Generic counting function:

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

Generic filtering function:

```
def some_filtering_function(s):
  answer = ""
  for pos in range(0, len(s), 1):
    if <test s[pos] for something>:
      answer = answer + s[pos]
  return answer
```

Generic filtering with multiple branches:

```
def some_filtering_function(s):
  answer = ""
  for pos in range(0, len(s), 1):
    if <test s[pos] for something
      answer = answer + <something>
    else:
       answer = answer + <something else>
  return answer
```

Filtering & counting practice:

1. Write a function called remove_capitals that returns the string s with capital letters removed.
   Example: remove_capitals("AbCDeFGhi9") returns "behi9"

2. Write a function called change_nums that increments all numbers in a string by one:
   Example: change_nums("a1b2") returns "a2b3"
   We guarantee that this function will never have strings containing numbers greater than 8.

3. Write a function called reverse that returns (not prints) the reverse of string s.
   Example: reverse("abc") returns "cba"

4. Write a function called encode that takes a string and encodes it using the simple cipher A=1, B=2, C=3, and so on. Make this work with uppercase and lowercase letters.
   Example: encode("abc") returns "1-2-3".

   Hint: use a variable letters = "abcdefgh…" and the find function.
   What is letters.find("a")? letters.find("b")?

5. 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

6. Challenge (hard): write a decode function that decodes a string like “1-2-3” back into “abc”.