Lab 2: Digits

Goal: Learn how to use integer and modular division in conjunction to isolate digits.

Required: Functions, Integer/Modular Division

Explanation: People use base 10 notation, meaning that each digit has 10 different possible values. The 10 possible values range from ___ to ___ . When learning arithmetic, special attention should have been paid to multiplication and division by 10 because they could be done by moving digits around.

Ex: Use regular division, not integer division.

1. 1021 / 10 = _____ **2. 314 / 100 = _____** **3. 1234 / _____ = 1.234**

A common way to work with digits in computer science is to use integer and modular division by powers of 10. Consider 12345 / 100. Integer division gives you 123, while modular division gives you 45. We can think of integer division as giving the part left of the decimal, and modular division as giving the part right of the decimal. Remember, this only works when dividing by powers of 10! (It actually works for other bases too, if you are comfortable working in other bases.)

Task 1: Create a function that returns the ones digit from an integer.

int onesDigit(int a); // Ex, onesDigit(1234) => 4

Task 2: Create a function that returns the tens AND ones digit from an integer.

int tensOnesDigit(int a); // Ex, tensOnesDigit(1234) => 34

Task 3: Looking at Task 2 and the example that returns 34, how can we isolate just the tens digit without including the ones?

int tensDigit(int a); // Ex, tensDigit(1234) => 3

Task 4: Create 2 functions that returns the hundreds and thousands digit from an integer.

int hundredsDigit(int a); // Ex, hundredsDigit(1234) => 2

int thousandsDigit(int a); // Ex, thousandsDigit(1234) => 1

Task 5: Create a function that counts the number of digits in a positive integer.

int countDigits(int a); // Ex, countDigits(1234) => 4

Task 6: Create a function that counts the number of even digits in a positive integer.

int countEvenDigits(int a); // Ex, countEvenDigits(1234) => 2

DO NOT SHARE YOUR ANSWER FOR THE NEXT PART, LET OTHERS FIGURE IT OUT ON THEIR OWN!!!

Task 7: Create a generic function that returns digits in a given range. The ones place corresponds to power 0, the tens place corresponds to power 1, the hundreds place corresponds to power 2, and so on. Think 10^3 => 1000 => thousands place. (Do not use exponents in your code.)

// Ex, getDigits(12345,0,0) => 5 // Ex, getDigits(12345,5,5) => 0

// Ex, getDigits(12345,1,1) => 4 // Ex, getDigits(12345,4,1) => 1234

// Ex, getDigits(12345,4,4) => 1 // Ex, getDigits(12345,3,0) => 2345

int getDigits(int a, int biggestDigit, int smallestDigit);