Lab 7 (13 Oct 2020)

Problem 1 : Implement Huffman's algorithm for encoding symbols. Take as input a list of n symbols (characters) and their corresponding frequencies. After constructing the optimal tree, your algorithm should print the encoding of each of the symbols. Also print the size of the encoded file.

Problem 2 : Implement the dynamic programming algorithm for computing the longest increasing subsequence. Read as input a sequence of numbers for e.g. 5 2 8 6 3 6 9 7 & print a longest increasing subsequence: for this example 2 3 6 9 (or 2 3 6 7).

Problem 3: Given two strings x[1..n] and y[1..m] we want to calculate the edit distance (the cost of the optimal alignment) of x and y. We are allowed three operations: insert a character, delete a character & replace a character, each operation having cost 1.

For e.g. for input strings x = 'TYPES' and y = 'STYLE' the edit distance is 3, since an optimal alignment is

```
_  T  Y  P  E  S
|  |  |  |  |  |
S  T  Y  L  E  _
```

whose cost (edit distance) is 3.

a) Write a top-down dynamic programming algorithm to solve this problem.
b) Write an iterative (bottom-up) version of the above algorithm
c) Print the optimal alignment of the two strings along with the cost of each matching. For the above input your program should print:

```
_  S  1
T  T  0
Y  Y  0
P  L  1
E  E  0
S  _  1
```

Problem 4: Write a decoder for the Huffamn encoder of Problem 1. You can test your program on text files: Encode an input text file as a binary file, and then decode it, ensuring that you obtain the same file. What is the percentage of compression achieved by the encoding?