

THE UNIVERSITY OF AUCKLAND

FIRST SEMESTER, 2005
Campus: City

COMPUTER SCIENCE

Principles of Programming

(Time Allowed: TWO hours)

Surname

Forenames

Student ID

Login name(UPI)

NOTE:

- Attempt **ALL** questions
- Answer the multiple choice questions in section A by circling the correct answer
- Write the answer to the questions in section B in the space provided

Examiner to complete:

Question	Mark
1 – 20	(/40)
21	(/8)
22	(/10)
23	(/12)
24	(/5)
25	(/10)
26	(/10)
27	(/5)
Total	(/100)

CONTINUED

SECTION A: MULTIPLE CHOICE QUESTIONS

Each question in this section is worth 2 marks. Circle the letter corresponding to your choice. There is only one correct answer for each question.

1. What is the output produced by the following code?

```
System.out.println("1 + 2" + 3) ;
```

- (a) 6
- (b) 123
- (c) 1 + 2 + 3
- (d) 1 + 23
- (e) 3 + 3

2. What is the output produced by the following code?

```
System.out.println(3 / 2 * 5 % 3);
```

- (a) 2
- (b) 3
- (c) 0
- (d) 1
- (e) None of the above

3. Which of the following is the correct way to declare a symbolic constant?

- (a) `int final size = 10;`
- (b) `final int size = 10;`
- (c) `int final SIZE = 10;`
- (d) `final int SIZE = 10;`
- (e) `int size = 10;`

ID.....

4. Which one of the identifiers shown below follows the style convention for naming classes?

- (a) countNumbers
- (b) Print_Alphabet
- (c) Spaceinvaders
- (d) JellyBeans
- (e) myfirstprogram

5. Which one of the five sections of code below does NOT produce the following output:

output: AAAA

- (a)

```
System.out.print("output: ");
for (int i = 1; i < 5; i++) {
    System.out.print("A");
}
```
- (b)

```
System.out.print("output: ");
for (int i = 1; i < 5 && i != 0; i++) {
    System.out.print("A");
}
```
- (c)

```
System.out.print("output: ");
for (int i = 1; i < 5; ) {
    i = i + 1;
    System.out.print("A");
}
```
- (d)

```
System.out.print("output: ");
for (int i = 1; i < 10; i = i + 2) {
    System.out.print("A");
}
```
- (e)

```
System.out.print("output: ");
for (int i = 5; i > 1; i--) {
    System.out.print("A");
}
```

6. Given the methods declared below, which statement would result in the value 10 being printed to the screen?

```
private int bar(int i) {  
    return i * 2;  
}
```

```
private int foo(int i) {  
    return i + 2;  
}
```

- (a) `System.out.println(bar(4));`
- (b) `System.out.println(foo(bar(4)));`
- (c) `System.out.println(foo(4));`
- (d) `System.out.println(bar(foo(4)));`
- (e) `System.out.println(foo(4) + bar(4));`

7. What is the output produced by the following code?

```
int a = 3;  
int b = 7;  
if (a > b) {  
    if (a < b) {  
        System.out.print("X");  
    } else {  
        System.out.print("Y");  
    }  
} else {  
    System.out.println("Z");  
}
```

- (a) X
- (b) Y
- (c) Z
- (d) YZ
- (e) XZ

8. Which declaration of variables will ensure that the boolean expression:

`(a || b) && (a && !c)`

evaluates to true?

- (a) `boolean a = false, b = true, c = true;`
- (b) `boolean a = false, b = false, c = false;`
- (c) `boolean a = true, b = true, c = true;`
- (d) `boolean a = false, b = true, c = false;`
- (e) `boolean a = true, b = false, c = false;`

9. What is the output produced by the following code?

```
String word1 = "FANTASTIC";
char c1;
int number = 0;
int counter = word1.length() - 1;

while (counter > 0) {
    c1 = word1.charAt(counter);
    if (c1 == 'T') {
        counter = counter - 2;
    } else {
        counter = counter - 1;
        number++;
    }
}
System.out.println("output: " + number);
```

- (a) output: 5
- (b) output: 6
- (c) output: 8
- (d) output: 4
- (e) output: 7

ID.....

10. The definition of a class, Numbers, is shown below:

```
public class Numbers {
    private static int number;
    private int amount;

    public Numbers(int num) {
        amount = num;
        number = amount * 10;
    }

    public void change(int amt) {
        number = number + amount;
        amount = amt;
    }

    public String toString() {
        return "number: " + number + ", amount: " + amount;
    }
}
```

What is the output produced by the following code?

```
Numbers ex1 = new Numbers(8);
Numbers ex2 = new Numbers(5);
ex1.change(3);
ex2.change(1);
System.out.println(ex1.toString());
System.out.println(ex2.toString());
```

- (a) number: 143, amount: 3
 number: 143, amount: 1
- (b) number: 54, amount: 3
 number: 54, amount: 1
- (c) number: 88, amount: 11
 number: 55, amount: 6
- (d) number: 88, amount: 3
 number: 55, amount: 1
- (e) number: 63, amount: 3
 number: 63, amount: 1

ID.....

