

SURNAME: ..... FORENAMES: .....

**Question 1 (10 marks)****Part A (5 marks)**

The contents of a simple HTML file is given below. The page contains an applet called `Test`, which has been compiled, and the class file for this applet is in the same directory as this HTML file.

The HTML contains 5 errors. Locate each of the 5 errors in the HTML below, and write the correction underneath the line with the error. You will lose marks for making unnecessary changes.

```
<html>

<head>

<title>Terms Test</title>

<head>

<body>

<h1>Here is an applet:</h1>

<applet class="Test.class" width=100 height=100>

What do you think?

<hr>

The answers to the test can be
found <a = "http://www.cs.auckland.ac.nz/ans.html">here</a>.

</body>

</html>
```

**(5 marks)****CONTINUED**

SURNAME: ..... FORENAMES: .....

**Part B (5 marks)**

There are 5 syntax errors in the following code. Locate each of the errors in the code below, and write the correction in the space provided underneath the line with the error. You will lose marks for making unnecessary changes.

```
import java.awt.*;

import java.applet.*;

/**

Check for syntax errors

@author Department of Computer Science

@version 30 April 2001

//

public class CoinToss extends applet {

    public void init () {

        int rand == (int)(Math.random * 3);

        if (rand == 1)

            System.out.println("Heads");

        else if (rand == 2)

            System.out.println("Tails");

        else

            System.out.println("On its side!")

    }

}
```

**(5 marks)****CONTINUED**

SURNAME: ..... FORENAMES: .....

**Question 2 (15 marks)****Part A (5 marks)**

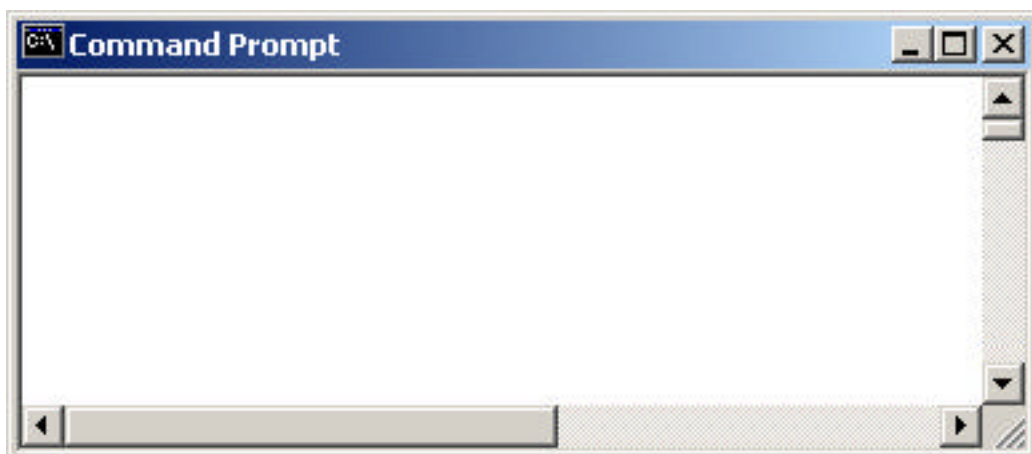
Complete the code segment below so that it prints out a random number between 10 and 20 inclusive. That is, the smallest possible random number will be 10, and the largest possible random number will be 20.

```
public void init() {  
    int randomNumber;  
  
    System.out.println(randomNumber);  
}
```

*(5 marks)***Part B (10 marks)**

What is the output of the following code segment:

```
int i;  
double d;  
  
i = 5;  
d = 5;  
  
System.out.println(i - d);  
System.out.println(i + (d + ""));  
System.out.println((i + d) + "");  
  
System.out.println((int)(i * d) / (i + d) - i);  
System.out.println((int)d + i);
```

*(10 marks)*

CONTINUED

SURNAME: ..... FORENAMES: .....

