CS 170 Section 002 Name:________________________________________
HW 1 – Spring 2014
Due: Monday, Feb. 3 by the start of class

Instructions: This is the first homework for CS170 (Section 002). Unlike labs, you are expected to do homework on your own. Future homework will involve programming, but this first one, is mostly written, and you should turn it in on paper.

Honor Code: Like all work for this class, the Emory Honor Code applies. You should do your own work on all problems, unless you are explicitly instructed otherwise. If you get stuck or have questions, ask your instructor or a TA for help.
Initial here to indicate that you followed the Honor Code and this work is your own. ______________

1. Define the following using your own words.

a) (3 pts) Describe the difference between a high-level language and a low-level language.

b) (3 pts) What does a compiler do?

c) (3 pts) What is an operating system?

d) (3 pts) List 3 examples of operating systems, 3 examples of high-level programming languages and 3 examples of output hardware.

e) (3 pts) Why is it convenient to have a hard drive if we already have RAM (or main) memory?
2. Complete the following:
(10 pts) If a program in RAM is using 1.1 Gb (base 2 definition), then it is using (write the exact number, even if, for example, fractional bits doesn't really make sense):

______________________________________ MB,
______________________________________ KB,
______________________________________ Bytes, and
______________________________________ Bits.

(5 pts) If a song has a size of 4.2 MB in Hard Drive (base 10 definition), then its size is
______________________________________ GB and
______________________________________ Bytes.

3. ASCII Conversions
(15 pts) The following sentence cannot be neither true nor false, because in any case it leads to a paradox. The sentence, however, is encoded in (decimal) ASCII. Using an ASCII table either from the book or the internet, decode the sentence:

084 104 105 115 032 115 116 097 116 101 109 101 110 116 032 105 115 032 102 097 108 115 101 033

The famous mathematician Kurt Gödel used a modified version of this paradox to propose the Incompleteness Theorem. Encode “Kurt Goedel” (without the quotations) in decimal ascii. Notice we cannot encode “ö” with ascii.


a) (3 pts) 145 in base 10 = ____________________________ in base 2
b) (3 pts) 238 in base 10 = ____________________________ in base 3
c) (3 pts) 101101 in base 2 = __________________________ in base 10
d) (3 pts) 10110 in base 3 = __________________________ in base 10
e) (3 pts) 1521 in base 10 = ____________________________ in base 16
25. (15 pts) Convert the following number to binary
314 in base 10 = ________________________ in base 2. If we tried storing this number in 1 byte, we would have an overflow, and the computer would only store the 8 right-most characters. Take the 8 right-most characters and convert it back to decimal.

In what way does this number relate to the original? By how much is it greater or lesser than the original? Why? (Hint: Think about how we convert numbers from binary to decimal, and what happens if we remove the left-most digits)

6. Algorithm Tracing
Consider the following algorithm:
\[
\begin{align*}
n &= 14; \\
a &= 7; \\
b &= 0; \\
as long as \ a < n do \\
\{ \\
&b = b + a; \\
&a = a + 1; \\
&\}
\end{align*}
\]
Output is: b
a) (15 pts) Trace out each step as begun below for you:
(Use as many rows as you need)
Initially: \( n = 14, \ a = 7, \ b = 0, \)
After 1 Step: \( n = 14 \quad a = 8 \quad b = 7 \)
After 2 Steps:
3 b) (10 pts) What does this algorithm accomplish? Explain the relationship between a, and n WITHOUT simply explaining the algorithm in the first part. You may want to try the algorithm for different values of n to see if the algorithm accomplishes what you think it does.

Extra credit (+5)  
Suppose you have 12 cubes, all of which look exactly the same. However, you know that one of them weighs differently (you don't know if it is heavier or lighter). You have at your disposition a weighing scale with which you can compare the weights of the cubes, but you can only use it three times. Describe the procedure you would follow to find the cube that weighs differently. Remember that an algorithm should have unambiguous instructions. You can try your algorithm with a few examples and see if at each step you know what to do next, and the algorithm manages to find out the cube that weighs differently.