11. Which one of the Java exceptions shown below is produced when the following code is executed?

```
String word = "012345678";
int sum = 0;
int counter = 0;
char c;
while (counter <= word.length()) {
    c = word.charAt(counter);
    sum = sum + Integer.parseInt("" + c);
    counter++;
}
System.out.println(sum);
```

- (a) NumberFormatException
 - (b) NullPointerException
 - (c) StringIndexOutOfBoundsException
 - (d) IOException
 - (e) ArrayIndexOutOfBoundsException
12. Consider the following code fragment:

```
int[] x = {0, 1, 2, 3};

int temp;
int i = 0;
int j = x.length - 1;

while (i < j) {
    temp = x[i];
    x[i] = x[j];
    x[j] = 2 * temp;
    i++;
    j--;
}
```

After this code is executed, array x contains the values:

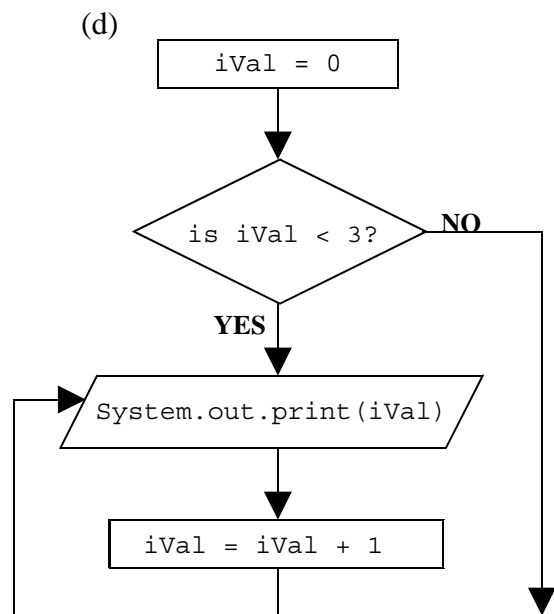
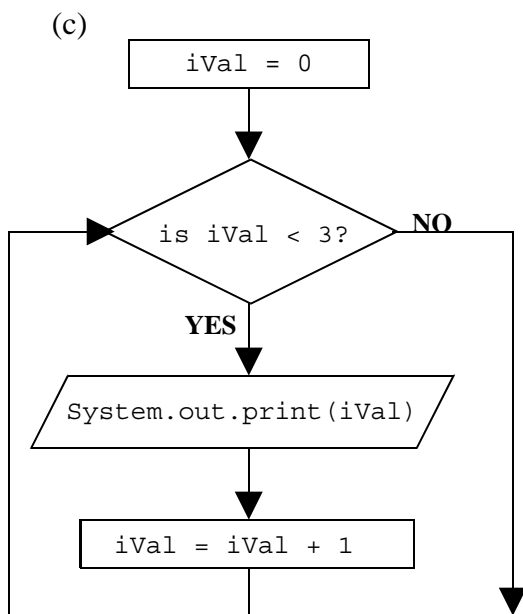
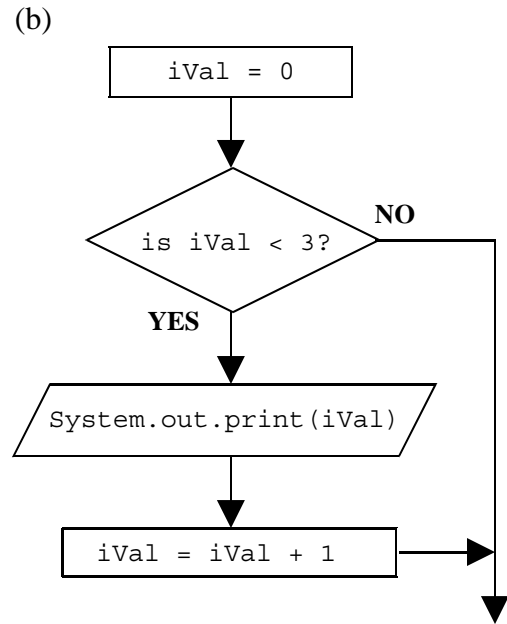
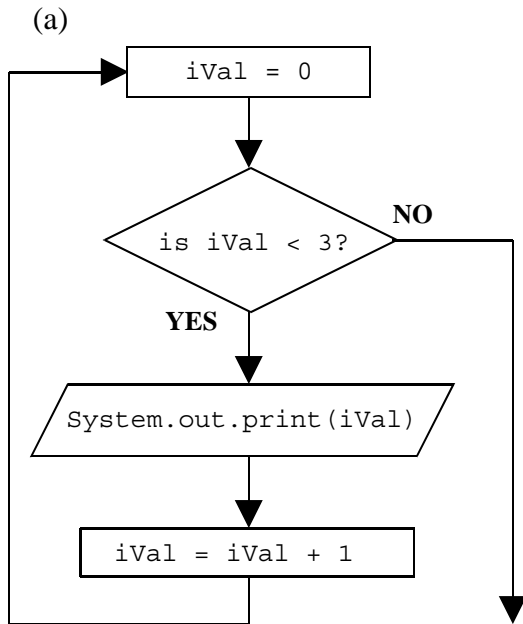
- (a) {3, 2, 2, 0}
- (b) {0, 1, 2, 3}
- (c) {3, 2, 1, 0}
- (d) {0, 2, 4, 6}
- (e) {6, 4, 2, 0}

ID.....

13. Consider the following code fragment:

```
int iVal = 0;
while (iVal < 3) {
    System.out.print(iVal);
    iVal = iVal + 1;
}
```

Which of the following flowcharts represents the logic of this code?



- (e) None of the above

ID.....

14. The following code checks to see if a number is within a given range. MIN_NUM and MAX_NUM are both integers, and MAX_NUM is larger than MIN_NUM. i is also an integer, and valid is a boolean.

```
if ((i < MIN_NUM) || (i > MAX_NUM)) {  
    valid = false;  
} else {  
    valid = true;  
}
```

Which of the following pieces of code will give exactly the same result as the code above?

