memory the way it is shown by the Java Visualizer).

   ```
   A a = new A(100);
   A b = new A(150);
   A c = b;
   b = a;
   a = null;
   ```

   (b) After the following statements have been executed, how many `A` objects will exist (not counting garbage objects) and which objects are they? Explain your answer and include in your explanation a picture of Java's memory (draw Java's memory the way it is shown by the Java Visualizer).

   ```
   A a1 = new A(200);
   A a2 = new A(250);
   A a3 = a2;
   a1 = null;
   a2 = a1;
   ```

2. Consider this code that creates some `Rectangle` objects:

   ```
   Rectangle a, b, c;
   a = new Rectangle(10, 20);
   b = new Rectangle(10, 20);
   c = b;
   ```

   After this code executes, what is the value for each of these six boolean expressions?

   ```
   a == b
   a.equals(b)
   a == c
   a.equals(c)
   b == c
   b.equals(c)
   ```

   Also, write two clear sentences that explain the difference between `==` and the `equals()` method.

3. Here is a simple `Point` and `Circle` class.

```
class Point
{  private double x, y;

   public Point(double x, double y)
   {  this.x = x;
      this.y = y;
   }
   public double getX(){ return x; }
   public   void setX(double z){x = z;}
   public double getY(){ return y; }
}
```

```
class Circle
{  private Point c;  // center
   private double r; // radius

   public Circle(Point c, double r)
   {  this.r = r;
      this.c = c;
   }
   // more stuff
}
```

(a) The constructor in `Circle` has a "privacy leak". Explain why.
Hint: Consider the following code.

```
Point p = new Point(1,2);
Circle c = new Circle(p, 10);
p.setX(100);
```

(b) Rewrite the `Circle` constructor to fix this problem.

4. Only one of the following four class definitions defines a proper abstract class. Which one of the four is it? For the remaining three classes, briefly explain why it does not properly define an abstract class.

```
class A1 {
   abstract void unfinished();
}

abstract class A2 {
   abstract void unfinished(){ }
}

abstract class A3 {
   abstract void unfinished();
}

abstract class A4 {
   private abstract void unfinished();
}
```

5. What does the following program print out. Explain why.

```
class Thing
{  public int a;
   public int b;
   public Thing(int a, int b){this.a=a; this.b=b;}
}

public class Test
{  public static void f(Thing x, int y)
   {  x.a++;
      y++;
   }
   public static void main(String[] args)
   {  Thing x = new Thing(1,1);
      int y = 1;
      f(x, y);
      System.out.println("x.a = " + x.a + " and x.b = " + x.b);
      System.out.println("  y = " + y);
   }
}
```

6. Suppose that we have classes `A`, `B`, `C` and `D`. Suppose that `B` is a subclass of `A`, that `C` is a subclass of `B`, and `D` is a subclass of `A`.

```
class A { }
class B extends A { }
class C extends B { }
class D extends A { }
```

Suppose that we make the following declarations.

```
A a1 = new A();
A a2 = new C();
D d1 = new D();
```

For each line below, explain what, if any, errors would be caused by the statement in that line. Be sure to consider both compile time and run time errors.

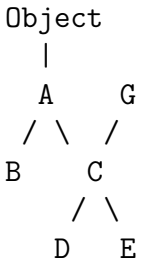
```
A a3 = new B();
B b1 = new A();
B b2 = (B) a1;
B b3 = (B) a2;
B b4 = (B) d1;
B b5 = (C)(A)new D();
```

7. Consider the following classes:

```
class Organization {
   Organization(){ /* default constructor */ }
   public void printMe() { System.out.println("Organize."); }
}
class Company extends Organization {
   Company(){ /* default constructor */ }
   public void printMe() { System.out.println("Be productive."); }
}
class MidCap  extends Company {
   MidCap(){ /* default constructor */ }
   public void printMe() { System.out.println("Think big."); }
}
public class InternetCo extends MidCap {
   InternetCo(){ /* default constructor */ }
   public void printMe() { System.out.println("Be cool."); }
   public static void main(String[] args) {
      Company mid = new MidCap();
      MidCap netscape = new InternetCo();
      Object obj = new Organization();
      Organization startup = new Company();
      mid.printMe();
      netscape.printMe();
      ((Organization) obj).printMe();
      obj = netscape;
      ((MidCap) obj).printMe();
      obj = startup;
      ((Organization) obj).printMe();
   }
}
```

What is the output from running the `InternetCo` class?

8. Suppose that we have classes `A, B, C, D, E` and an interface `G` that are related to each other this way.



(a) In the following code fragment, what can we replace the `???` with (where `...` represents appropriate parameters)? Be precise. Don't just say "methods in `c`"

```
C c = new D();
c.???(...);
```

(b) In the following code fragment, what can we replace the `???` with?

```
G g = new ???();
```

(c) In this code fragment, what can we replace the `???` with in the declaration for `x`?

```
??? x = new E();
```

(d) In this code, what can replace `???` in the cast of `c` so that the code compiles?

```
C c;
// some code where c gets a value
A a = (???)c;
```

9. Suppose that `a` and `b` are arrays of integers. What is the value of `b[0]` and `b[1]` after the following statements have been executed? Explain why. Your explanation should include a simple drawing of Java's memory (draw Java's memory the way it is shown by the Java Visualizer).

```
a[0] = 1;
b[0] = 2;
b = a;
b[1] = 3;
a[1] = 4;
```

10. Draw a picture of Java's memory after the following two lines of code are executed. Your picture should be similar to the ones drawn by the Java Visualizer.

```
Color[][] c = new Color[3][2];
c[1][0] = new Color(1f, 0f, 0f); // red
```