**Question 3 (5 marks)**

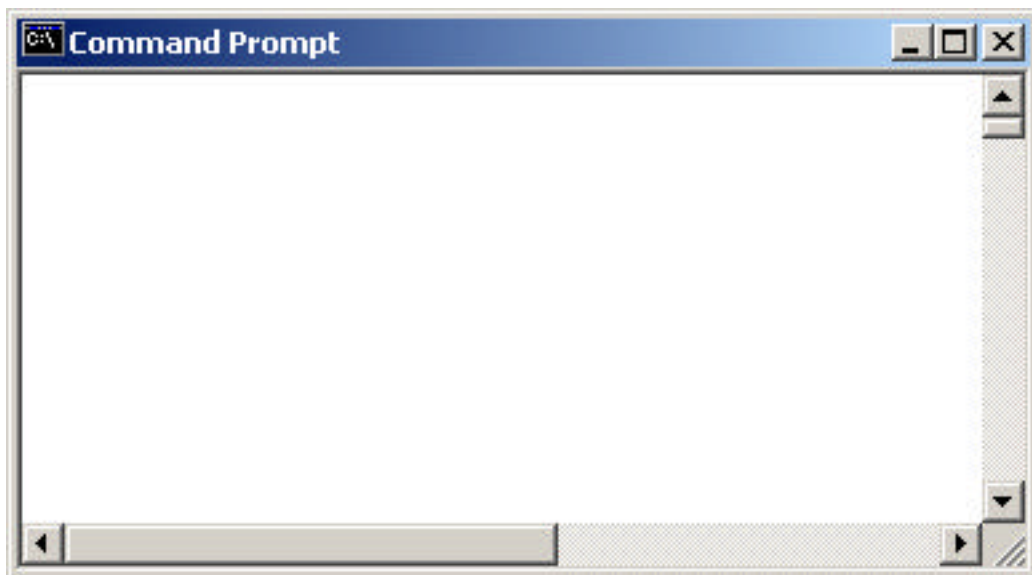
What is the output from the following program?

```
import java.awt.*;
import java.applet.*;

public class Variables extends Applet {
    int aInt = 1;
    int bInt = 2;
    int cInt = 0;

    public void paint(Graphics g) {
        int aInt = 200;
        System.out.println(aInt + bInt);
        System.out.println(cInt);
    }

    public void init() {
        int aInt = 10;
        int bInt = 20;
        cInt++;
        System.out.println(aInt);
        System.out.println(bInt);
        System.out.println(cInt);
    }
}
```

*(5 marks)*

SURNAME: ..... FORENAMES: .....

**Question 4 (10 marks)****Part A (5 marks)**

Evaluate the following boolean expressions to `true` or `false`.

```
int i = 10;  
int j = 20;  
int k = 30;
```

i) `(!(i <= 10) || ((k - j) != i))`

ii) `i + k < 35 || k < j + i || j < k`

(5 marks)

**Part B (5 marks)**

Write a boolean expression for the following descriptions, where *value* is an integer variable.

i) *value* is greater than 31 and *value* is not equal to 37

ii) *value* is even or *value* is equal to 13 or *value* is greater than 17

(5 marks)

CONTINUED

SURNAME: ..... FORENAMES: .....

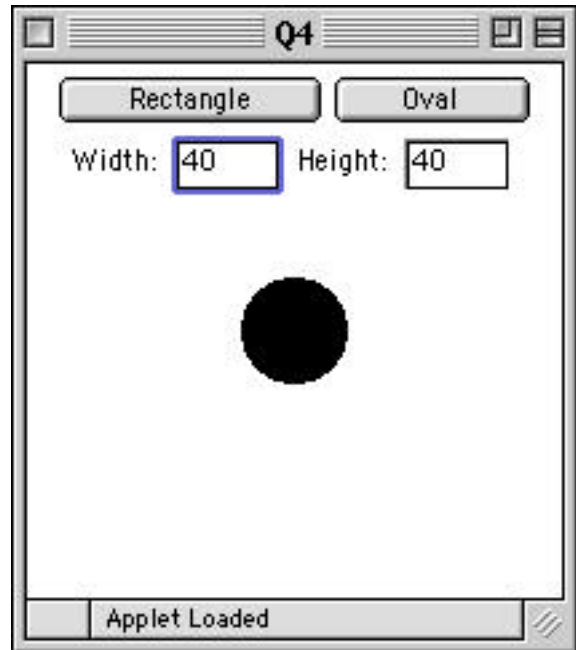
**Question 5 (15 marks)**

You need to complete the `paint()` method for an applet which draws ovals and rectangles centred at (100, 100).