- (a)

```
if ((i >= MIN_NUM) && (i <= MAX_NUM)) {  
    valid = true;  
} else {  
    valid = false;  
}
```
- (b)

```
if ((i > MIN_NUM) && (i < MAX_NUM)) {  
    valid = true;  
} else {  
    valid = false;  
}
```
- (c)

```
if ((i > MIN_NUM) || (i < MAX_NUM)) {  
    valid = false;  
} else {  
    valid = true;  
}
```
- (d)

```
if ((i <= MIN_NUM) || (i >= MAX_NUM)) {  
    valid = false;  
} else {  
    valid = true;  
}
```
- (e) None of the above

15. The following code processes two arrays of integers.

```
int[] nums1 = {5, 18, 7, 14};
int[] nums2 = {29, 11, 16, 9, 21};

int count = 0;

for (int i = 0; i < nums1.length; i++) {
    for (int j = 0; j < nums2.length; j++) {
        if (nums1[i] > nums2[j]) {
            count++;
        }
    }
}
```

What will the value of `count` be at the end of this?

- (a) 2
 - (b) 3
 - (c) 5
 - (d) 6
 - (e) 20
16. In the following code, `nums` is an array of integers:

```
int[] nums = {3, 6, 13, 27, 12, 2, 7};

int i = 0;
int sum = 0;

while (i < nums.length) {
    sum = sum + nums[i];
    i = i + 1;
}
```

What is the value stored in the variable `sum` after the code fragment is executed?

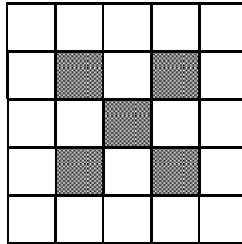
- (a) the sum of all the values of `i`
- (b) the sum of all elements in the array
- (c) the sum of all elements in the array except for the first element
- (d) the sum of all elements in the array except for the last element
- (e) there will be an `ArrayIndexOutOfBoundsException` exception

ID.....

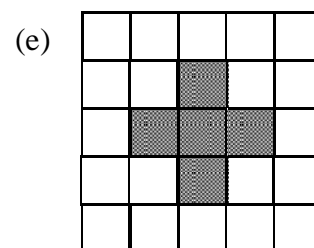
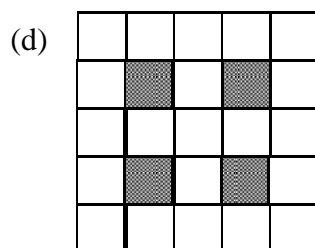
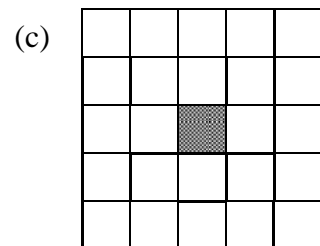
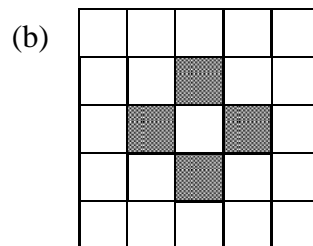
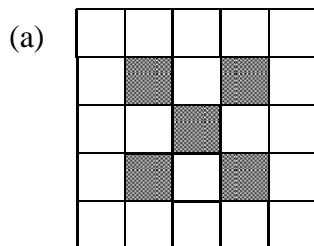
17. The following two rules specify the conditions for a given cell in one generation to be alive in the next generation of the Game of Life simulation:

- if the cell is currently dead, yet it has exactly three live neighbours
- if the cell is currently alive, and it has either two or three live neighbours

Given the following starting configuration (the filled in cells are alive):



which of the following diagrams represent the configuration of cells in the very next generation:



18. A *palindrome* is a sequence that reads the same left to right as it does right to left. For example:

121
3443
12321

are all palindromes.

The following code attempts to recognise the palindrome pattern in an array of integers. If the array contains the palindrome pattern, the value of the variable `isPalindrome` should be set to `true`. Otherwise, the value of the variable `isPalindrome` should be left unchanged as `false`.

In this code, `nums` is the array of numbers being tested.

```
int iEnd = nums.length - 1;
int i = 0;

boolean isPalindrome = false;

while ((i < iEnd) && (nums[i] == nums[iEnd])) {
    i = i + 1;
    iEnd = iEnd - 1;
}

if ( XXXXXXXXXXXX ) {
    isPalindrome = true;
}
```

The missing condition in the `if` statement (XXXXXXXXXX) determines whether the palindrome pattern was found, and should be replaced by:

- (a) `i == iEnd`
- (b) `i <= iEnd`
- (c) `i < iEnd`
- (d) `i > iEnd`
- (e) `i >= iEnd`

19. Consider the following code fragment:

```
int[] x1 = {1, 2, 4, 7};
int[] x2 = {1, 2, 5, 7};

int i1 = x1.length - 1;
int i2 = x2.length - 1;
int count = 0;

while ((i1 > 0) && (i2 > 0)) {

    if (x1[i1] == x2[i2]) {
        count = count + 1;
        i1 = i1 - 1;
        i2 = i2 - 1;
    } else if (x1[i1] < x2[i2]) {
        i2 = i2 - 1;
    } else {
        i1 = i1 - 1;
    }
}
```

After the while loop finishes, the variable count contains what value?

- (a) 4
 - (b) 3
 - (c) 2
 - (d) 1
 - (e) 0
20. Consider the following code fragment:

```
int[] nums = {1, 2, 3, 4, 5, 6, 7};

printArray(nums);
shift(nums);
printArray(nums);
```

The `printArray()` method prints all of the elements in the array passed to it as a parameter. This method is defined as follows:

```
private void printArray(int[] nums) {
    for (int i = 0; i < nums.length; i++) {
        System.out.print(nums[i] + " ");
    }
    System.out.println();
}
```

The `shift()` method shifts all elements in the array one position towards the left, with the first element in the array being cycled around to the last position. The output of the code fragment above is given below:

```
1 2 3 4 5 6 7
2 3 4 5 6 7 1
```

Which of the following definitions of the `shift()` method is correct?

