COMP 141

Strings III

Rhodes College

Announcements

Reminders:

Program 6 - due 11/3 (Thursday)

Rhodes College

Practice From Last Time

- Write a function called **`total_seconds`** that takes one string argument. This argument will be a string of the form "M:SS" where M is a number of minutes (a single digit) and SS is a number of seconds (2 digits). This function should calculate the total number of seconds in this amount of time and **return** it as an integer.
- Write a function called **`count_digits`** that returns the number of digits in a string.
  - `count_digits("abc123def5") returns 4`
- Write a function called sum_digits that returns the sum of all the digits in a string.
  - `sum_digits("abc123def5") returns 11`
  `(because 1 + 2 + 3 + 5 = 11)`

Rhodes College

String Concatenation

- **Combines two strings into a new, longer string**
- **Uses the same plus sign as addition**

```
s1 = "CS141"
s2 = "rocks!"
bigstring = s1 + s2
print(bigstring) #prints CS141rocks!
```

Rhodes College

String Concatenation

- Unlike print(), string concatenation does not put spaces between your strings.

```
s1 = "CS141"
s2 = "rocks!"
bigstring = s1 + " " + s2
print(bigstring) #prints CS141 rocks!
```


Sample Problem

- All professor email addresses at Rhodes are constructed from the professor's last name, followed by the first initial of their first name.
- We want to design a function that take's a prof's first and last name and returns their email address.

Sample Problem Solution

```
def make_prof_email(first, last):
    init = first[0]
    address = last + init + "@rhodes.edu"
    return address

def main():
    firstname = input("First name: ")
    lastname = input("Last name: ")
    addr = make_prof_email(firstname, lastname)
    print("Email:", addr)
```


The Repetition Operator

- **Repetition operator:** makes multiple copies of a string and joins them together
  - The * symbol is a repetition operator when applied to a string and an integer
    - String is left operand; number is right
  - General format: *string_to_copy * n*
  - Variable references a new string which contains multiple copies of the original string

```
s = "a"
s2 = s * 10
print(s2)    #Output is aaaaaaaaaa
```

Other String Methods

- **Programs commonly need to search for substrings**
- **Several methods to accomplish this:**
  - `endswith(substring)`: checks if the string ends with *substring*
    - Returns `True` or `False`
  - `startswith(substring)`: checks if the string starts with *substring*
    - Returns `True` or `False`

Rhodes College

More String Methods

- **Several methods to accomplish this (cont'd):**
  - `find(substring)`: searches for *substring* within the string
    - Returns lowest index of the substring, or if the substring is not contained in the string, returns -1
  - `replace(substring, new_string)`:
    - Returns a copy of the string where every occurrence of *substring* is replaced with *new_string*

Rhodes College

Using the `find` method

```
def main():
    filename = "First Last_assignsubmission_file_lastname_firstname_prg6.py"
    print(renameFile(filename))

def renameFile(fileName):
    ind = fileName.find("file_")
    fileName = fileName[ind+5:]
    return fileName

main()
```

Output:
`lastname_firstname_prg6.py`

Rhodes College

String Methods

Table 9-3 Search and replace methods

Method	Description
`endswith(substring)`	The *substring* argument is a string. The method returns true if the string ends with *substring*.
`find(substring)`	The *substring* argument is a string. The method returns the lowest index in the string where *substring* is found. If *substring* is not found, the method returns −1.
`replace(old, new)`	The *old* and *new* arguments are both strings. The method returns a copy of the string with all instances of *old* replaced by *new*.
`startswith(substring)`	The *substring* argument is a string. The method returns true if the string starts with *substring*.

Rhodes College

Testing, Searching, and Manipulating Strings

- **You can use the `in` operator to determine whether one string is contained in another string**
  - General format: *`string1 in string2`*
    - *`string1`* and *`string2`* can be string literals or variables referencing strings
- **Similarly you can use the `not in` operator to determine whether one string is not contained in another string**

Splitting a String

- **`split` method:** returns a list containing the words in the string
  - By default, uses space as separator
  - Can specify a different separator by passing it as an argument to the `split` method

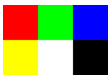

```
#This program calls the split method, using the
# '/' character as a separator

def main():
    date_string = '4/1/2016'

    month, day, year = date_string.split('/')

    print("Month: ", month)
    print("Day: ", day)
    print("Year: ", year)


main()
```

Program Output
Month: 4
Day: 1
Year: 2016

PPM File Format

- PPM = portable pixmap format
- Image file format

```
P3
# The P3 means colors are in ASCII, then 3 columns and 2 rows,
# then 255 for max color, then RGB triplets
3 2
255
255   0   0     0 255   0     0   0 255
255 255   0   255 255 255     0   0   0
```



Converting a PPM image to Black and White

Algorithm

- Assume no comments in our PPM file
- Create a new ppm file to hold the B/W image
- Read in first 3 lines of color PPM file and write them directly out to our B/W file.
- For each line after first 3:
  - Read in RGB values
  - Find the average of these values
  - Write the new pixel value to the B/W file (replace the RGB values with the average)
- Close the B/W PPM file
- Close the Color PPM file
- Test the new PPM file in our viewer

Rhodes College

Practice

- **Write a function that returns a Rhodes student email address. (Assume this email address is for a new student). Your function will need to take in 4 arguments: first name, last name, middle name and class year.**
- **Write a function called reverse that takes a string argument and returns the string argument with all characters in the reverse order.**
  - `reverse("Welsh") returns "hsleW"`
- **Write a function called filter_digits that returns only the digits from a string.**
  - `filter_digits("abc123def5") returns "1235"`
- **Write a function called count_unique that counts the number of unique characters in a string.**
  - `count_unique("abracadabra") returns 5.`
- **Write a function called count_dups that counts the number of back-to-back duplicated characters in a string.**
  - `count_dups("balloon") returns 2`