Programming with Python Feb 14 2025

Note: Brief Summary of contents discussed.

Strings: Sequence of characters enclosed within quotes (single, double or triple quotes).

```
>> msg = 'Hello World'
>> msg = "Hello World"
>> msg = ```Hello
            World```
```

- Triple quotes are typically used for stings spanning multiple lines.
- string containing single quotes can be enclosed with double quotes and vice-versa. E.g. `"It's a sunny day!"`

```
>>len('Hello World') #length of a string
11
```

Indexing: indices start from 0 (left or right) or -1 (from right to left)

```
>> msg = 'hello world'
>> msg[0]
>> msg[-1]
>> msg[20] #string out of range, not a problem when
slicing
>> msg[-15] #string out of range
```

Strings in Python are **immutable**(Assignment to part of string gives error)

```
>> msg[1] = 'H' #error
>> msg = "Bye world" #why this doesn't give errors.
```

String concatenation(+) and Repetition(*)

```
'Hello' + 'world'
'Hello' * 3
```

max/min

```
max('AB', 'BC', 'AC')
min('ABCD', 'BCD, 'CA')
```

Slicing [start:end:inc/dec]
Forward (0:) and backward indexing(:-1)

```
msg = 'hello world'
msg[0: 4]
msg[:4]
msg[2:5]
msg[-1]
msg[-1:-5]
msg[::-1] #string reverse
```

Membership

```
'H' in 'Hello' #True
#space separated string
st = ''
for c in 'python':
    st = st + c + ' '
```

Built-in Functions on Strings: Some common built-in string methods are listed below with examples for a few of them. Explore python string documentation for details.

count

```
'hello'.count('l') #2
#vowel count
vowels = 'aeiouAEIOU'
count = 0
for c in vowels:
    count += 'Programming with Python'.count(c)
```

#compare it with earlier program where we looped over the input strings instead of vowels

find/rfind

```
colors = 'green, red, blue, red'
colors.find('red') #1, first location, -1 if not found
colors.rfind('red') #3
```

lower, upper

```
'heLLo'.lower()
'heLLo'.upper()
```

islower, isupper, isspace, isdigit, isalnum

```
‘    \n \t  ‘.isspace() #True
```

replace

```
‘abcdefab’.replace(‘ab’, ‘aa’) #’aacdefab’
```

strip

```
‘     hello world      ‘.strip() #’hello world
```

split

```
colors =’red, green, blue’
colors.split(‘,’) #[‘red’, ‘ green’, ‘ blue’]
```

join

```
‘+’.join(‘abc’) #’a+b+c’
```

startswith/endswith

```
‘helloooo’.startswith(‘he’) #True
```

Example 1

Count the number of common characters in a pair of strings only if the character has not been encountered before:

```
for i in range(len(t1)):
    ch1 = t1[i]
    if not (ch1 in t1[:i]):
        for ch2 in t2:
            if ch1 == ch2:
                count = count+1
                break
return count
```

Here's a table with common Python string functions and their explanations:

Function	Explanation
`str.lower()`	Converts all characters in a string to lowercase.
`str.upper()`	Converts all characters in a string to uppercase.
`str.title()`	Converts the first character of each word to uppercase.
`str.capitalize()`	Capitalizes the first character of the string.
`str.strip()`	Removes leading and trailing whitespace.
`str.lstrip()`	Removes leading whitespace.
`str.rstrip()`	Removes trailing whitespace.
`str.replace(old, new)`	Replaces all occurrences of `old` with `new`.
`str.find(sub)`	Returns the index of the first occurrence of `sub`, or -1 if not found.
`str.index(sub)`	Similar to `find()`, but raises an error if `sub` is not found.
`str.count(sub)`	Returns the number of times `sub` appears in the string.
`str.split(sep)`	Splits the string into a list of substrings based on `sep`.
`str.join(iterable)`	Joins elements of `iterable` into a single string, using the string as a separator.
`str.startswith(prefix)`	Returns `True` if the string starts with `prefix`.
`str.endswith(suffix)`	Returns `True` if the string ends with `suffix`.
`str.isdigit()`	Returns `True` if the string contains only digits.
`str.isalpha()`	Returns `True` if the string contains only alphabetic characters.
`str.isalnum()`	Returns `True` if the string contains only alphanumeric characters.
`str.isspace()`	Returns `True` if the string contains only whitespace.
`str.zfill(width)`	Pads the string with zeros on the left to match `width`.

Exercises:

1. Write a function that takes two strings and returns `True` if they are anagrams and `False` otherwise. A pair of strings is anagrams if the letters in one word can be arranged to form the second one.

2. Write a program to remove all vowels from a given string.

3. Write a program to reverse the order of words in a sentence.

4. Write a program to extract and print all digits from a given string.

5. Write a function that takes a sentence as input parameter and returns the number of words in the sentence.

6. Write a function that takes a string as a parameter and returns a string with every successive repetitive character replaced with a star (`*`). For example `'balloon'` is returned as `'bal*o*n'`.

7. Write a program to display the most frequently occurring character in a string.

8. Write a program to:
    a. Remove all spaces from a given string.
    b. Check if a given substring is present in a string.
    c. Convert the first letter of each word in a sentence to uppercase.

9. Write a program to find the longest word in a given sentence.

10. Write a program to remove duplicate characters from a string while maintaining order.

Note:

- Read about docstrings
  https://peps.python.org/pep-0257/#what-is-a-docstring
  https://stackoverflow.com/questions/19074745/docstrings-vs-comments