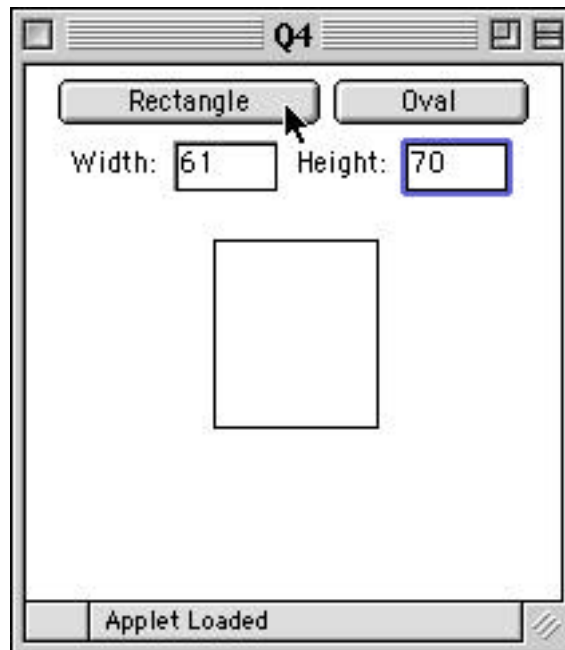
When the applet first starts, the `TextFields` both have the value 40. A filled oval is drawn with a width of 40 and a height of 40, and with its centre at (100, 100). The screen shot to the right shows the applet when it first starts. The applet window is 200 pixels wide and 200 pixels high:

The user can change the size of the shape by entering new width or height values in the `TextFields`. The new shape is drawn when one of the two buttons is pressed, and is always centred at (100, 100).

The shape will either be drawn in outline or filled in according to the following rule: if the sum of the width and height values is an even number, then the shape is drawn filled in. If the sum of the width and height values is an odd number, then the shape is drawn in outline. When the applet first starts, the oval is drawn filled in because the sum of the width and height values is 80, which is even.



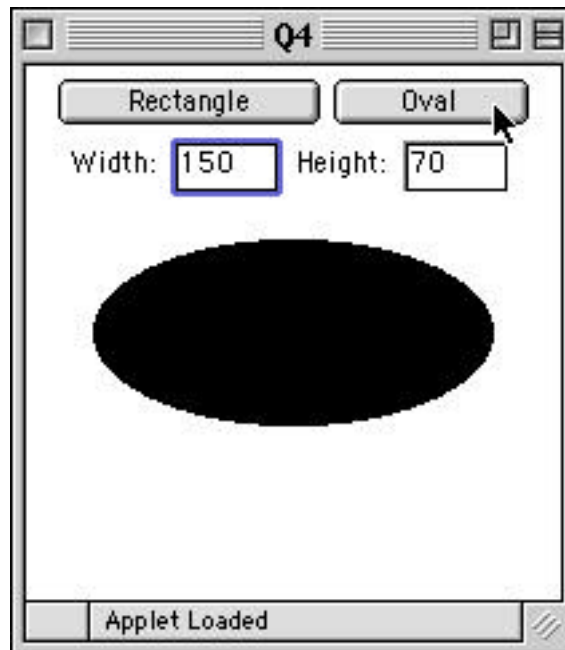
For example, if the user changes the width and height values to 61 and 70 respectively, and then presses the "Rectangle" button, the rectangle is drawn in outline, with its centre still at (100, 100). This is shown in the screen shot below:



Whenever either of the two Buttons is pressed, the values from the `TextFields` must be examined to work out where to draw the shape and whether to draw the shape filled in or in outline.

SURNAME: ..... FORENAMES: .....

