

1. True/False [10 pts]

Answer the following questions True or False. Be sure to write out True and False, NOT just a T or F.

- 3** (a) Each instance of a public class that is created receives its own individual copy of all the instance variables and static variables declared within the class.
- 3** (b) If you do not write a constructor for a class, java will supply a default constructor.
- 3** (c) All the case values in a switch statement must be constants.
- 3** (d) The body of a while loop is always executed at least once.
- 3** (e) A java class can implement more than one interface.

2. Accessors and Mutators [16 pts]

Write a class named `Rectangle`. The class must provide the following functionality:

- a) Declare two integer instance variables `width` and `height` with proper visibility to protect them from being seen outside the class.
- b) Write a constructor that takes two integer parameters and uses them to initialize the instance variables.
- c) Write a constructor with no parameters that initializes `width` to 10 and `height` to 20.
- d) Write an accessor method for the width and a mutator method for the height.

(Space for Problem 1 continued)

3. Iteration [20 pts]

a) Write a method `foo` that takes two positive integer parameters and prints the first the number of times of its value, and then the prints the second the number of times of its value. For instance,

`foo(5,3)` prints 55555333

`foo(2,4)` prints 224444

```
public void foo(int p1, int p2)
{
```

```
}
```

b) Write a method `bar` that takes two positive integer parameters, `hi` and `lo` (assume the first is greater than the second), and prints out the countdowns from `hi` to 1, 2, ... all the way to `lo`, such as is shown in the following examples:

`bar(5,3)` prints

```
54321
5432
543
```

`bar(7,4)` prints

```
7654321
765432
76543
7654
```

`bar(5,2)` prints

```
54321
5432
```

```
public void bar(int hi, int lo)
{
```

```
}
```

4. Static and Tracing [12 pts]

What is printed by the following code.

```
public class ABC {
    private int A=0;
    private static int B=0;
    private int C=0;

    public void increment() {
        int C=0;
        A++;
        B++;
        C++;
    }

    public void display() {
        System.out.println(A+" "+B+" "+C);
    }

    public static void main(String[] args){
        ABC x = new ABC();
        ABC y = new ABC();

        x.increment();
        y.increment();
        x.increment();

        x.display();
        y.display();
    }
}
```

Output:

5. Strings, Conditionals and Iteration [16 pts]

Write a method `pickOutChars` that takes two `char` parameters and a `String` parameter. It should print out occurrences of either of the two characters in the order they appear in the `String`.

So for the call

```
pickOutChars('e','T',"This Text Freezes Me To No End.");
```

the output is

```
TTeeeeeT
```

Hint: Remember the useful `String` method

```
char charAt(int index)
```

```
void pickOutChars(char first, char second, String str)
{
```

```
}
```

6. Visibility and Scope [14 pts]

Consider the Dog and Cat classes below. At each commented position, determine whether the access operation is legal or not. If illegal, give brief reason why. Write “Yes” or “No” in the numbers that follow the code (one for each Position) and explain the illegality for each “No” response.

```
public class Dog{
    private String name;
    private Cat chaseObject;
    public Dog(Cat c, String n) {
        this.name = n;//_____Position 1
        chaseObject=c;
    }//end constructor Dog

    public void goDog(){
        System.out.println(chaseObject.name);//_____Position 2
        chaseObject.run();//_____Position 3
    }//end method goDog

    public static void main(String[] args) {
        Cat c = new Cat("Fluffy");
        Dog d = new Dog(c,"Brutus");
        goDog(); //_____Position 4
        System.out.println(Cat.LIVES);//_____Position 5
        System.out.println(chaseObject.lives);//_____Position 6
    }//end main method
}//end class Dog

public class Cat{
    private String name;
    public static final int LIVES = 9;
    public int lives;

    public Cat(String n){
        name=n;
        lives=LIVES;//_____Position 7
    }//end constructor Cat

    public void run(){
        /*some code here*/
    }//end method run
}//end class Cat
```

1. Legal?

2. Legal?

3. Legal?

4. Legal?

5. Legal?

6. Legal?

7. Legal?

7. EXTRA CREDIT [5 pts]

Consider the java class below.

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

public class ExamPanel extends JPanel
{
    private JLabel label;
    private JTextField text;
    private JButton button;
    private String holder = "near";

    public ExamPanel()
    {
        button = new JButton("GO");
        text = new JTextField (6);
        label = new JLabel ("Up");
        text.addActionListener (new Listener1());
        button.addActionListener (new Listener2());
        add(label);
        add(text);
        add(button);
        setBackground (Color.white);
    }

    public void changeIt(String str) {
        label.setText(str+holder);
    }

    private class Listener1 implements ActionListener
    {
        public void actionPerformed (ActionEvent event)
        {
            holder = "far";
        }
    }

    private class Listener2 implements ActionListener
    {
        public void actionPerformed (ActionEvent event)
        {
            changeIt(text.getText());
        }
    }
}
```

Also assume that there is another “driver” class that has a main method that creates a JFrame, instantiates this panel, and does all the needed operations such as setting it visible, adding, etc.

Suppose that the user is running the code and performs the following operations on the interface. First, she types the string “Where?” (without any quotes) into the text field and then presses Return. Next, she clicks the button.

Draw how the interface will look after these operations. Assume that the parent window is short and wide and that the components fit in left-to-right. (This is an all-or-nothing question.)