Test 2 study guide:

1) Definitions
    a. Composition vs. inheritance
    b. GUI (What are they)
    c. Array vs. Array list
2) Inheritance
    a. **Develop a base class**
    b. **Develop a derived class**
    c. **Write driver to use them**
3) **Aggregation/Composition**
    a. Develop base class
    b. Develop aggregated/composed classes
    c. Write driver to use them
4) **Combination of Composition and Inheritance (for example Array of derived class used in a new class)**
5) ArrayList and arrays
    a. Know how to use them(length, add, remove,..)
    b. Know how to declare them
    c. Trace code segment using arrays and ArrayList
    d. Write code to use them

Sample questions:

1) Develop a base class named PersonalInfo to include name, ID and address of a person. Include all the necessary components: constructor, getters, setters, …..

2) Develop a derived class named Employee to extend PersonalInfo and include all the necessary components, use as many as the base class methods as possible such as super, over-ridden methods and any other one.

3) Develop a test class to use both base and derived classes PersonalInfo and Employee and an array of PersonalInfo.

4) Write the function removeAdjacentEvens, which removes from a list any even numbers that directly follow another even number in the list. Make sure it is efficient. Use the following heading for the method:

```
// removes from the list all even numbers that immediately follow another even number
public static void removeAdjacentEvens (int [] list) {
    int  ct = 0 ;
    for (int I = 0 ; I < list.length-1 ; I ++) {
            if ( list[I ] % 2 == 0 && list[I +1] %2 == 0){
                    for (int j = I + 1 ; j < list.length-1 ; j++)
                            list[j] = list[j+1];
                    ct ++ ;
            }
    int [] l1 = new l1[list.length – ct] ;
    for (int I = 0 ; I < l1.size ; I  ++)
            l1[I ] = list[i] ;
    list = l1 ;
    }
```

Examples:

list	list after call to removeAdjacentEvens(list):
(6 2 5 2 8 4 3)	(6 5 2 3)
(4 2 5 1)	(4 5 1)
()	()
(5 3 7)	(5 3 7)
(2 6 4)	(2)

What is printed?

```
class Base {
    public void show() {
    System.out.println("Base::show()   called");
    }
}

class Derived extends Base {
    public void show() {
    System.out.println("Derived::show()   called");
    }
}

public class Main {
    public static void main(String[] args) {
            Base b = new Derived();;
            b.show();
    }
}
```

What is printed?

```
class Base {
    final public void show() {
       System.out.println("Base::show() called");
    }
}

class Derived extends Base {
    public void show() {
       System.out.println("Derived::show() called");
    }
}

class Main {
    public static void main(String[] args) {
        Base b = new Derived();;
        b.show();
    }
}
```

What is Output of following Java program?

```
class Base {
    public void Print() {
        System.out.println("Base");
    }
}

class Derived extends Base {
    public void Print() {
        System.out.println("Derived");
    }
}

class Main{
    public static void DoPrint( Base o ) {
        o.Print();
    }
    public static void main(String[] args) {
        Base x = new Base();
        Base y = new Derived();
        Derived z = new Derived();
        DoPrint(x);
        DoPrint(y);
        DoPrint(z);
    }
}
```

What is printed?

```
class Grandparent {
  public void Print() {
    System.out.println("Grandparent's   Print()");
  }
}

class Parent extends Grandparent {
  public void Print() {
  super.Print();
    System.out.println("Parent's   Print()");
  }
}

class Child extends Parent {
  public void Print() {
    super.Print();
    System.out.println("Child's   Print()");
  }
}
class Main {
  public static void main(String[] args) {
    Child c = new Child();
    c.Print();
  }
}
```

What is printed?

```
class Complex {

    private final double re;
    private final double im;

    public Complex(double re, double im) {
        this.re = re;
        this.im = im;
    }

    public String toString() {
        return "(" + re + " + " + im + "i)";
    }
}

class Main {
  public static void main(String args[])
  {
       Complex c = new Complex(10, 15);
       System.out.println("Complex number is " + c);
  }
}
```