If the user changes the width from 61 to 150, and then presses the "Oval" button, the oval is drawn filled in because the sum of the width and height values is even. This is shown in the screen shot below:



Here is the source code for the applet. You must complete the `paint()` method in the space provided:

```
import java.awt.*;
import java.applet.*;
import java.awt.event.*;

public class Q4 extends Applet implements ActionListener {
    Button bRect, bOval;
    TextField tWidth, tHeight;
    boolean isRect;

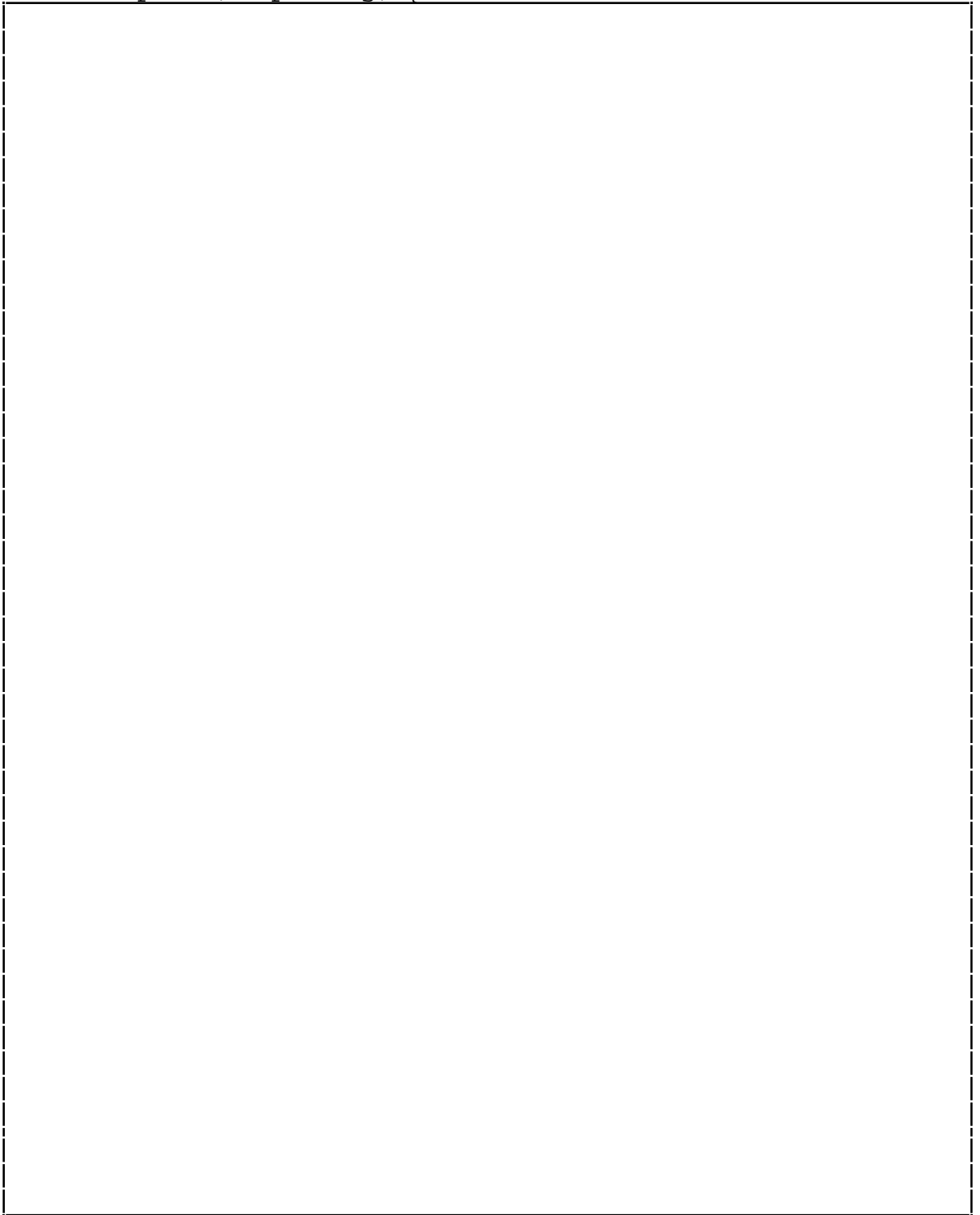
    public void init() {
        isRect = false;
        bRect = new Button("Rectangle");
        bOval = new Button("Oval");
        bRect.addActionListener(this);
        bOval.addActionListener(this);
        tWidth = new TextField("40");
        tHeight = new TextField("40");
        add(bRect);
        add(bOval);
        add(new Label("Width:"));
        add(tWidth);
        add(new Label("Height:"));
        add(tHeight);
    }

    public void actionPerformed(ActionEvent e) {
        if (e.getSource() == bRect)
            isRect = true;
        else
            isRect = false;
        repaint();
    }
}
```

**CONTINUED**

SURNAME: ..... FORENAMES: .....

```
public void paint(Graphics g) {
```



```
}
```

*(15 marks)***CONTINUED**



SURNAME: ..... FORENAMES: .....

