Data Structures
Homework #2

Due: Nov 09, 2021

1. Please refer the code shown as below. Give the number of operations (steps) for each statement. Then, sum up these numbers to derive the total number of operations and present this total number using the asymptotic notations.

```
PROD(A[p][q], B[q][r])
1  for i = 1 to p
2      for j = 1 to r
3          C[i][j] = 0
4          for k = 1 to q
5              C[i][j] = C[i][j] + A[i][k] × B[k][j];
```

2. An array A contains $n-1$ unique integers in the range $[0, n-1]$. So, there is one number in this range that is not in A. Design an $O(n)$-time algorithm for finding that number. You are only allowed to use $O(1)$ additional space besides the array A itself. Please write your idea first. Then, present your algorithm in pseudo-code and analyze it in time and space.

3. Find a Big-O notation in terms of n for the number of times the statement $x = x + 1$ is executed in the following segment

```
1  j = n
2  while (j ≥ 1)
3      for i = 1 to j
4          x = x + 1;
5      j = j/2;
```

4. Please show the followings:

   (a) let x and y be real numbers with $0 < x < y$. Prove that n^x is $\mathrm{O}(n^y)$, but n^y is not $\mathrm{O}(n^x)$.

   (b) $\log_a n$ is $\mathrm{O}(\log_b n)$ for any real numbers $a > 1$ and $b > 1$.

5. Show that $O(max\{f(n), g(n)\}) = O(f(n) + g(n))$.

6. (**Programming**)

   The binomial coefficients may be defined by the following recurrence relation, which is the idea of ***Pascal's triangle***. The top of Pascal's triangle is shown in Figure 1.

$$\begin{aligned} C(n,0) &= 1 \quad \text{and} \quad C(n,n) = 1 && \text{for } n \geq 0 \\ C(n,k) &= C(n-1,k) + C(n-1,k-1) && \text{for } n > k > 0 \end{aligned}$$

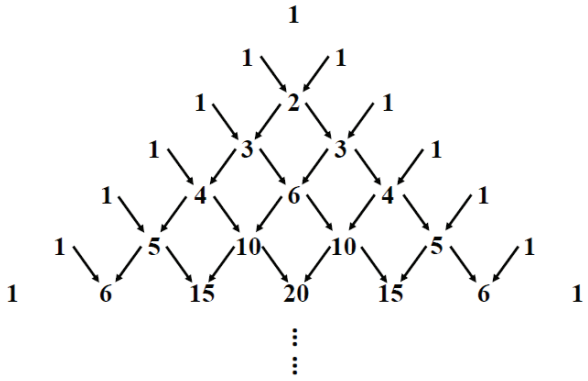

Figure 1: The top of Pascal's triangle of binomial coefficients

(a) Draw the recursion tree for calculating $C(6, 4)$.

(b) Write a function `PA_r` using *recursive* function to generate Pascal's triangle in the lower left half of the array.

(c) As (b), write a *nonrecursive* function `PA_n` to generate Pascal's triangle in the lower left half of the array.

(d) Again, write a nonrecursive program `PA_d` that uses neither an array to calculate each $C(n, k)$ directly.

(e) Please do some experiments by yourself to observe the running time and perform a comparison on these three versions by the execution time. Thus, you need to write a Python program to do the comparison, which will call `PA_r`, `PA_n`, and `PA_d`, respectively. Please see the execution example below. To measure the execution time. Please use Python's `timeit` module to do the measuring.

(f) Determine the approximate space and time requirements for each of the algorithms devised in parts (b), (c), and (d).

Execution Example

```
input the degree n:  6
Recursion execution time:  6.79966206962469e-05
Non-Recursion execution time:  5.8053718613296224e-05
Direct Computation execution time:  9.942902082935023e-06
input the degree n:
```

About submitting this homework

1. For problem 1, 2, 3, 4, and 5, Please

   (1) write all of your solutions on the papers of size A4,

   (2) leave you name and student ID on the first page, and

(3) hand in your solutions on papers to me in class (depending on the class on-line or practically)

Note: If we still have the on-line class, please have the written part in the given `.ipynb` file with the last programming problem.

2. For problem 6, please upload the completed `.ipynb` file with the filename as `HW2_studentID.ipynb` to *ischool* platform (`http://www.ischool.ntut.edu.tw/`).

3. The **deadline** is the **midnight of November 9**, 2021 and **Late work** is not acceptable.

4. Honest Policy: We encourage students to discuss their work with the peer. However, each student should write the program or the problem solutions on her/his own. Those who copy others work will get 0 on the homework grade.