CSCI 141
Computer Programming I

Filip Jagodzinski
Announcements

- HW 5 (second to last homework)
  - Posted to the course website

Programming task 1: DJ Name

Santa Clause

Sause Sizzle

How might you retrieve the first two letters of Santa?

- Can you slice?
- Can you use a while loop?
- Can you use a for loop?
Announcements

• HW 5 (second to last homework)
  • Posted to the course website

How might you retrieve the first two letters of Santa?

Santa

As a first step, identify the “location” number for each letter

• Can you slice?
  • Can you use a while loop?
  • Can you use a for loop?
Announcements

- HW 5 (second to last homework)
  - Posted to the course website

How might you retrieve the first two letters of *Santa*?

- Can you slice?
  - Can you use a while loop?
  - Can you use a for loop?

Now that you know the indices, you can use slice by specifying the correct starting and ending positions of the desired slice.

Q: What is the code that uses slice to print Sa
Announcements

- HW 5 (second to last homework)
  - Posted to the course website

How might you retrieve the first two letters of Santa?

- Can you slice?
- Can you use a while loop?
- Can you use a for loop?

Now that you know the indices, you can use slice by specifying the correct starting and ending positions of the desired slice

```python
word = "Santa"
print(word[0:2])
print(word[-5:-3])
```
Announcements

- HW 5 (second to last homework)
  - Posted to the course website

How might you retrieve the first two letters of Santa?

- Can you slice?
- Can you use a while loop?
- Can you use a for loop?

Q: What is the code that uses while to print Sa
Announcements

• HW 5 (second to last homework)
  • Posted to the course website

How might you retrieve the first two letters of Santa?

• Can you slice?
  • Can you use a while loop?
  • Can you use a for loop?

```python
word = "Santa"
newWord = ""
currentChar = 0
while (currentChar < 2):
    newWord = newWord + word[currentChar]
    currentChar = currentChar + 1
print(newWord)
```
Announcements

- HW 5 (second to last homework)
  - Posted to the course website

How might you retrieve the first two letters of Santa?

- Can you slice?
- Can you use a while loop?
- Can you use a for loop?

Q: What is the code that uses for to print Sa
Announcements

• HW 5 (second to last homework)
  • Posted to the course website

How might you retrieve the first two letters of Santa?

• Can you slice?
• Can you use a while loop?
• Can you use a for loop?

Live demo
Announcements

- HW 5 (second to last homework)
  - Posted to the course website

How might you retrieve the first two letters of Santa?

- Can you slice?
- Can you use a while loop?
- Can you use a for loop?

Take home message: Now that you have for, while, and slice, there are multiple ways to achieve the same outcome.
Q: Which of the following lines of code print the same thing to the screen?

```python
mySentence = "Fred"
A. print(mySentence[0])
B. print(mySentence[0:])
C. print(mySentence[:0])
D. print(mySentence[-1])
E. print(mySentence[-1:0])
F. print(mySentence[len(mySentence)-1])
G. print(mySentence[0:-1])
H. print(mySentence[-1:-2])
I. print(mySentence[-2:-1])
J. print(mySentence[-4])
```
Warmup

Q: Which of the following lines of code print the same thing to the screen?

```python
mySentence = "Fred"
A. print(mySentence[0])
B. print(mySentence[0:])
C. print(mySentence[:0])
D. print(mySentence[-1])
E. print(mySentence[-1:0])
F. print(mySentence[len(mySentence)-1])
G. print(mySentence[0:-1])
H. print(mySentence[-1:-2])
I. print(mySentence[-2:-1])
J. print(mySentence[-4])
```

Go through Choices A through J, and determine what is printed.
Warmup

Q: Which of the following lines of code print the same thing to the screen?

```
mySentence = "Fred"
A. print(mySentence[0])    F
B. print(mySentence[0:])    D
C. print(mySentence[:0])    D
D. print(mySentence[-1])    D
E. print(mySentence[-1:0])  F
F. print(mySentence[len(mySentence)-1])    D
G. print(mySentence[0:-1])  D
H. print(mySentence[-1:-2]) D
I. print(mySentence[-2:-1]) F
J. print(mySentence[-4])    F
```