**Question 6 (15 marks)****Part A (10 marks)**

What is the output from the following program?

```
import java.applet.*;

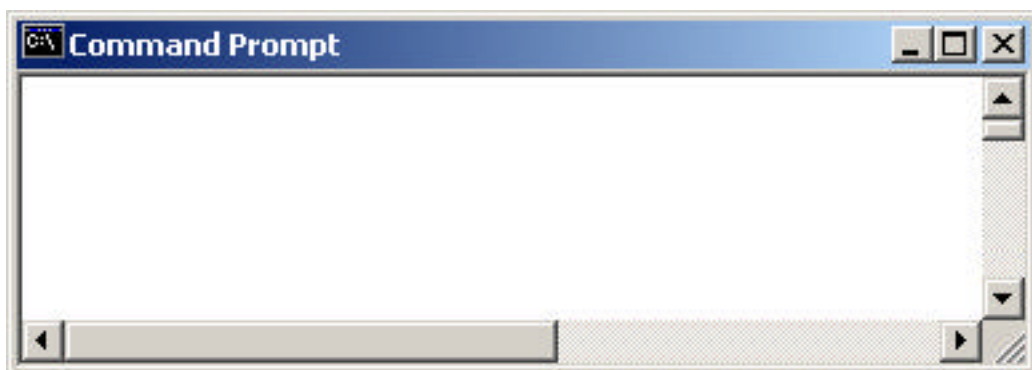
public class Print extends Applet {

    public void init() {
        int a;
        int b;
        a = 10;
        b = 5;
        printTwo(b, a, 15);
    }

    private void printTwo(int a, int b, int c) {
        printOne(a, b, c);
        printOther(a, b, c);
    }

    private void printOne(int x, int y, int z) {
        int one;
        if (x >= y)
            one = x;
        else
            one = y;
        if (one < z)
            one = z;
        System.out.println(one);
    }

    private void printOther(int i, int j, int k) {
        if (j <= i && i <= k)
            System.out.println(i);
        else if (i <= j && j <= k)
            System.out.println(j);
        else
            System.out.println(k);
    }
}
```

*(10 marks)*

CONTINUED

SURNAME: ..... FORENAMES: .....

**Part B (5 marks)**

What is the output from the following program?

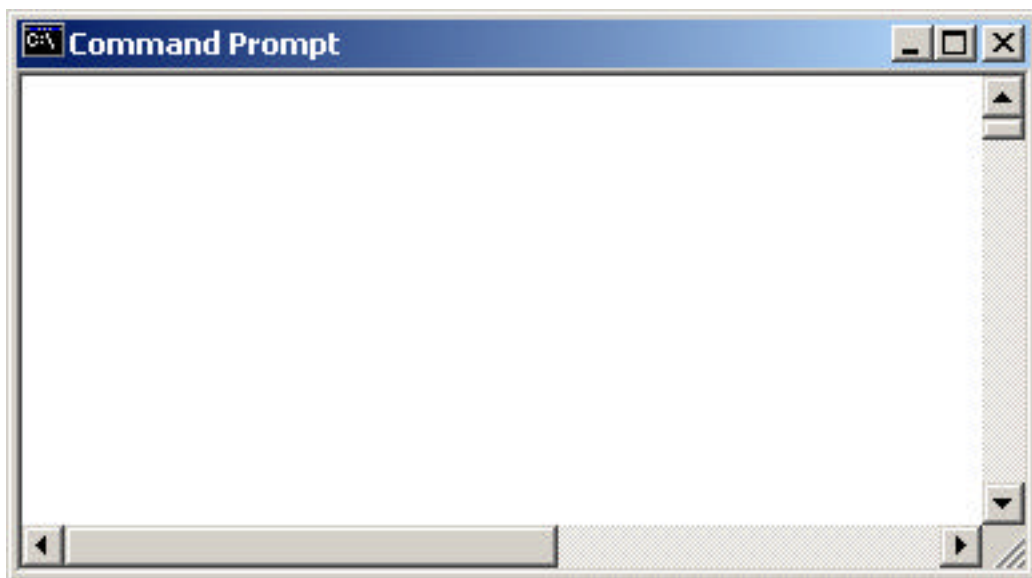
```
import java.applet.*;

public class Maths extends Applet {

    public void init() {
        int a;
        int b;
        int c;
        a = 5;
        b = 3;
        a = first(a, b);
        c = second(b, a);
        System.out.println(a);
        System.out.println(b);
        System.out.println(c);
    }

    private int first(int x, int y) {
        return x+y;
    }

    private int second(int y, int x) {
        return x-y;
    }
}
```