- (a)

```
private void shift(int[] nums) {
    int temp = nums[0];
    for (int i = 1; i < nums.length; i++) {
        nums[i] = nums[i + 1];
    }
    nums[nums.length - 1] = temp;
}
```
- (b)

```
private void shift(int[] nums) {
    int temp = nums[0];
    for (int i = 1; i < nums.length - 1; i++) {
        nums[i] = nums[i + 1];
    }
    nums[nums.length - 1] = temp;
}
```
- (c)

```
private void shift(int[] nums) {
    int temp = nums[0];
    for (int i = 0; i < nums.length - 1; i++) {
        nums[i + 1] = nums[i];
    }
    nums[nums.length - 1] = temp;
}
```
- (d)

```
private void shift(int[] nums) {
    int temp = nums[0];
    for (int i = nums.length - 1; i > 0 ; i--) {
        nums[i - 1] = nums[i];
    }
    nums[nums.length-1] = temp;
}
```
- (e)

```
private void shift(int[] nums) {
    int temp = nums[0];
    for (int i = 1; i < nums.length; i++) {
        nums[i - 1] = nums[i];
    }
    nums[nums.length - 1] = temp;
}
```

SECTION B

21. Draw the output produced by the following code.

The grid lines are not part of the output but are there to help you place the drawing in the correct position. The size of each square in the grid is 10 pixels by 10 pixels.

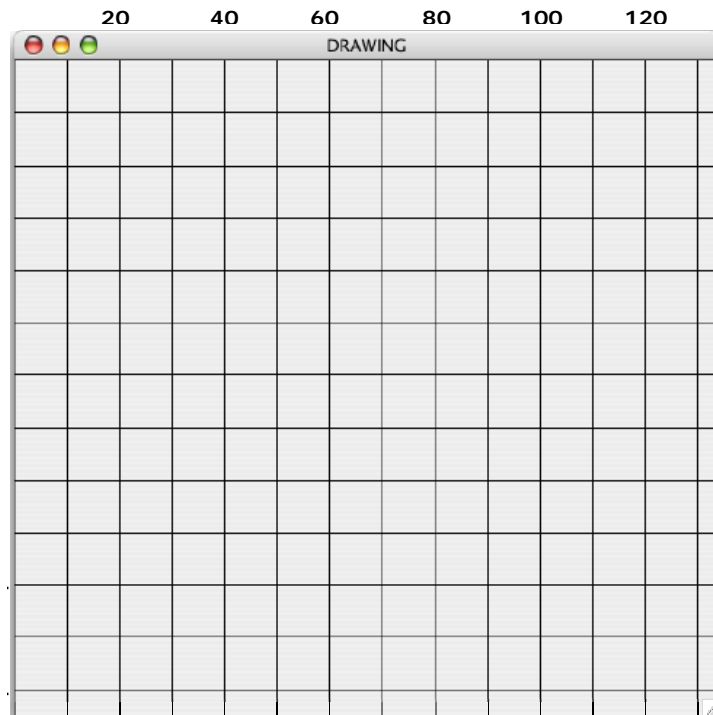
```
import java.awt.*;
import javax.swing.*;

public class DrawJPanel extends JPanel{

    public void paintComponent(Graphics g) {
        super.paintComponent(g);

        draw(g, 20, 20, 10, "good work");
        draw(g, 10, 40, 50, "bravo");
    }

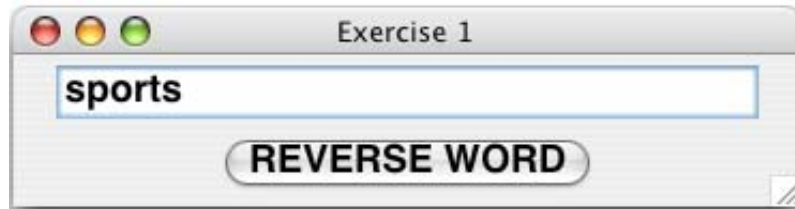
    private void draw(Graphics g, int size, int left, int top,
                      String word) {
        g.setColor(Color.BLACK);
        g.drawRect(left, top, size*2, size);
        g.drawRect(left, top, size, size);
        g.drawLine(left, top, left, top+size*4);
        g.drawString(word.toUpperCase(), left+size*2, top);
        g.drawOval(left - size, top+size*3, size, size);
    }
}
```



(8 marks)

22. The `JPanel` defined on the next page contains two components:
- a `TextField` which contains a word or phrase,
 - the 'REVERSE WORD' `Button`.

