Notes: Strings and loops

COSC 101, 2018-02-26

Announcements

● Exam #1 tomorrow @ 7pm in 207 Lathrop Hall

Outline

● Warm-up
● String methods
● String comparison
● Iterating by character
● Iterating by index
● Accumulator pattern

Warm-up

*What is the output of this program?*

```
phrase = 'Once upon a time'
print(phrase[1], phrase[-1])
print(phrase[5:7], phrase[-6:-5])
print(phrase[:4], phrase[12:])
```

Output:

```
n e
up a
Once time
```

String methods

● Can perform operations on string, similar to performing operations on a turtle
  ○ Example:
    ```
    medal = 'silver'
    print(medal.upper())  # Outputs SILVER
    ```

● Common operations
  ○ `upper`, `lower` --- converts all characters to upper or lower case
  ○ `strip`, `lstrip`, `rstrip` --- removes leading/trailing whitespace (space, tab, newline)
  ○ `replace` --- replace all occurrences of one substring with another
  ○ `find`, `rfind` --- get index where substring is found

String comparison

● Equality --- use double equals (==) to check if the strings are exactly the same
● Ordering --- use less than (<), greater than (>), etc. to determine whether one string comes before or after the other according to the lexicographical order
  ○ Lexicographic order is based on the numeric value a computer uses to represent a character
  ○ `ord` --- gets the numeric value for a character
    ```
    Example
    print(ord('a'), ord('z'))
    print(ord('A'), ord('Z'))
    print(ord('0'), ord('9'))
    print(ord(',', '.'))
    ```
chr --- gets the character for a numeric value

Example

```python
print(chr(97), chr(122))
print(chr(65), chr(90))
print(chr(48), chr(57))
print(chr(44), chr(46))
```

Output:

```
a z
A Z
0 9
, .
```

Contains --- use `in` to determine whether one string contains another; use `not in` for the inverse

Iterating by character

- for loop is used to process each item in a sequence
  - Examples
    - for item in ['apple', 'banana', 'cherry']
    - for item in range(3)
  - String is a sequence of characters → use a for loop to process each character in a string
  - Example
    ```python
    word = 'winter'
    for character in word:
        print(character)
    ```
    Output:
    ```
w
i
n
t
e
t
r
```

What is the output of the following programs?

a) phrase = '12cd'
   for character in phrase:
       print(ord(character)) # Hint: ord('a') is 97 and ord('0') is 48
   Output:
   ```
   49
   50
   99
   100
   ```

b) word = 'three two one blast-off!'
   for character in word:
       if character == ' ':
           print('...', end='')
else:
    print(character, end='')

Output:
three...two...one...blast-off!

c) word = 'meow'
vowels = 'aeiou'
for character in word:
    if character in vowels:
        print(character*3, end='')
    else:
        print(character, end='')

Output:
meeeeoow

Interating by index

- Each character in a string is assigned an index → use a sequence of indices (integers) to process characters in a string
- Example
  word = 'winter'
  for index in range(len(word)):
      print(word[index])

  Output:
w
  i
  n
  t
  e
  r

- What is the output of the following programs?
  a) word = 'WIKIPEDIA'
     for i in range(len(word)):
         if i % 3 == 0:
             print(word[i])

     Output:
     W
     I
     D

  b) word = 'magical'
     for i in range(len(word)):
         if i % 2 == 0:
             print(word[i].upper(), end='')
         else:
             print(word[i], end='')

     Output:
     MagiCaL

  c) phrase = 'I do not understand'
     stop = (len(phrase) + 1) * -1
     for i in range(-1, stop, -1):

Accumulator pattern

- Previously, we have iteratively computed numeric results --- e.g., sum, factorial
- We can also build a string by iteratively concatenating one or more characters onto an existing string
- Example
  
  ```python
  word = 'winter'
  new_word = ''
  for character in word:
      new_word = new_word + character.upper()
  print(new_word)
  Output:
  WINTER
  ```

- What is the output of the following programs?
  a) letters = 'lbm'
     noun = 'A'
     for ch in letters:
         noun = noun + ch + 'a'
     print(noun)
     Output:
     Alabama
  b) letters = 'ssp'
     noun = 'M'
     for ch in letters:
         noun = noun + 'i' + ch * 2
         noun = noun + 'i'
     print(noun)
     Output:
     Mississippi
  c) k = ''
     for i in ['alice', 'bob', 'carlos']:
         for j in i:
             k = k + j.upper()
         k = k + ''
     print(k)
     Output:
     ALICE BOB CARLOS

Programming practice

Write a function called `pig_latin` that takes a single word (as a string) as an argument and translates the word to pig latin. To translate a word to pig latin, move the first letter to the end of the word and add “ay” to the end. For example:

- `pig_latin('pig')` returns 'igpay'
- `pig_latin('latin')` returns 'atinlay'