CSC-421 Applied Algorithms and Structures
Winter 2019

Instructor: Iyad Kanj
Office: CDM 832
Phone: (312) 362-5558
Email: `ikanj@cs.depaul.edu`
Office Hours: Monday & Wednesday 4:00-5:30
Course Website: https://d2l.depaul.edu/

Assignment #3
(Due March 6)

1. Illustrate the execution of the **Coin Change** algorithm on $n = 10$ in the system of denominations $d(1) = 1$, $d(2) = 5$, and $d(3) = 8$.

2. Pascal's triangle looks as follows:

```
1
1 1
1 2 1
1 3 3 1
1 4 6 4 1
...
```

   The first entry in a row is 1 and the last entry is 1 (except for the first row which contains only 1), and every other entry in Pascal's triangle is equal to the sum of the following two entries: the entry that is in the previous row and the same column, and the entry that is in the previous row and previous column.

   (a) Give a recursive definition for the entry $C[i, j]$ at row i and column j of Pascal's triangle. Make sure that you distinguish the base case(s).

   (b) Give a recursive algorithm to compute $C[i, j], i \geq j \geq 1$. Illustrate by drawing a diagram (tree) the steps that your algorithm performs to compute $C[6, 4]$. Does your algorithm perform overlapping computations?

(c) Use dynamic programming to design an $O(n^2)$ time algorithm that computes the first n rows in Pascal's triangle.

3. Consider the two sequences $X = \langle A, C, T, C, C, T, G, A, T\rangle$ and $Y = \langle T, C, A, G, G, A, C, T\rangle$ of characters. Apply the Longest Common Subsequence algorithm to X and Y to compute a longest common subsequence of X and Y. Show your work (the contents of the table), and use the table to give a longest common subsequence of X and Y.

4. Textbook, pages 397, exercise number 15.4-5.

5. **The subset-sum problem.** Let $S = \{s_1, \ldots, s_n\}$ be a set of n positive integers and let t be a positive integer called the *target*. The subset-sum problem is to decide if S contains a subset of elements that sum to t. For example, if $S = \{1, 2, 4, 10, 20, 25\}$, $t = 38$, then the answer is YES because $25 + 10 + 2 + 1 = 38$. However, if $S = \{1, 2, 4, 10, 20, 25\}$, $t = 18$, then the answer is NO. Let $s = s_1 + \ldots + s_n$.

   (a) Let $T[0..n, 0..s]$ be a table such that $T[i, s'] = S'$ if there exists a subset of elements S' in $\{s_1, \ldots, s_i\}$ whose total value is s', and $T[i, s'] = \dagger$ otherwise; $\dagger$ is a flag indicating that no such S' exists. Show how $T[0, k]$ can be easily computed for $k = 0, \ldots, s$.

   (b) If $T[i, s']$ exists ($T[i, s'] \neq \dagger$) and element s_i does not belong to $T[i, s']$, how can the value of $T[i, s']$ be expressed using table entries in previous rows? What about when $T[i, s']$ exists and element s_i belongs to $T[i, s']$? Show how entry $T[i, s']$ can be computed from table entries in previous rows.

   (c) Design an $O(n \cdot s)$ time algorithm that decides if S contains a subset of elements A that sum to t.