Lab 1: Integer Representations

This is an INDIVIDUAL assignment. Due date is as indicated on BeachBoard. Follow ALL instructions otherwise you will lose points. In this lab, you will be implementing two functions. This will require you to utilize the Python concepts that we have discussed so far and the algorithms of integer representation.

Instructions:

1. Take a close look at the `bin_oct_hex.py` file. There are two empty functions: `to_decimal(num, base)` and `to_base(dec_num, base)`. Read the instructions carefully! You will lose points if you do not follow the instructions. We are using a grading script!

`to_decimal(num, base)`	
input	• num (a non-negative non-decimal int as string)- ex: 11101, 7712, ABC • base (the number system you're converting from as an int)- ex: 2, 8, 16
output	• decimal representation of num AS INTEGER
restrictions	• You are NOT allowed to use the Python int function that will convert it to decimal for you. You must implement the algorithm discussed in class
assumptions	• num will always be non-negative • num will always be a valid number ex: 31112 (base2) won't be an input • if num has letters, they will always be capitalized • base will be 2, 8, or 16

`to_base(dec_num, base)`	
input	• dec_num (a positive decimal integer)- ex: 1, 6, 10, 68, 102... • base (the number system you're converting to as an int)- ex: 2, 8, 16
output	• non-base-10 representation of dec_num AS STR
restrictions	• You are NOT allowed to use the Python int function that will convert it to decimal for you. You must implement the algorithm discussed in class • If converting to hex, use capital letters only
assumptions	• num will always be non-negative • base will be 2, 8, or 16

2. Your job is to implement both of these functions so that it passes any test case. There are some sample test cases provided for you, but these are not the only cases that we will test. There will be 30 test cases in addition to the ones that you see.
3. **After completing these functions, comment out the test cases (or delete them) or else the grading script will pick it up and mark your program as incorrect.**
4. Convert your newly named `bin_oct_hex.py` file into a `.txt` file. Submit your newly named `bin_oct_hex.py` file and your `.txt` file on BeachBoard. Do NOT submit it in compressed folder.
5. Do not email us your code asking us to verify it. We will answer general questions, but we will not debug your code over email.

Grading rubric

Points	Requirement
2	Correct submission (2 files, not in any folder)- all or nothing
3	Passes the test cases listed in `bin_oct_hex.py` and followed instructions by deleting/commenting the test cases in the file (all or nothing)
10	Passes the remaining 30 test cases (you can get a fraction of these points)

If you use any of the following functions, you will get an AUTOMATIC ZERO!

- `bin(number)`
- `"{0:#b}".format(number)`
- `f"{number}"`
- `int(bin(x)[2:])`
- `int(number, 16)`
- `int(number, 8)`
- `int(number, 2)`

You are allowed to use the `int()` function, but not in the converting context!

This is not an exhaustive list. Essentially, any functions or libraries that we have not introduced in class is off limits. The purpose of this lab is to improve your algorithm knowledge and your Python programming. Again, do not attempt to make shortcuts that require outside libraries. If you are unsure, feel free to reach out to an ISA or the instructor.