1. **The `in` keyword checks whether a substring is in a string:**

"a" in "classroom" — True

To check if character c is a vowel:

if c in "aeiouAEIOU"

in works for multi-char strings:

if "xy" in "wxyz" — True

2. **Python has built-in functions `find()` and `rfind()` to find the position of a substring within a string.** ↖reverse find

"classroom".find("a") returns 2
(0 1 2)

"classroom".find("oo") returns 6
(0 1 2 3 4 5 6)

"classroom".rfind("o") returns 7
(7)

EXAMPLE:

```
mystring = "classroom"
while "o" in mystring:
    i = mystring.find("o")
    mystring = mystring[:i] + mystring[i+1:]
print(mystring)
```

3. **Write a function that reverses its string argument.**
Hint: Use the accumulator pattern!

```
def reverse(mystring):
    newstring = ""
    i = -1
    n = len(mystring)
    for j in range(n):
        newstring = newstring + mystring[i]
```

PLAN:
initialize accumulator to ""
take characters in mystring from back to front:
append to the accumulator

```
    newstring = newstring + mystring[i]
    i = i - 1
return newstring
```

EXAMPLE: mystring = "classroom"

accumulator: moorssalc

```
def reverse(mystring)
    newstring = ""
    for i in range(-1, -n, -1):
        newstring = newstring + mystring[i]
    return newstring
```

step

```
def reverse(mystring):
    newstring = ""
    n = len(mystring)
    for i in range(-1, -(n+1), -1):
        print("i:", i, "; character is:", mystring[i])
        newstring = newstring + mystring[i]
    return newstring

print(reverse("classroom"))
```

4. The string module provides useful strings of characters of certain types:

→ import string

string.ascii_lowercase — "abcdefghijklmnopqrstuvwxyz"

if c in string.ascii_lowercase:

string.ascii_uppercase — "ABC..."

string.ascii_letters

string.digits — "0123456789"

string.punctuation — . , : _ ! etc.

PRACTICE WITH STRINGS – SOLUTIONS

CS 125

Working with a partner/group, use the following steps to solve each of the following problems.

(a) Plan your function on the white board (either on the classroom wall or on Zoom). Write out your entire program. Think about what errors might occur and how to fix them.

(b) Plan multiple test cases for your function. What input will you send to your function? What value should the function return?

(c) *Only after you have completed steps (a) and (b) should you type your code in Python.*

(d) After you have typed your function, run your test cases. Does your function work? If not, how can you fix it?

1. Write a function `removeNonAlpha(mystr)` that removes all non-alphabetic characters from a string `mystr`. That is, your function should accept a string of text, and then return a string containing the same text but with all non-alphabetic characters removed. For example:

   `removeNonAlpha("abc123!xyz")` returns `"abcxyz"`

```
import string
def removeNonAlpha(mystr):
    alpha = string.ascii_lowercase + string.ascii_uppercase
    newstr = ""
    for c in mystr:
        if c in alpha:
            newstr = newstr + c
    return newstr

#testing
print(removeNonAlpha("abc123ABC!=+"))
print(removeNonAlpha("q8A(p^L."))
```

2. Write a function `removeSubstr(mystr, sub)` that removes the first occurrence of string `sub` from string `mystr`. For example:

   `removeAll("Mississippi", "ss")` returns `"Miissippi"`

```
def removeSubstr(mystr, sub):
    i = mystr.find(sub)  # index of sub, or -1 if not found
    if i == -1:
        return mystr
    newstr = mystr[:i] + mystr[i+len(sub):]
    return newstr
```

```
#testing
print(removeSubstr("abcdefg","bc"))
print(removeSubstr("abcdefg","xy"))
print(removeSubstr("abcdabcd","ab"))
```

3. Write a function that extracts text from inside of HTML tags. Your function declaration should be:

   `def extractText(HTMLstring):`

   A call to `extractText("<i>some text</i>")` should return "`some text`".

   Hint: consider the `find` and `rfind` Python string methods.

```
def extractText(HTMLstring):

    #find the first > in the string
    i = HTMLstring.find(">")

    #find the last < in the string
    j = HTMLstring.rfind("<")

    return HTMLstring[i+1:j]

#testing
print(extractText("<i>some text</i>"))
print(extractText("<b>yay</b>"))
```

4. Write a function that converts a word to pig Latin. The procedure for converting a word to pig Latin is as follows:

   - If the word begins with a constant, then all letters before the initial vowel are moved to the end of the word, and then "ay" is added to the end. For example, "pig" becomes "igpay", and "glove" becomes "oveglay".
   - If the word begins with a vowel, then add "yay" to the end. For example, "eat" becomes "eatyay".

```
def toPigLatin(word):
    newstr = ""
    vowels = "aeiouAEIOU"
    if word[0] in vowels:
        newstr = word + "yay"
    else:
        i=0
        while (word[i] not in vowels):
            i = i + 1
        newstr = word[i:] + word[:i] + "ay"
    return newstr
```