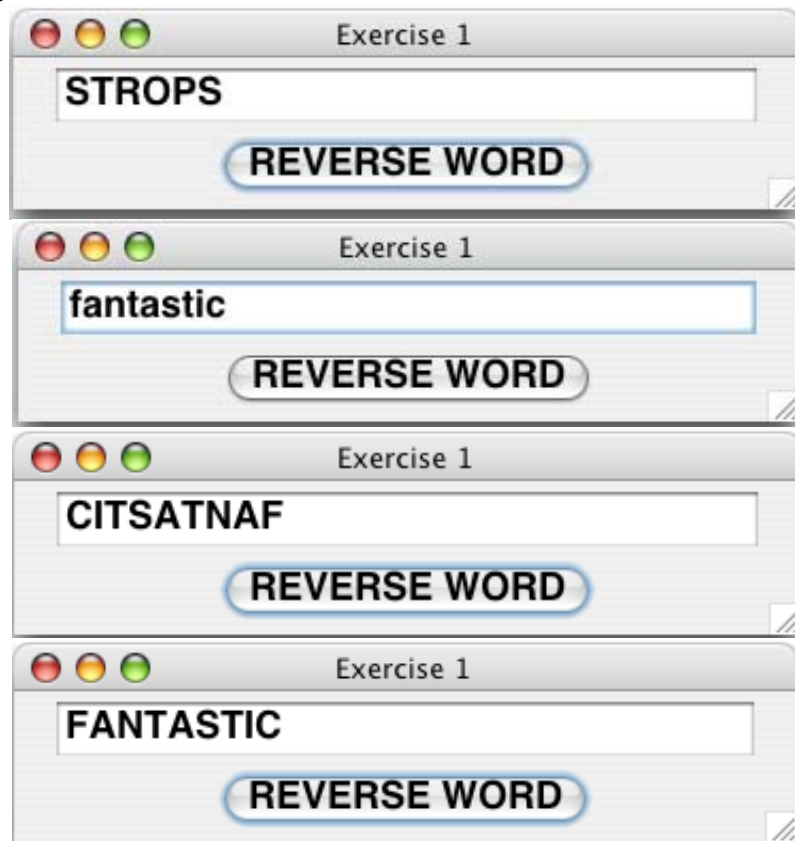
When the `JPanel` first appears it displays the string, "sports", in the `TextField`.



The `JPanel` behaves as follows: whenever the user presses the 'REVERSE WORD' button the word or phrase in the `TextField` is displayed in reverse order. The word is displayed in upper case characters.

You are required to complete the `JPanel` definition on the next page so that the `JPanel` behaves as described above.

The screenshots below show the `JPanel` in action. In the screenshots below the user has clicked the 'REVERSE WORD' `Button`, then the user has typed in the string, "fantastic" and then clicked the 'REVERSE WORD' `Button`. Lastly the user has clicked the 'REVERSE WORD' `Button` once again.




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

public class ReverseWordJPanel extends JPanel implements
    ActionListener {

    private JTextField reverseWordT;
    private JButton reverseB;

    public ReverseWordJPanel() {
        reverseWordT = new JTextField(20);
        reverseB = new JButton("REVERSE WORD");
        
        add(reverseWordT);
        add(reverseB);
    }

    public void actionPerformed(ActionEvent e) {
        
    }
}
```

(10 marks)

ID.....

23. Read the information given on the following three pages and complete the definition of the TaxiRide class below.

A TaxiRide instance represents a taxi ride. All TaxiRide instances are either after hours or not after hours.

When a TaxiRide object is constructed, four values are specified to the constructor method:

- a String which indicates the source suburb of the taxi ride,
- a String which indicates the destination suburb of the taxi ride,
- an int which indicates the distance of the taxi ride: the distance is given as a number of units, the higher the number of units the further the distance,
- a boolean which indicates whether the taxi ride is after hours or not.

A taxi ride costs \$3 per unit of distance if the ride is after hours and \$2 per unit of distance if the ride is not after hours. The cost per unit distance if the ride is after hours is given by the constant, PRICE_AFTER_HOURS, and the cost per unit distance if the ride is not after hours is given by the constant, PRICE_NORMAL_HOURS.

```
public class TaxiRide{
    private static final double PRICE_AFTER_HOURS = 3;
    private static final double PRICE_NORMAL_HOURS = 2;

    //source suburb of the taxi ride
    private String source;

    //destination suburb of the taxi ride
    private String destination;

    //indicates whether the taxi ride is after hours or not
    private boolean isAfterHours;

    //units of distance for the taxi ride
    private int distanceUnits;

    public TaxiRide(String source, String destination, int units,
                    boolean isAfterHrs) {
        this.source = source;
        this.destination = destination;
        distanceUnits = units;
        isAfterHours = isAfterHrs;
    }
}
```

```
public          getDestination() {

}
}
```

ID.....

```
public          setSource(                                ) {  
  
  
}
```

```
public void setIsAfterHours() {  
  
}  
}
```

```
public                                getCost() {  
  
  
  
  
  
  
  
  
  
}
```

```
public hasSameDestination( ) {  
  
  
  
  
  
  
  
  
}
```

```
public String toString() {
    String info = "TAXI RIDE ";
    info += "FROM: " + source + " TO: " + destination;

    if (isAfterHours)
        info += " - AFTER HOURS ";
    else
        info += " - NORMAL HOURS ";

    info += "$" + getCost();
    return info;
}
```

The following method is used to test the TaxiRide class.

```
private void testTaxiRide() {
    TaxiRide ride1, ride2;
    ride1= new TaxiRide("Newmarket", "City", 5, false);
    ride2 = new TaxiRide("Takapuna", "City", 12, true);

    System.out.println("1 " + ride1.toString());
    System.out.println("2 " + ride2.toString());

    System.out.println("3 " + ride1.getDestination());
    System.out.println("4 " + ride1.getCost());
    System.out.println("5 " + ride2.getCost());

    if (ride1.hasSameDestination(ride2))
        System.out.println("6 Same destination");
    else
        System.out.println("7 Different destinations");

    ride1.setSource("Epsom");
    ride1.setIsAfterHours(true);
    ride2.setIsAfterHours(false);

    System.out.println("8 " + ride1.toString());
    System.out.println("9 " + ride2.toString());
}
```

The testTaxiRide() method above uses a class, TaxiRide, to create two TaxiRide objects. Given a correct implementation of the TaxiRide class, the output from the testTaxiRide() method above should be exactly as shown below:

```
1 TAXI RIDE FROM: Newmarket TO: City - NORMAL HOURS $10.0
2 TAXI RIDE FROM: Takapuna TO: City - AFTER HOURS $36.0
3 City
4 10.0
5 36.0
6 Same destination
8 TAXI RIDE FROM: Epsom TO: City - AFTER HOURS $15.0
9 TAXI RIDE FROM: Takapuna TO: City - NORMAL HOURS $24.0
```

(12 marks)

ID.....

24. Consider the two-dimensional array `nums` visualised as below:

	0	1	2	3
0	1	0	-1	-2
1	2	1	0	-1
2	3	2	1	0
3	4	3	2	1

Complete the body of the nested loop below which would initialise the array as shown in the diagram above:

```
int[][] nums = new int[4][4];
for (int row = 0; row < nums.length; row++) {
    for (int col = 0; col < nums[row].length; col++) {
```

[illegible]

} }

(5 marks)

25. Consider the definition of the Shape class given below:

```
import java.awt.*;

public class Shape {

    private Rectangle pos;

    public Shape(int x, int y, int w, int h) {
        pos = new Rectangle(x, y, w, h);
    }

    public Rectangle getPos() {
        return pos;
    }
}
```

Each Shape object stores its position in the instance variable pos which is of type Rectangle.

For this question, you need to complete the countIntersections() method. This method is passed a Shape object and an array of Shape objects as parameters, and should return the number of Shape objects in the array which intersect with the first Shape object. Two Shape objects are considered to intersect if the Rectangle objects which define their positions intersect.

For example, the following code:

```
Shape[] shapes;
Shape extraShape;

shapes = new Shape[3];
shapes[0] = new Shape(56, 32, 5, 5);
shapes[1] = new Shape(10, 20, 30, 40);
shapes[2] = new Shape(25, 25, 10, 15);

extraShape = new Shape(11, 21, 10, 10);

int n = countIntersections(extraShape, shapes);

System.out.println( n );
```

should produce the output:

1

because the Shape object referred to by the extraShape variable intersects with the second element of the shapes array, but not the other two elements.

ID.....

Complete the `countIntersections()` method below. You may assume that the parameters `s` and `shapes` are not `null`. You may also assume that the elements in the array `shapes` are not `null`.

```
public int countIntersections(Shape s, Shape[] shapes) {
```

```
}
```

(10 marks)

CONTINUED

ID.....

26. For this question, you need to define a class called `MovingSquare` such that the program described below works correctly. The source code for the program is given below:

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

public class MyJPanel extends JPanel implements ActionListener {

    private Timer t;
    private Rectangle boundary;
    private MovingSquare ms;

    public MyJPanel() {
        setBackground(Color.white);

        t = new Timer(50, this);
        t.start();

        boundary = randomRectangle();
        ms = new MovingSquare(boundary.x, boundary.y);
    }

    public Rectangle randomRectangle() {
        int x = (int) (Math.random() * 200);
        int y = (int) (Math.random() * 200);
        int w = (int) (Math.random() * 400) + 100;
        int h = (int) (Math.random() * 400) + 100;

        return new Rectangle(x, y, w, h);
    }

    public void actionPerformed(ActionEvent e) {
        ms.move(boundary);
        repaint();
    }

    public void paintComponent(Graphics g) {
        super.paintComponent(g);

        ms.draw(g);
        g.drawRect(boundary.x, boundary.y, boundary.width,
                    boundary.height);
    }
}
```

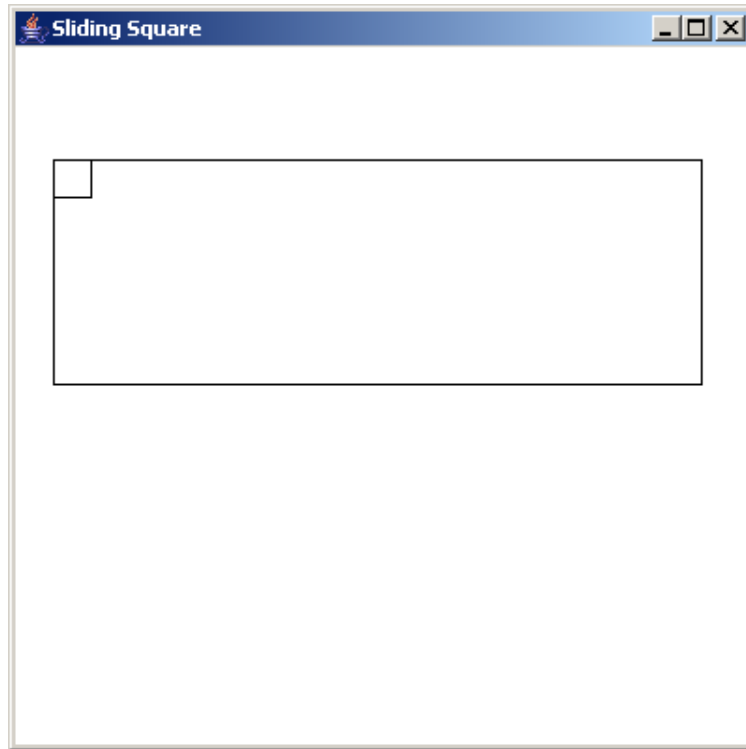
When the program first starts, two rectangles should be seen drawn on the window. The first rectangle has a random position and a random width and height (although the width and the height are at least 100). The second rectangle is initially positioned such that its top left hand corner is at exactly the same position as the first rectangle. The width and height of this second rectangle are both 20.