*(5 marks)*

SURNAME: ..... FORENAMES: .....

**Question 7 (10 marks)****Part A (5 marks)**

Consider the following switch statement:

```
switch(who) {  
  case 1:  
  case 2:  
    System.out.println("For Mum");  
    break;  
  case 3:  
  case 4:  
    System.out.println("For Dad");  
}
```

Which of the two following code segments is equivalent to the switch statement above? (In other words, which one gives exactly the same output for all possible values of who?) Place a tick in the box next to the equivalent code segment.

a)    `if (who == 1 || who == 2)`  
          `System.out.println("For Mum");`  
      `else if (who == 3 || who == 4)`  
          `System.out.println("For Dad");`

(a)

☐

b)    `if (who == 1);`  
      `else if (who == 2)`  
          `System.out.println("For Mum");`  
      `else if (who == 3);`  
      `else if (who == 4)`  
          `System.out.println("For Dad");`

(b)

☐**(5 marks)**

SURNAME: ..... FORENAMES: .....

**Part B (5 marks)**

Consider the following switch statement:

```
switch(dayOfWeek){  
  case 3:  
    System.out.println("I go to a lab");  
  case 5: case 6:  
    System.out.println("I go to a lecture");  
    break;  
  default:  
    System.out.println("I can stay at home");  
}
```

What is the output if dayOfWeek has the value 3?

What is the output if dayOfWeek has the value 4?

What is the output if dayOfWeek has the value 5?

**(5 marks)**

SURNAME: ..... FORENAMES: .....

**Question 8 (10 marks)**

Consider the following code for the definition of a Face class:

```
import java.awt.*;

public class Face {
    private int xPos;
    private int yPos;
    private int size;

    public Face() {
        xPos = 75;
        yPos = 0;
        size = 150;
    }

    public Face(int required) {
        if (required == 50) {
            xPos = 75;
            yPos = 30;
            size = 50;
        }
        else if (required == 100) {
            xPos = 0;
            yPos = 75;
            size = 100;
        }
        else if (required == 150) {
            xPos = 150;
            yPos = 75;
            size = 150;
        }
    }

    public Face(int x, int y) {
        xPos = x;
        yPos = y;
        size = 50;
    }

    public boolean equals(Face other) {
        return other.xPos == xPos && other.yPos == yPos
            && other.size == size;
    }

    private void drawEye(Graphics g, int xCentre, int yCentre) {
        g.fillOval(xCentre-size/8, yCentre-size/8, size/4, size/4);
    }

    public void draw(Graphics g) {
        g.setColor(Color.yellow);
        g.fillOval(xPos, yPos, size, size);
        g.setColor(Color.black);
        g.drawOval(xPos, yPos, size, size);
        drawEye(g, xPos + size/3, yPos + size/3);
        drawEye(g, xPos + size/3 + size/3, yPos + size/3);
        g.drawArc(xPos + size/2 - size/3, yPos + size/3 - size/6,
            size/3*2, size/3*2, 0, -180);
    }
}
```

