Do not turn this page until you have received the signal to start.
(Please fill out the identification section above, and read the instructions below.) Good Luck!

This midterm consists of 3 questions on 3 pages (including this one). When you receive the signal to start, please make sure that your copy is complete. Comments are not required except where indicated, although they may help us mark your answers. They may also get you part marks if you can’t figure out how to write the code.

For 1 bonus mark write your student number at the bottom of pages 2-3 of this test.
If you use any space for rough work, indicate clearly what you want marked.

TOTAL: _____/50

Short Java API descriptions (all methods are public):

class Integer:
  Integer(int i) // An Integer with value i
  static int parseInt(String s) // s’s value, as an int.

class Double:
  Double(double d) // A Double with value d
  static double parseDouble(String s) // s’s value, as a double.

class String:
  String substring(int i, int j) // = the letters between i (inclusive) and j (non-inclusive).
  String substring(int i) // = the letters from i (inclusive) to the end.
  int indexOf(String s) // = the index of s in this String; -1 if s is not a substring.
  int indexOf(String s, int i) // = index of s in this String after index i; -1 if s not found.
  int length() // = the number of characters in this String.

class JOptionPane:
  static String showInputDialog(String m) // get input from the user, prompting with m.


Question 1. [14 marks]

1. Given two double variables payment and cost, and an int variable amount, write an expression that multiplies payment by cost, and stores the result in amount.

2. Given an int variable mark, write an expression that evaluates to true if the value of mark is even, and evaluates to false otherwise.

3. Given double variables balance and owing, write an expression that evaluates to true if balance is negative or if balance is greater than owing, and evaluates to false otherwise.

4. Write a statement that declares an Integer reference named numStudents, and creates and assigns it an instance of Integer with an initial value of 30.

5. Write the definition of a class Midterm. The class has no methods, one instance variable of type String called section, and one static int variable called quantity, which is initialized to 150.

6. Write the definition of the method makePhoneNumber, which has an int parameter representing the area code and a String parameter representing a 7-digit number of the form 555-5555. The method returns a String. For example, given the parameters 111 and 222-3333, the method would return a String of the form: (111) 222-3333

7. You are given a class called Clock, which has an instance variable of type int called hours and a boolean instance variable called ticking. Write a constructor for the class that takes a Clock object as a parameter and copies that object’s instance variable values to the object being created.
Question 2. [18 marks]

In lecture, we built a Java class used to keep track of books in a library. It looked something like this:

class BookRecord {
    private String ISBN;
    private String author;
    private String title;

    public void setISBN(String i) { this.ISBN = i; }
    public void setAuthor(String a) { this.author = a; }
    public void setTitle(String t) { this.title = t; }

    public String getISBN() { return this.ISBN; }
    public String getAuthor() { return this.author; }
    public String getTitle() { return this.title; }

    public String toString() {
        return this.ISBN + " : " + this.author + " : " + this.title;
    }
}

Now, you will write a new class, SalesBookRecord, which extends the BookRecord class for use in a system that is used to sell books. Specifically, your new class will include the following:

1. a private field used to store the price of the book (in dollars, allowing for prices such as 10.95).
2. public get and set methods for price.
3. A discount procedure, which takes a discount, n, (int between 0 and 100), and reduces the cost of the book by n percent (for example, if the price of the book is 10.0, and this method is called with a value of 30, the price of the book is reduced to 7.0).
4. A toString function, which returns a String representation of this SalesBookRecord in the following format:

author : title : price : isbn

Write your class on the following page. Be sure to include some comments for tricky parts of code.
Write your answer to Question 2 here:
Question 3.  [18 marks]

Consider these two classes.

```java
public class Store {
    private String location;
    private Item i1;
    private Item i2;

    public Store(String l) {
        this.location = l;
    }

    public void setItem(Item i) {
        this.i1 = i;
    }

    public void setOtherItem(Item i) {
        this.i2 = i;
    }

    public String toString() {
        return this.location + "", " + this.i1 + ", " + this.i2;
    }

    public static int total(int p1, int p2) {
        return p1 + p2;
    }
}
```

```java
public class Item {
    private static int count = 0;
    private String name;
    private String barcode;
    private int price = 0;

    public Item(String n, String b) {
        this.name = n;
        this.barcode = b;
        Item.count++;
    }

    public void setPrice(int p) {
        this.price = p;
    }

    public void discount(double d) {
        this.price = this.price - (int)(this.price * d);
    }

    public int getPrice() {
        return this.price;
    }

    public String getBarcode() {
        return this.barcode;
    }

    public String toString() {
        return this.name + ", ", " + this.barcode;
    }

    public static int getCount() {
        return count;
    }
}
```

On the next page is a `TestCase` subclass that tests these classes. There are four test methods and 8 `assertEquals` calls. In the space provided, for each `assertEquals` call write:

- **P** if the `assertEquals` passes,
- **F** if the `assertEquals` fails, and
- **N** if the `assertEquals` call is not reached because a previous `assertEquals` failed.
To the right of each failure, write the actual value of the 2nd parameter in the space provided.

import junit.framework.TestCase;
public class StoreTester extends TestCase {

    public void test1() {
        Item i = new Item("soup", "34567");
        i.setPrice(33);
        i.discount((double)1/3);
        assertEquals(22, i.getPrice());
    }

    public void test2() {
        Store s1 = new Store("Oakville");
        Item i2 = new Item("soup", "34567");
        Item i1 = new Item("yogurt", "33311");
        s1.setItem(i1);
        s1.setOtherItem(i2);
        assertEquals("Oakville, soup, 34567, yogurt, 33311", s1.toString());
    }

    public void test3() {
        Store s1 = new Store("Oakville");
        Store s2 = new Store("Toronto");
        Item i2 = new Item("soup", "34567");
        Item i1 = new Item("yogurt", "33311");
        s1.setItem(i1);
        s1.setOtherItem(i2);
        i2.setPrice(33);
        i1.setPrice(95);
        assertEquals("45", i2.getBarcode().substring(1, 3));
        s2.setOtherItem(i1);
        assertEquals("soup", i2.getBarcode(), s2.getBarcode());
    }
}
public void test4() {
    Item i2 = new Item("soup", "34567");
    Item i1 = new Item("yogurt", "33311");
    i1.setPrice(55);
    i2.setPrice(95);
    
    assertEquals(150,
                Store.total(i1.getPrice(), i2.getPrice()));

    i1 = new Item("spaghetti", "92345");
    
    assertEquals(3, Item.getCount());

    i1.setPrice(65);
    
    assertEquals(Store.total(Store.total(i1.getPrice(),
                                i2.getPrice())),
                 i1.getPrice(), 160);
}
Use this page for rough work.

You can tear this page off, but we will collect it. You must fill in your student number if you tear it off, and you will lose 20% if you keep this page.