CS 345: Homework 3 (Due: Before midnight of Nov 15, 2016)

Rules. Individual homeworks; see Handout 1 (aka Syllabus).

Problem 1. (15 points) **v-byte encoding**
You are given an inverted list for a term t that looks like

```
(100,1)(100,5)(100,11)(100,15)(100,17)(100,20)(120,10)(120,18)(120,30)(120,40)(130,11)(130,25)(140,80)(140,81)
```

v-byte encode this inverted list by providing the following information. (Note that v-byte encoding is also discussed on page 150 of the textbook. However, the example of the textbook incorrectly applies gap encoding to a count field shown here in bold-face, contradicting the algorithm stated there or in the notes: *1,2,1,6,1,***2***, 6, 11, etc.*)
(b) Give the flattened (no parenthesis) form of the to be v-byte encoded list just before the v-byte encoding is applied when the list is flattened but still in decimal notation, and
(c) after the v-byte encoding has been applied and the list given in hexadecimal notation (use two hexadecimal digits, comma separated, do not use 0x prefixes).

Problem 2. (15 points) **Elias and Golomb**
Provide Elias-γ, Elias-δ and Golomb-7 codes for $k = 71$. Which one is shorter?

Problem 3. (15 points)
Under a Zipf's Law scenario, what fraction of words in a corpus have frequency more than 3?

Problem 4. (10 points) **Heaps' Law revisited**

We revisit Heaps' Law (read solutions of a HW2 problem to avoid making the same mistakes). In 2016, Google has approximately 25G documents, with average text content 110KiB (roughly 120,000B). How many distinct words does Heaps' Law imply? Justify your claims. Round to the closest million or multiple of ten-million as needed (i.e. do not dare write 123,456,789 or 12,345,567 but 120,000,000 or 13,000,000).

How does this compare to the 14,000,000 figure of Google in 1998? Can you explain the discrepancy? ("The English language is 10 times richer in words is not an explanation of interest".)

Problem 5. (15 points) **Huffman and More...**
(a) The text below of 25 characters A, C, G, T, W is first encoded in a byte-aligned ASCII code, and then using Huffman coding. (Ignoring spaces that are provided for readability only), what are the prefix codes for each one of the five characters under Huffman coding, and what is the total number of bits used to encode just the text (consisting of A, C, G, T, W and ignoring the spaces or other auxiliary information)? How does this compare to the ASCII encoding (express bit and byte SAVINGS as a percentage).

```
GGCTA AAGGG WCCTA AGGCC WGCTC
```

(b) What is the entropy of the text? Compare it to the average number of bits per character in the Huffman coding.

Date Edited: 10/18/2016