THE UNIVERSITY OF AUCKLAND

SECOND SEMESTER, 2007

Campus: City

COMPUTER SCIENCE

TEST

Principles of Programming

(Time allowed: 75 MINUTES)

NOTE: Attempt **ALL** questions
Write your answers in the space provided
There is space at the back for answers that overflow the allotted space
No calculators are permitted

Surname:	
Forenames:	
Student ID number:	
Login name:	

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

CompSci 101 Test Results

Question	Marks	Out of
Question 1		10
Question 2		5
Question 3		5
Question 4		10
Question 5		20
Question 6		10
TOTAL		60

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

Question 1 (10 marks)

What is displayed by each of the following pieces of Java program?

a)
```
System.out.println("\"quote");
```
(1 mark)

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

c)
```
int value = 13;
System.out.println(value < 6);
```
(1 mark)

d)
```
int i = 5;
int[] numbers = {4, 2, -7, 5, 1, 6, 3};
System.out.println (numbers[i]);
```
(1 mark)

e)
```
String name = "banana split";
System.out.println(name.indexOf('n'));
```
(1 mark)

f)
```
String name = "facile";
System.out.println(name.substring(1, 3));
```
(1 mark)

g)
```
String name = "peasy";
System.out.println(name.substring(1));
```
(1 mark)

h)
```
 int x = 5, y = 4;
System.out.println(Math.max(x, y));
```
(1 mark)

CONTINUED

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

i)
```
int [] numbers = {2,3,4,5,6,7,8};
System.out.println(numbers[0]*numbers[3]);
```

(1 mark)

j)
```
String [] strings = {"cat","dog","snake","ocelot"};
System.out.println(strings[2].length());
```

(1 mark)

Question 2 (5 marks)

Add the return type to the following Java methods:

a)
```
private _____ addNums(int num1, int num2) {
     return num1 + num2;
}
```

(1 mark)

b)
```
private _____ printNumber(int num) {
     System.out.println(num);
}
```

(1 mark)

c)
```
private _____ getString(char c) {
     return "" + c;
}
```

(1 mark)

d)
```
private _____ increase(int num) {
     return num + 1.0;
}
```

(1 mark)

e)
```
private _____ isOdd(int num) {
     return ((num % 2) == 1);
}
```

(1 mark)

CONTINUED

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

Question 3 (5 marks)

a) Write a Java `boolean` expression which tests if the value of the `int` variable, `number`, is less than 50.

(1 mark)

b) Write a Java `boolean` expression which tests if the value of the `int` variable, `number`, is even.

(1 mark)

c) Write a Java `boolean` expression which tests if the value of the `boolean` variable, `isThin`, is false

(1 mark)

d) Write a Java `boolean` expression which tests if the value of the `String` variable, `name`, has the same characters in the same order as **`"George".`**

(1 mark)

e) Write a Java `boolean` expression which tests if the value of the `int` variable, `number`, is not zero and divides evenly into 500

(1 mark)

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

Question 4 (10 marks)

Complete each of the methods below as specified in the comment preceding each method.

a)

```
// method to find the smallest of three numbers
private int smallest(int x1, int x2, int x3){




}
```

(2 marks)

b)

```
// method to decide whether two numbers are in ascending
// order
private boolean areInOrder(int n1, int n2){




}
```

(2 marks)

c)

```
// method to return the last half of a String (including
// the middle character if the length is odd)
private String lastHalf(String s){




}
```

(2 marks)

d)

```
// method to tell whether one String, s1, is the first
// part of a longer String, s2 (e.g. isStart("ab","abcd")
// is true, but isStart("ab","acbd") is false
private boolean isStart(String s1, String s2){









}
```

(2 marks)

CONTINUED

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

e)

```
// method to display the first parameter, s, the number of
// times given by the second parameter, n. The output
// should all be on one line, with no extra spacing.
private void reiterate(String s, int n){




}
```

(2 marks)

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

Question 5 (20 marks)

a. Complete the method `incomeInRange()` which accepts a positive `double` parameter `income` and two `double` bounds, `lower` and `upper`, where `lower` <= `upper`. The method returns the amount of the `income` that is between the two bounds. You may use if-statements to answer this question. Do not use `Math` class methods.

```
// method to find the income within a range
private double incomeInRange
                (double income, double lower, double upper) {









}
```

(5 marks)

b. Complete the method called `getBoundedInt()` which prompts the user to type in an `int` value less than the value of the `int` parameter `bound`. A user who types a number that is greater than, or equal to, `bound`, is continually re-prompted to provide an `int` value less than `bound`. When the user has typed a correct value it is returned as the value of the method. You should make use of the "helper" method `getUserInt()` that reads the user input and returns it as an `int`

```
// method to get a bounded int from the user

private int getBoundedInt(int bound) {
     System.out.print("Type in a number less than " + bound);
     int number = getUserInt();









}
```

(5 marks)

CONTINUED

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

c. Complete the method **`rotateLeft()`** which accepts a **`String`** parameter `word,` and returns the **`String`** rotated one character to the left, e.g. `"luck"` gives the result `"uckl"`

```
// method to find a word rotated to the left, e.g. "luck"
// gives result "uckl"

private String rotateLeft (String word){

}
```

(5 marks)

d. Complete the method **`isAllUpperCase()`** which accepts a **`String`** parameter, `word,` and returns `true` if every character in `word` is upper case, and `false` otherwise. You may use the method `Character.isUpperCase(char c)` that determines whether `char c` is an upper case character.

```
// method to see if a word is all upper case

private boolean isAllUpperCase(String word) {
     boolean allSoFar = true; // all letters so
                              //far tested are upper case

}
```

(5 marks)

CONTINUED

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

Question 6 (10 marks)

a) What is the output when the following `start()` method is executed?

```
// mystery program
public void start() {
     String s1 = "trial";
     String s2 = "";
     for (int i = 0; i<s1.length(); i++) {
          char c = s1.charAt(i);
          s2 = c + s2;
          System.out.println(s2);
     }
}
```

Show the output here:

(5 marks)

b) What is the output when the following `start()` method is executed? Note that the "helper" method `printArray()` just displays all of the elements of its parameter array on one line.

```
// mystery program
public void start() {
     int[] numbers = {1,1,1,1,1};
     printArray(numbers);
     numbers = bump(numbers);
     printArray(numbers);
     numbers = bump(numbers);
     printArray(numbers);
     }


// helper method to print an array of ints on one line
private void printArray(int[] a) {
     for (int j=0; j<a.length; j++) {
          System.out.print(" " + a[j]);
     }
     System.out.println();
}
```

CONTINUED

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

```
// method to do something with an array of int numbers
private int[] bump(int[] nums) {
     int[] value = new int[nums.length];
     for (int j=0; j<nums.length; j++) {
          value[j] = nums[j] + j;
     }
     return value;
}
```

Show the output here:

(5 marks)

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

ROUGH WORKING (WILL NOT BE MARKED)

(You may detach this page from the answer booklet and use it for rough working)

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

ROUGH WORKING (WILL NOT BE MARKED)

(You may detach this page from the answer booklet and use it for rough working)