CONTINUED

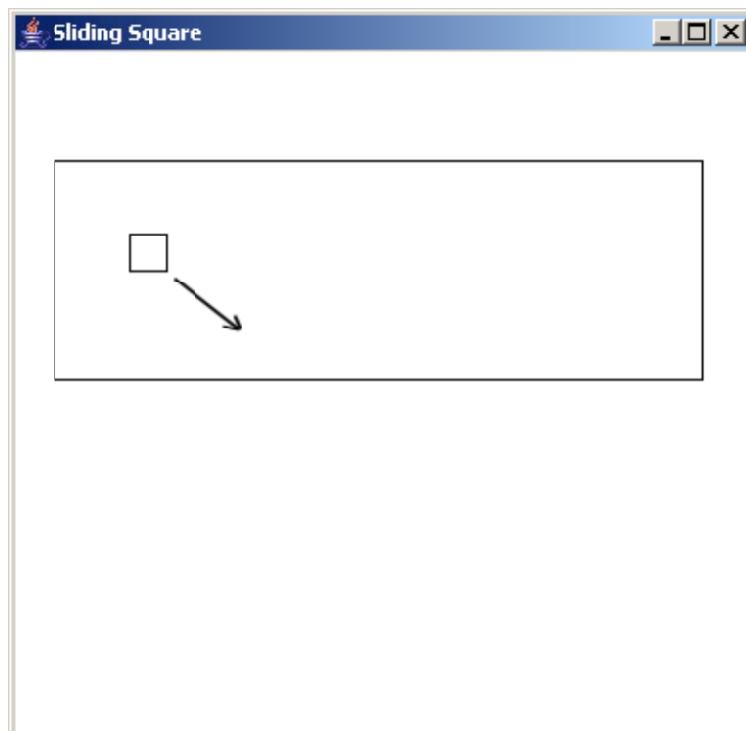
ID.....

As soon as the program starts, the smaller rectangle should start moving diagonally down and to the right. It should move as far as it can down, and as far as it can right without crossing the boundary of the larger rectangle.

The screenshot below shows one example of what the program might look like when it first starts.



The smaller square will start moving diagonally down and to the right, as shown below.



CONTINUED

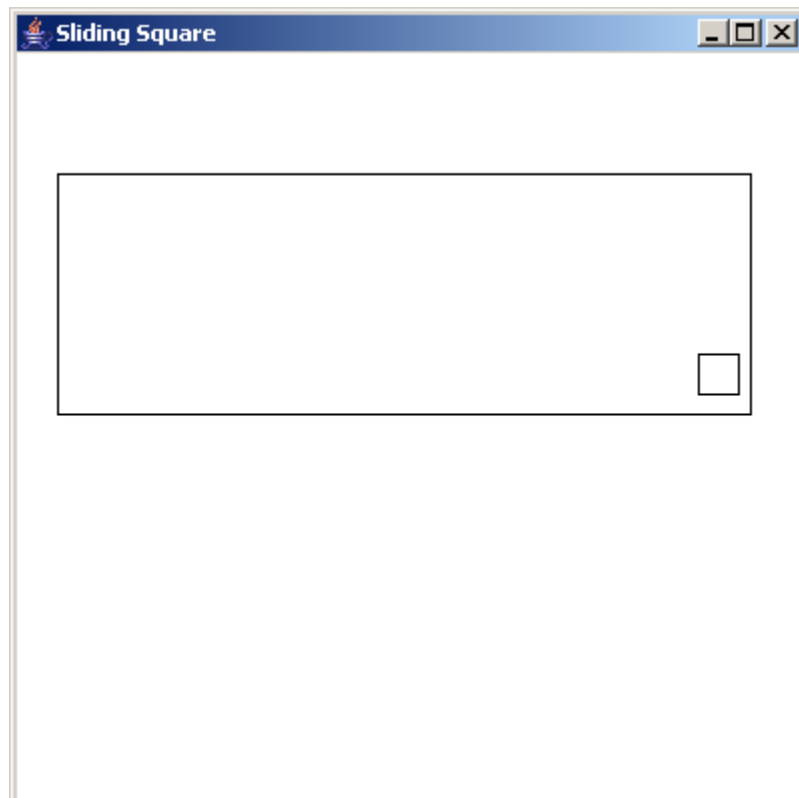
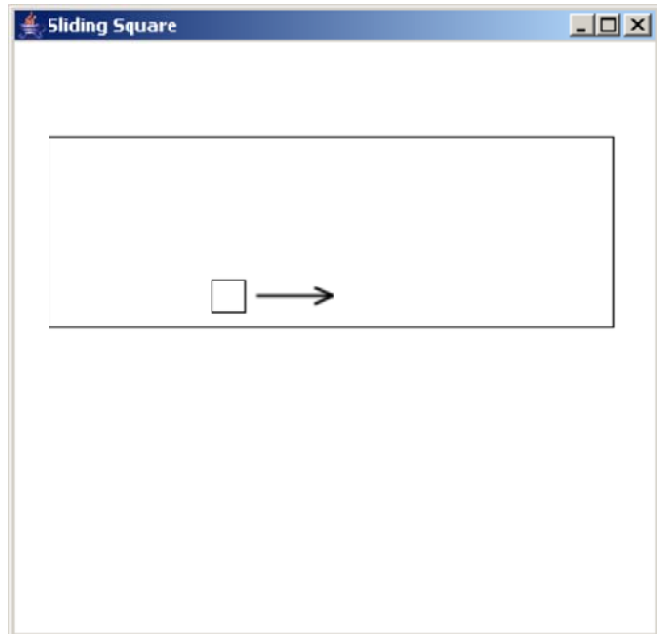
The smaller square moves 10 pixels in the x direction and 10 pixels in the y direction.