11. The following code segment creates and initializes a two dimensional ragged array. Draw two representations for the array and its contents. One picture should represent the array as most people think about it, as a two dimensional array with rows and columns. The other picture should represent how Java actually stores the array.

```
int[][] a = new int[6][];
for (int i = 0; i < 3; i++)
{  a[i] = new int[i+1];
   a[5-i] = new int[i+1];
   for (int j = 0; j < a[i].length; j++)
   {  a[i][j] =   10*i + (j+1);
      a[5-i][j] = 10*(5-i) + (j+1);
   }
}
```

12. Suppose that we define the following two classes.

```
class Class1
{
   int x = 10;

   public int methodA(int y)
   {  return y + this.x; }

   public void methodB(int y)
   {  this.x = y + this.x; }
}
```

```
class Class2
{
   static int x = 0;

   public static int methodA()
   {  return x; }

   public static void methodB(int y)
   {  x = y; }
}
```

Draw a picture of Java's memory after the execution of the following code (in, for example, DrJava's interactions pane or in the Java Visualizer). Be sure that your memory picture includes the (final) values of all the object and class variables. Clearly denote in your drawing any non-referenced objects that will be garbage collected.

```
Class1 ref1 = new Class1();
Class1 ref2 = new Class1();
Class1 ref3 = ref2;
int x = 5;
ref1.methodB(x);
ref2.methodB( ref1.methodA(2) );
ref1 = ref3;
Class2.methodB( ref1.methodA(Class2.methodA()) );
```

13. (a) What do method overloading and method overridding have in common?

    (b) Give two ways in which method overloading and method overridding are different.

14. (a) Explain the meaning of the keywords `super` and `this` when they are used as method names.

    (b) Explain the meaning of the keywords `super` and `this` when they are used as a "calling object."

15. Below is the outline of a simple class called `BusyWork`. In the blank space provided in the class, add two methods to the class. The two methods that you add can make use of the two given methods `setNumber()` and `getNumber()`.

    (a) Add a mutating method `increment()` that adds 1 to the value of `m`.

    (b) Add a (non-mutating) returning method `decrement()` that returns a `BusyWork` object whose value of `m` is one less than the value stored in the calling object.

```
public class BusyWork
{
   int m = 0;

   public void setNumber(int newM)
   {
      this.m = newM;
   }

   public int getNumber()
   {
      return this.m;
   }



}//BusyWork
```

16. Suppose that

```
x = y;
```

    is a "widening" assignment.

    (a) If `x` and `y` are primitive variables, explain why the assignment can also be referred to as a "widening conversion".

    (b) If `x` and `y` are reference variables, how are the types of `x` and `y` related to each other?

    (c) If `x` and `y` are reference variables, explain why the assignment should not be referred to as a "conversion".

    (d) If `x` and `y` are reference variables, you can even make a case that the assignment should be called a "narrowing" assignment. Explain why. (Hint: What can you say about the methods callable on `x` as compared to the methods callable on `y`?)

17. Consider the following class definition. The public interface of this class is that it has three fields named `area`, `length`, and `WIDTH` (which is a public constant), but the implementation should be that the class contains a single (private) double `area`. Write implementations for the constructor and the two set and the two get methods in such a way that `length` is derived from the variable `area`.

```
class RestrictedRectangle
{
   public final double WIDTH = 3.0;
   private double area;

   public RestrictedRectangle(double length)
   {

   }

   public double getArea()
   {

   }

   public double getLength()
   {

   }

   public void setArea(double area)
   {

   }

   public void setLength(double length)
   {

   }
}
```

18. What does the following program print out? Explain why.

```
public class Mystery
{
   public static void doSomething(int i)
   {  System.out.println("my int is " + i); }

   public static void doSomething(double x)
   {  System.out.println("my double is " + x); }

   public static void main(String[] args)
   {  double z = 3.21;
      int k = -2;
      doSomething( z );
      doSomething( (int)z );
      doSomething( k );
      doSomething( (double)k );
   }
}
```

19. Draw an inheritance diagram that shows the relationships between the following eight classes.

`Person`	`Classroom`
`Employee`	`Instructor`
`Student`	`Secretary`
`Object`	`ComputerLab`

20. Suppose that class `B` is a subclass of class `A` and we define the following variables.

```
A a = new A();
B b = new B();
```

Which of the following assignments are legal?

```
a = b;
b = a;
a = new B();
b = new A();
```

21. Suppose that `A` is a class that implements interfaces `I` and `J`.

```
interface I {}
interface J {}
class A implements I, J { }
```

Suppose that we define the following variables.

```
A a = new A();
I i = null;
J j = null;
```

Which of the following assignments require a cast?

```
i = a;
j = a;
j = i;
a = i;
```

22. Suppose that `A` and `B` are classes that implement interface `I`.

```
interface I {}
class A implements I { }
class B implements I { }
```

Suppose that we make the following definitions.

```
I ia = new A();
I ib = new B();
```

Which of the following casts will throw an exception?

```
A a1 = (A) ia;
A a2 = (A) ib;
B b1 = (B) ia;
B b2 = (B) ib;
```

23. Suppose we have the following definitions.

```
interface I { }
interface J { }
class A { }
class B extends A { }
class C implements I { }
class D extends C implements J { }
class E extends B { }
class F extends A implements I, J { }
```

Draw an inheritance diagram for these definitions (as in Problem 8). Try to draw as neat a diagram as you can (you can draw it without any lines crossing each other).