# EC330 Applied Algorithms and Data Structures for Engineers
# Spring 2024

## Homework 1

**Out:** January 24, 2024
**Due:** February 7, 2024

*This homework has a written part and a programming part. Both are due at 11:59 pm on February 7. You should submit both parts on Gradescope.*

*This is an **individual** assignment with a **pair programming** component (Problem 5 Part b). See course syllabus for policy on collaboration.*

**1. Sums [10 pt]**

Provide a closed-form solution to the following problems. Make sure you show the steps.

a) $\sum_{i=1}^{33}(\frac{1}{3})^i$

b) $\sum_{i=1}^{N}(i^3 + 2i^2 - 3i + 4)$

**2. Exponents and Logs [5 pt]**

Simplify the following expression. Make sure you show the steps.

$log_{330}(330^{330} \cdot 330)$

**3. Combinatorics [5 pt]**

How many integer solutions of $x_1 + x_2 + x_3 = 12$ satisfy $x_1 \geq 1$, $x_2 \geq 3$ and $x_3 \geq -3$? Make sure you show the steps.

**4. Proof by Induction [10 pt]**

Consider the function $f$ defined as follows.

$$f(x) = x \qquad x = 1, 2, 3$$
$$f(x) = f(x-1) + f(x-2) + f(x-3) \qquad x \in \mathbb{N} \text{ and } x > 3$$

Show that $\forall x \in \mathbb{N}, f(x) < 2^x$.

**5. Programming [70 pt]**

*Make sure to acknowledge any source you consult at the top of your program.*
*Do not include a `main` in your submitted files. Do not modify the header files. For part b), you should make a group submission (for both you and your partner) on Gradescope.*

a) **[Individual]** Write a *recursive* program to generate the result of *fun(x, y)* for non-negative integer inputs *x* and *y*. The function *fun(x, y)* is defined as follows.

$$fun(x,y) = \begin{cases} 1 & x = 0 \; and \; y = 0 \\ y + 1 & x = 0 \; and \; y \neq 0 \\ fun(x-1,1) & x \neq 0 \; and \; y = 0 \\ fun(x-1, fun(x-1, y-1)) & otherwise \end{cases}$$

Your implementation must *not* make redundant recursive calls, e.g. if the value of $fun(2,3)$ has been computed before then your program should not compute it again.

Implement the *fun* function in *fun.cpp*. Try to make your algorithm as efficient as you can.

Submit your completed *fun.cpp* file on Gradescope. **[30 pt]**

b) **[Pair Programming]** Wenchao wants to divide the class into two (non-empty) project groups based on the students' birthdays. In particular, we are going to consider only the day of birth (i.e. an integer between 1 and 31) and not the month or the year. We want the two groups to be "balanced" in such a way that *the difference between the sum of birthdays of one group and the sum of birthdays of the other group is minimized*. For example, say the birthdays for students Alice, Bob, Charlie, Drew and Edward are 3, 27, 4, 5 and 20 respectively. The most balanced group assignment would be {Alice, Bob} and {Charlie, Drew, Edward} since | (3 + 27) - (4 + 5 + 20) | = 1 (where | | indicates taking the absolute value) is the smallest among all possible assignments.

Develop an *efficient* algorithm to help Wenchao determine the most balanced group assignment for the class. Implement the *balancedGroups* function in *balancedGroups.cpp* and submit this file on Gradescope. **[40 pt]**

*Hint: Think about the different cases for assigning a student to either of the two project groups.*