In the example above, the square will soon reach a point where it can no longer move down any further without crossing the bottom edge of the larger rectangle. At this point, the smaller square will start to move directly towards the right. The screenshot below illustrates this.

Notice that the bottom edge of the smaller square may not line up exactly with the bottom edge of the larger square (the larger square has a random width and height). A gap is clearly visible in the screenshot above – this gap is less than 10 pixels however because the position of the smaller square changes by 10 pixels each time it moves.

If the larger rectangle had been taller than it was wide, then the smaller square would have moved downwards when it reached the point where it could no longer move to the right.

When the smaller square reaches a position where it can no longer move down and no longer move to the right without crossing the boundary of the larger rectangle, then it will no longer move. The final position of the smaller square in the example above is shown in the screenshot below:

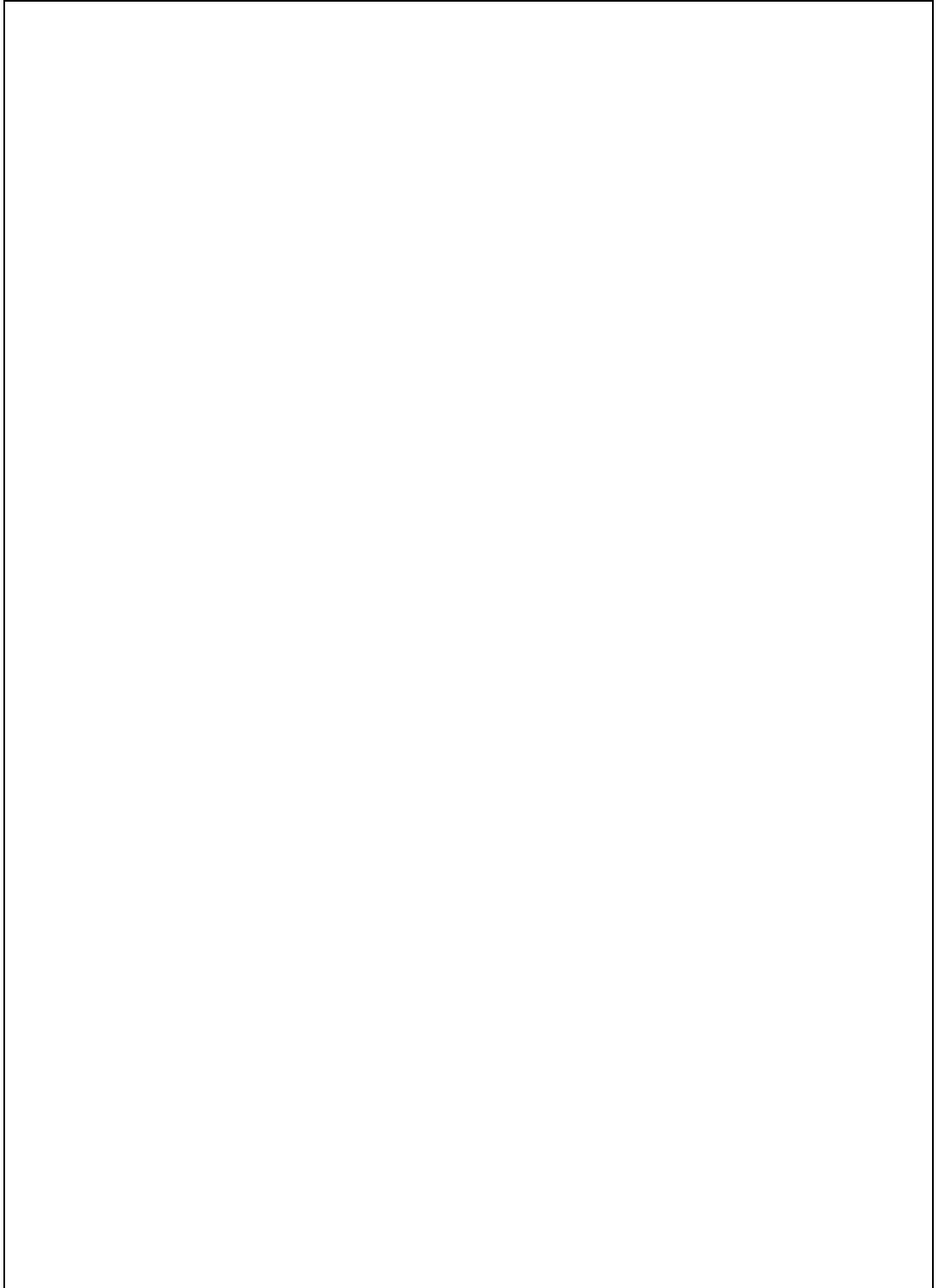


ID.....

Complete the definition of the MovingSquare class below:

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

```
public class MovingSquare {
```



```
}
```

(10 marks)

CONTINUED

ID.....

27. Answer the following questions about ethics:

(a) Name one professional computing society

(1 mark)

(b) A fast food chain installs a safe in each of its branches to store cash securely overnight. You are asked to develop a system to electronically lock and unlock the safes, using mobile phone technology (there is a mobile phone in each safe, and calling the phone with a special code will open or close the lock).

i) Identify TWO stakeholders and their interest in the project

(1 mark)

ii). Briefly describe any significant risks that will need to be addressed by the developers of this system

(3 marks)

ID.....

OVERFLOW PAGE

(If you have used this page, please indicate clearly under the relevant question that you have overflowed to this page)

ID.....

OVERFLOW PAGE

(If you have used this page, please indicate clearly under the relevant question that you have overflowed to this page)

ID.....

ROUGH WORKING
(Will not be marked)

ID.....

ROUGH WORKING
(Will not be marked)
