University of Cape Town
Department of Computer Science

Computer Science CSC115F

Final Exam

- Answer all questions.
- All questions that refer to elements of programming make reference to the Java programming language as studied in class.
- Good luck !

Marks: 25

Time: ?? minutes

- Approximate marks per question are shown in brackets
- The use of calculators is permitted

NAME: Surname Initials

STUDENT NO: **COURSE CODE:** CSC

This paper consists of 2 questions and 4 pages (including this cover page).

Mark Allocation							
Quest	Marks	Internal	External	Quest	Marks	Internal	External
1	[15]			2	[10]		
	Total				**Total**		
					Grand Total		
					Final Mark		
Internal Examiner:				**External Examiner:**			

Section 1. Java Basics

Question 1. [15 marks]

a) List the 3 syntax errors in the following code fragment (line numbers are added so you can refer to specific lines):

```
line1: public float func ( integer a, float b )
line2: {
line3:    float result = (1.0f+((a*2)+(b*3))
line4:    return result;
line5: }
```

i) line1: integer is not a valid data type

ii) line3: missing)

iii) line1: missing ;

[3]

b) List one difference between constants and literals.

Constants are named values so can be reused by name while literals have to be inserted verbatim wherever needed.

[2]

c) Why does the expression `"day"<31` result in an error?

Incompatible data types.

[1]

d) Write the method `calcRoot` to calculate a root of a quadratic equation (i.e., a value of x for which $ax^2 + bx + c = 0$). Your method must assume that a, b and c are `double` instance variables. `calcRoot` must take no parameters and return a floating point result. In addition, if $b^2 - 4ac$ is negative, the returned value must be 0 and an error message must be written to the console (screen).

Remember that the roots are given by: $x = \frac{-b \pm \sqrt{b^2-4ac}}{2a}$ and that the `Math.pow (x, y)` method calculates x^y.

```
public double calcRoot ()
{
    if ((b * b - 4 * a * c) < 0)
    {
        System.out.println ("Error");
        return 0;
    }
    return (- b + Math.pow (b * b - 4 * a * c, 0.5)) / (2 * a);
}
```

Marking guide: ignore syntax errors, -1/2 mark for each real error, round off to next whole mark

or

1 – some basic structure eg. correct header
2 – expression written properly but lots of errors
3 – a few errors but too many to warrant a full score
4 – only minor syntax errors, with possibly one real error

[4]

e) Write a method `adjustedAverage` to calculate the average of the two highest marks in a set of three marks. For example, the adjusted average of 12, 24 and 36 is 30 because the smallest value is ignored. Your method must take 3 integer parameters and return an integer.

```
public int adjustedAverage (int f, int g, int h)
{
    if ((f <= g) && (g <= h))
        return (g + h) / 2;
    if ((g <= f) && (g <= h))
        return (f + h) / 2;
    return (f + g) / 2;
}
```

Marking guide: ignore syntax errors, assume 3 unique numbers, concentrate on correctness of algorithm, -1/2 mark for each real error, round off to next whole mark

or

1 – some basic structure eg. correct header
2 – somewhat of an idea in terms of algorithm
3 – algorithm works but not for all cases, or almost works
4 – algorithms works for most cases
5 – only minor syntax errors, with possibly one real error or oversight

[5]

Section 2. Number Systems

Question 2. [10 marks]

a) In boolean addition, explain what an overflow is and illustrate with an example.

An overflow occurs when adding the leftmost bits results in a carry, thereby signifying that the number of bits is insufficient to store the number. [2]

For example: 1001+1001 = (1)0010 overflow [1]

Marking notes: answer will still be correct if interpreted as 1-bit scenario.

[3]

b) Convert 23.125_{10} into its hexadecimal representation. Show full calculations and clearly indicate your final answer.

2	23		
2	11	r	1
2	5	r	1
2	2	r	1
2	1	r	0
	0	r	1

Thus, $23_{10} = 10111_2$ [1]

0.125 * 2 = 0.250 Intpart = 0
0.250 * 2 = 0.500 Intpart = 0
0.500 * 2 = 1.000 Intpart = 1
Thus, $0.125_{10} = 0.001_2$ [1]

Thus, $23.125_{10} = 10111.0010_2 = 17.2_{16}$ [1]

[3]

c) Write an algorithm to add together 2 whole binary numbers in 1's complement, where the numbers have differing numbers of bits. Assume the numbers are already in 1's complement and leave the result in 1's complement form.

1. If the leading digit of the smaller number is 0, pad with 0's to the left (else pad with 1's) until numbers are the same length
2. Add pairs of corresponding digits from the lsb to the msb, adding the carry from each operation to the next addition
3. If there is a carry from the last addition, drop the carry and add 1 to the sum

Marking notes: first step may be omitted without losing marks, ignore sign-dependence of padding, 1 mark lost for missing last step, marks lost for errors in algorithm

[4]