Generic counting function:

```
def some_counting_function(string):
  total = 0
  for pos in range(0, len(string)):
    if <test string[pos] for something>:
      total = total + 1
  return total
```

Generic filtering function:

```
def some_filtering_function(string):
  answer = ""
  for pos in range(0, len(string)):
    if <test string[pos] for something>:
      answer = answer + string[pos]
  return answer
```

The generic count/filter functions can be altered in various ways to build more sophisticated functions, for instance, by using multiple counter variables or if/elif/else rather than just if.

Practice:

1. Write a function called count_digits that returns the number of digits in a string.

   Example: count_digits("abc123def5") returns 4

   Hint: the definition line will look like: `def count_digits(string):`

2. Write a function called filter_digits that returns only the digits from a string.

   Example: filter_digits("abc123def5") returns "1235"

3. Write a function called sum_digits that returns the sum of all the digits in a string.

   Example: sum_digits("abc123def5") returns 11

4. Write a function called reverse that returns (**not prints**) the reverse of the string parameter.

   Example: reverse("abc") returns "cba"

5. Write a function called remove_capitals that returns the string with capital letters removed.

   Example: remove_capitals("AbCDeFGhi9") returns "behi9"

6. Write a function called switch_capitals that returns the string with all lowercase letters switched to uppercase and all uppercase letters switched to lowercase. Anything that is not an uppercase or lowercase character remains unchanged.

   Example: switch_ capitals("AbCDeFGhi9") returns "aBcdEfgHI9"

7. Write a function called count_first that counts the number of characters in a string that are identical to the first character.

   Example: count_first("purple") returns 2

8. Write a function called count_distinct that counts the number of distinct characters in a string. In other words, count the total number of different characters that make up the string.
   Example: count_unique("abracadabra") returns 5.

   Hint: Think about how you would solve this on paper. If I give you a string, and you look at each character in the string left to right, how can determine if you've already counted this character or not? Hint 2: If you are looking at the character at position `p`, take a slice from position `0` to `p`.

9. Write a function called count_2later that counts the number of characters in a string that match the character *two* positions later.

   Example: count_2later("abracadabra") returns 2. [there are two "a"s that match another "a" 2 characters later]