Lab 1 Data Representation

October 1, 2012

The goal of this lab is to understand how computers internally represent data. In this lab the student will develop a set of programs using the Java programming language to work with integer and IEEE 754 single precision floating point numbers. We strongly recommend the student to review the following theoretical aspects:

- Java operators to manage bits. E.g. the following link presents detailed information about bit operators in Java:
  1. `http://vipan.com/htdocs/bitwisehelp.html`
  2. `http://es.wikibooks.org/wiki/Programaci%C3%B3n_en_Java/Operadores_de_bits` in Spanish.
- Representation using IEEE 754 single precision.

This lab consists of five exercises.

1 Exercise 1

Write a program in Java that calculates the following sequence:

$$S_n = \frac{1}{2^0} + \frac{1}{2^1} + \frac{1}{2^3} + ... + \frac{1}{2^{n-1}} + \frac{1}{2^n}$$

Use single precision numbers (float variables) and double precision numbers (double variables). The goal is to calculate the mentioned series and indicate the maximum number of iterations before we obtain the same value. In other words, $S_n = S_{n+1}$. The student must submit:

- **Series.java**. File containing the source code used to calculate the requested series. This file must contain the Javadoc describing how it works, the authors, input and output values.
- **Series.txt**. A txt format file indicating the answer to the following questions.

integerNumber	Representation
12553	0000 0000 0000 0000 0011 0001 0000 1001
-12553	1111 1111 1111 1111 1100 1110 1111 0111

1. What is the maximum number of iterations using single precision? And double precision?
2. What is the difference between a floating number in Java using 32 and 64 bits architectures? And in the case of double? Justify your answer.

2 Exercise 2

Write a Java program called IntegerNumber.java that given an integer number returns its 2's complement representation. In order to do this, you must complete the method intToComp2 contained in the provided source code.

- `String intToComp2(int integerNumber)`

  This method receives an integer number as input and returns its 2's complement representation in a character string. E.g.:

The students must submit:

- **IntegerNumber.java**. The source code of the method intToComp2. The class can contain as many methods as the student considers necessary.

  Notes: Consider $n = 32$ when calculating the 2's complement. Is not permitted to employ methods from Integer and Byte.

3 Exercise 3

Write a Java program named RealNumber.java employing the baseline source code. This program must receive a float number returning its IEEE 754 standard representation in a string of characters. The sign, exponent and mantissa must be separated by a blank space.

- `String floatToIEEE754(float realNumber)`

  This method receives a floating point number as input and returns its single precision IEEE 754 representation in a character string. E.g.:

Answer in the final document the following question:

realNumber	Representation
+25.6	0 10000011 10011001100110011001101
-53.25	1 10000100 10101010000000000000000

1. Calculate the number of representable numbers between 8 and 9, and between 15 and 16. In what interval can we represent a larger amount of numbers? Justify your answer.

The students must submit:

- **RealNumber.java**. The source code of the method floatToIEEE754. The class can contain as many methods as the student considers necessary. Additionally, the source code must contain the Javadoc documentation describing input, outputs, method working and authors.

  Notes: Is not permitted to employ methods from Integer, Byte and Float.

4 Exercise 4

Write a Java program named RealMultiplier.java employing the baseline source code. This program must receive two float numbers get the IEEE 754 representation of both numbers and return a characters string with the representation in IEEE 754 of the resulting multiplication. The sign, exponent and mantes must be separated by a single blank space.

- `String floatMultiplier(float num1, float num2)`

  This method receives two floating point numbers, transforms them into IEEE 754 and returns the resulting multiplication.

The students must submit:

- **RealMultiplier.java**. The source code of the method floatMultiplier. The class can contain as many methods as the student considers necessary. Additionally, the source code must contain the Javadoc documentation describing input, outputs, method working and authors.

  Notes: The selected rounding method will be truncation. If the methods developed in previous exercises are going used, these must be copied into the submitted file. It is not permitted to employ methods from Integer, Byte and Float. It is not permitted to multiply the numbers and then represent the resulting number. The numbers MUST be multiplied using their IEEE 754 representations. In case of overflow an empty string must be returned.

5 Exercise 5

Write a Java program named AdditionComplement.java employing the baseline source code. This program must receive two integer numbers, get their 2's complement representation and return a string of characters with the addition of both numbers in 2's complement.

- `String integerAdder(int num1, int num2)`

  This method receives two integer numbers, get theris 2's complement representation and return the resulting addition in 2's complement.

The students must submit:

- **AdditionComplement.java**. The source code of the method integerAdder. The class can contain as many methods as the student considers necessary. Additionally, the source code must contain the Javadoc documentation describing input, outputs, method working and authors.

  Notes: The selected rounding method will be truncation. If the methods developed in previous exercises are used, these must be copied into the submitted file. It is not permitted to employ methods from Integer and Byte. It is not permitted to multiply the integer numbers and then represent the resulting number. The numbers MUST be multiplied using their 2's complement. In case of overflow the method must return an empty string.

Lab evaluation

The evaluation of this lab is divided into three parts:

- During lab session (2 points). This part will be done during the lab session. The material will be submitted before the lab finishes. This submission includes the Exercise 1.
- Mandatory part (4 points). This part consists of exercises 2 and 3 and the final document. This part will be evaluated as follows:
  - Exercise 2 (1.5 points)
  - Exercise 3 (1.5 points)
  - Final document (1 point)
- Optional part 1 (3 points). Exercise 4 and its corresponding section in the final document.

  - Exercise 4 (2.75 points)
  - Final document modifications to include exercise 4 (0.25 points)

- Optional part 2 (1 point) Exercise 5 and its corresponding section in the final document.
  - Exercise 5 (2.75 points)
  - Final document modifications to include exercise 5 (0.25 points)

Submission

The submission will be as follows:

- **Exercise 1** will be submitted through aula global during the lab session. JUST ONE ZIPPED file with name **ec_p1_e1_A...A_B...B.zip** with A...A and B...B the student identifiers of the group members. The zip must contain:
  - Series.java
  - Series.txt

- The remainder sections will be submitted with the final document in a zip file with name **ec_p1_e2_A...A_B...B.zip** before October 22^{nd} at 23:55 through Aula Global.

  ONLY one zipped file will be submitted containing the following files:
  - IntegerNumber.java
  - RealNumber.java
  - RealMultiplier.java
  - AdditionComplement.java
  - Document.pdf

    The final document must contain at least the following elements:
    1. Front page indicating authors and their student numbers.
    2. Index of contents.
    3. Description of the implemented methods.
    4. Done tests and their results.
    5. Conclusions.
    6. Don't forget to justify the margins and numerate the pages.

The final document is fundamental to get a good mark in your lab work. It is also necessary to submit the correct code, with the requested method names. The length of the final document MUST NOT exceed 10 pages (index and front page included). If the final document does not fulfill the mentioned requirements, the lab will be failed.

- The submission will be done ONLY through AULA GLOBAL.
- The version to be reviews is the last one submitted.

Rules

1. The code MUST compile and MUST work as expected.
2. Non-commented code is considered void.
3. The submission MUST be done through AULA GLOBAL. No alternative methods can be used.
4. The submitted code and final document MUST be original work. If cheating is detected, all the students involved in the copy will fail the lab.