CS 325 - Spring 2020

Homework #4

Problem 1. (6pts)

a) Assume the following symbols *a, b, c, d, e* occur with frequencies 1/2, 1/4, 1/8, 1/16, 1/16 respectively. What is the Huffman encoding of the alphabet? (3 pts)

b) If the encoding is applied to a file consisting of 1 million characters with the same given frequencies, what is the length of the encoded file in bits? (3 pts)

Problem 2. (5pts)

Complete problem 16.2-2 on page 427 in the book:

Give a dynamic-programming solution to the 0-1 knapsack problem that runs in $O(n\ W)$ time, where n is the number of items and W is the maximum weight of items that a thief can put in their knapsack.

Problem 3. (8 pts)

Consider the problem of making change for n cents using the fewest number of coins. Assume that each coin's value is an integer.

Problem 3.a. (4 points)

a) Suppose that the available coins are in the denominations that are powers of *c, i.e.*, the denominations are $c^0; c^1; \ldots; c^k$ for some integers c > 1 and k _ 1. Show that the greedy algorithm of *picking the largest denomination* first always yields an optimal solution. You are expected to reason about why this approach gives an optimal solution. (*Hint*: Show that for each denomination c^i, the optimal solution must have less than *c* coins.)

Problem 3.b. (4 points)

b) Design an $O(nk)$ time algorithm that makes change for any set of *k* different coin denominations, assuming that <u>one</u> of the coins is 3 cents in value.

Problem 4. (6 pts)

Implementation:
Implement the make change algorithm you designed in the previous problem. Your program should read a text file “data.txt" where each line in “data.txt" contains three values *c, k* and *n*. Please make sure you take your input in the specified order *c, k* and *n*. For example, a line in “data.txt" may look like the following:

4 3 73

where c = 4; k = 3; n = 73. That is, the set of denominations is $\{4^0; 4^1; 4^2; 4^3\} = \{1; 4; 16; 64\}$, and we would like to make change for n = 73. The file "data.txt" may include multiple lines like above.

The output will be written to a file called "change.txt", where the output corresponding to each input line contains a few lines. Each line has two numbers, where the first number denotes a denomination and the second number represents the cardinality of that denomination in the solution. For example, for the above input line '4 3 73', the optimal solution is the multiset {64; 4; 4; 1}, and the output in the file "change.txt" is as follows:

Data input: c = 4, k = 3, n = 73
Denomination: 64 Quantity: 1
Denomination: 16 Quantity: none
Denomination: 4 Quantity: 2
Denomination: 1 Quantity: 1

which means the solution contains one coin of denomination 64, none of 16, two coins of 4, and one coin of 1. You can use a delimiter line to separate the outputs generated for different input lines.

Problem 5 – Extra Credit (4 pts)

a) Using Huffman encoding of n symbols with the frequencies $f_1, f_2, f_3 \ldots f_n$, what is the longest a codeword could possibly be? (2pts)
b) Give at least one example set of frequencies that would produce the case above. (2pts)

Submit a copy of all your code files and a README file that explains how to compile and run your code in a ZIP file to TEACH. We will only test execution with an input file named data