**CONTINUED**

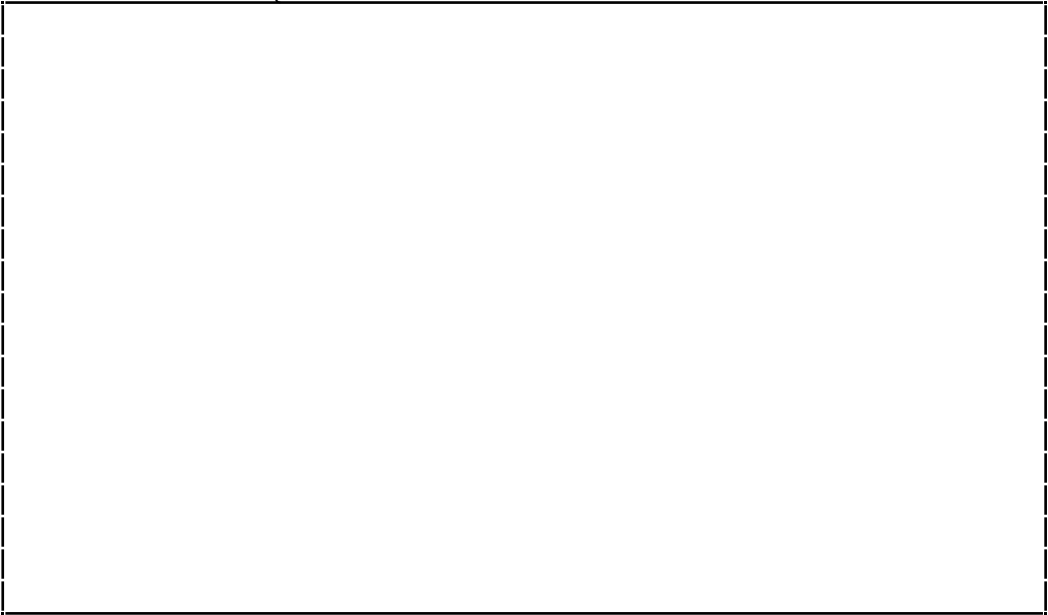
SURNAME: ..... FORENAMES: .....

Note that this Face class provides three constructors - one which takes no parameters, one which takes a single int parameter, and one which takes two int parameters.

Consider the following applet, FaceTester, which uses this Face class. The `init()` method is incomplete. Read the questions which follow this code for instructions on how to complete the `init()` method.

```
import java.awt.*;
import java.awt.event.*;
import java.applet.*;

public class FaceTester extends Applet {
    private Face face1, face2, face3, face4, face5;

    public void init() {
        
        areTheyEqual();
    }

    public void paint(Graphics g) {
        face1.draw(g);
        face2.draw(g);
        face3.draw(g);
    }

    public void areTheyEqual(){
        if (face2.equals(face4))
            System.out.println("Same Face.");
        else
            System.out.println("Different Face.");
        if (face2.equals(face5))
            System.out.println("Same Face.");
        else
            System.out.println("Different Face.");
    }
}
```

**CONTINUED**

SURNAME: ..... FORENAMES: .....

**Part A (5 marks)**

The FaceTester applet declares the following 5 instance variables of type Face:

```
private Face face1, face2, face3, face4, face5;
```

Complete the `init()` method of the FaceTester applet (in the space provided on the previous page) by writing code to construct 4 Face objects. You can use any of the constructors provided in the Face class on page 12. The faces you construct must have the correct sizes, and must be assigned to the variables as specified below:

- a) `face1` – a face with diameter 50,
- b) `face2` – a face with diameter 100,
- c) `face3` – a face with diameter 150,
- d) `face4` – another face with diameter 100.

