Quiz 1 (Practice)

This quiz is **open-book and closed-internet**. Feel free to use any physical materials you have brought, but you may not access resources online (except funprog). Proctors will be available to answer administrative questions and clarify specifications of coding problems, but they should not be relied on for coding help.

Each problem is worth 25 points, for a total of 100 points.

You *must* submit your quiz via funprog before the deadline in order to receive credit. This quiz assumes you have Python 2.7 installed on your machine.

The `resources` directory contains **Python documentation** for commonly-used data structures.

Problem 1: `is_unique`

Given a list of numbers, return a Boolean: `True` if the elements are unique (no element is repeated), and `False` otherwise.

Examples:

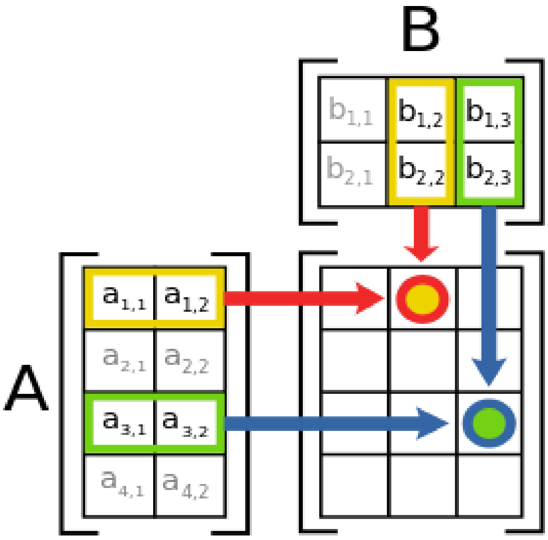
`is_unique([1, 2])` should return `True`
`is_unique([1, 1])` should return `False`

Problem 2: `matrix_product`

Implement `matrix_product(A, B, m, n, k)`.

Given a `(m x n)` matrix `A` and a `(n x k)` matrix `B`, both represented as lists in row-major order, return the `(m x k)` matrix product `AB` in row-major order. Recall that an `(R x C)` matrix has `R` rows (height), and `C` columns (width). When multiplying matrixes `A` and `B`, the *width* of `A` must equal the *height* of `B`.

Matrix product is defined as follows: for each row `i` and column `j` of the product `AB`, the element AB_{ij} = sum(A_{iz} * B_{zj}) for all possible values of `z`. Row `i` of `A` is multiplied by column `j` of `B`, and the result summed to produce the `ij` element of `AB`.



Note that `m`, `n`, and `k` are necessary specifications. Without them the matrix representation is ambiguous (e.g., the row-major list [1, 2] could represent either a `(1 x 2)` or `(2 x 1)` matrix).

Examples:

`matrix_product([1, 0, 0, 1], [1, 0, 0, 1], 2, 2, 2)` should return `[1, 0, 0, 1]`
`matrix_product([1, 0], [1, 1, 1, 1, 1, 1], 1, 2, 3)` should return `[1, 1, 1]`

Problem 3: `mode`

Given a list of numbers, return the mode (the most common value). If there is a tie for the most common value, return whichever the one that appears first in the list.

Examples:

`mode([1,2,2,3])` should return `2`
`mode([1,1,2,2,3])` should return `1`

Problem 4: `transpose`

Implement `transpose(A, m, n)`.

The transpose `T` of an `(m x n)` matrix `A` is an `(n x m)` matrix satisfying the following property: $A_{ij} = T_{ji}$ for all `i`, `j`. Each row of the input becomes a column in the output.

Given a matrix A in row-major order and its dimensions (m x n), return its transpose, also in row-major order.

Examples:

`transpose([1, 1, 0, 0], 2, 2)` should return `[1, 0, 1, 0]`

`transpose([1, 0, 0, 1], 2, 2)` should return `[1, 0, 0, 1]`