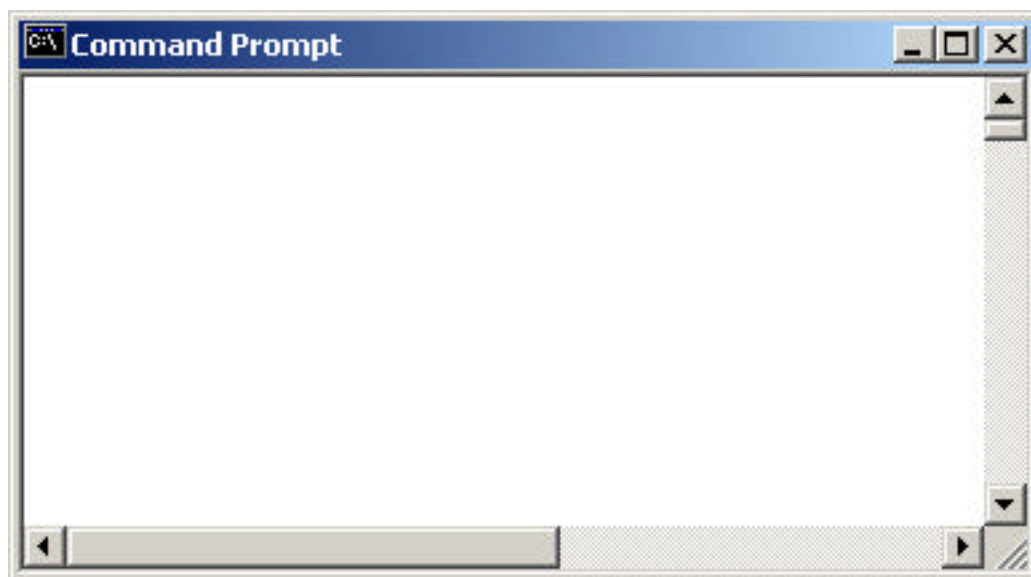
The reference variable `face2` should now be pointing at a Face object.

- e) Include code in the `init()` method to make the variable `face5` refer to the same Face object that `face2` refers to.

**(5 marks)**

**Part B (5 marks)**

Now that the 5 variables `face1`, `face2`, `face3`, `face4` and `face5` refer to Face objects, what is the output of the method `areTheyEqual()` which is called from the `init()` method?



**(5 marks)**

SURNAME: ..... FORENAMES: .....

**Question 9 (10 marks)**

Complete the `printLengths()` method below which is passed a `Vector` of `Strings` as parameter and prints out the length of every `String` in the `Vector` on a separate line.

```
public void printLengths(Vector v) {
```

```
}
```

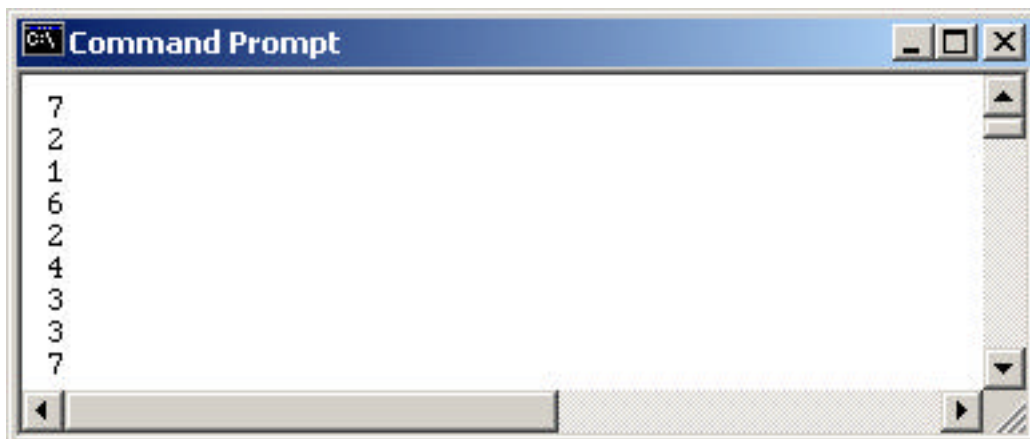
For example, consider the `Vector` `words`, which is created as follows :

```
Vector words;  
words = new Vector();  
words.addElement("success");  
words.addElement("is");  
words.addElement("a");  
words.addElement("matter");  
words.addElement("of");  
words.addElement("luck");  
words.addElement("ask");  
words.addElement("any");  
words.addElement("failure");
```

If this `Vector` is passed to the method `printLengths()` as below:

```
printLengths(words);
```

the output should be as follows:



(10 marks)

CONTINUED



SURNAME: ..... FORENAMES: .....

**OVERFLOW PAGE**