Print statements that access a single location of a string are easiest
Warmup

Q: Which of the following lines of code print the same thing to the screen?

```python
mySentence = "Fred"
A. print(mySentence[0])
B. print(mySentence[0:])
C. print(mySentence[:0])
D. print(mySentence[-1])
E. print(mySentence[-1:0])
F. print(mySentence[len(mySentence)-1])
G. print(mySentence[0:-1])
H. print(mySentence[-1:-2])
I. print(mySentence[-2:-1])
J. print(mySentence[-4])
```

Print statements that do not include one of the operands to the slice operator default to 0
Warmup

Q: Which of the following lines of code print the same thing to the screen?

```
mySentence = "Fred"

A. print(mySentence[0])
B. print(mySentence[0:])
C. print(mySentence[0:0])
D. print(mySentence[-1])
E. print(mySentence[-1:0])
F. print(mySentence[len(mySentence)-1])
G. print(mySentence[0:-1])
H. print(mySentence[-1:-2])
I. print(mySentence[-2:-1])
J. print(mySentence[-4])
```

The slice operator ALWAYS takes the value to the left of :, treats that as the starting location, and then proceeds (left-to-right in the string) to the index on the right side of :
Warmup

Q: Which of the following lines of code print the same thing to the screen?

```python
mySentence = "Fred"
```

<table>
<thead>
<tr>
<th>Option</th>
<th>Code</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>A.</td>
<td><code>print(mySentence[0])</code></td>
<td>F</td>
</tr>
<tr>
<td>B.</td>
<td><code>print(mySentence[0:])</code></td>
<td>Fred</td>
</tr>
<tr>
<td>C.</td>
<td><code>print(mySentence[:0])</code></td>
<td>d</td>
</tr>
<tr>
<td>D.</td>
<td><code>print(mySentence[-1])</code></td>
<td>d</td>
</tr>
<tr>
<td>E.</td>
<td><code>print(mySentence[-1:0])</code></td>
<td>d</td>
</tr>
<tr>
<td>F.</td>
<td><code>print(mySentence[len(mySentence)-1])</code></td>
<td>d</td>
</tr>
<tr>
<td>G.</td>
<td><code>print(mySentence[0:-1])</code></td>
<td>Fre</td>
</tr>
<tr>
<td>H.</td>
<td><code>print(mySentence[-1:-2])</code></td>
<td>e</td>
</tr>
<tr>
<td>I.</td>
<td><code>print(mySentence[-2:-1])</code></td>
<td>e</td>
</tr>
<tr>
<td>J.</td>
<td><code>print(mySentence[-4])</code></td>
<td>F</td>
</tr>
</tbody>
</table>

A and J
C and E and H
D and F
Strings are Immutable
Traversing Strings
Strings are immutable

Immutable ... what does that mean?
Strings are immutable

A unique property of Strings that we haven’t seen before. Which of the following are allowed?

```python
myVariable = 32
myVariable = 324

aPhrase = "Hello World"
aPhrase[6] = "w"
```
Strings are immutable

A unique property of Strings that we haven’t seen before. Which of the following are allowed?

```java
myVariable = 32
myVariable = 324
aphrase = "Hello World"
aphrase[6] = "w"
```

This is not allowed because strings CANNOT be modified once created. So can we change the value of a string?
Strings are immutable

A unique property of Strings that we haven’t seen before. Which of the following are allowed?

```java
myVariable = 32
myVariable = 324
aPhrase = "Hello World"
aPhrase[6] = "w"
```

This is not allowed because strings CANNOT be modified once created. So can we change the value of a string?

No. But we can create a new string and have an “old” variable refer to the new string object.

On the board explanation

```java
aPhrase = "Hello World"
aPhrase = "Hello world"
```
Strings are immutable

This means that once a String is created, you cannot change the value of any one character in the string.
Strings are immutable

This means that once a String is created, you cannot change the value of any one character in the string

Task: Locate the error(s) in the below code

```python
aString = "CSCI141 WWU"
print(aString[3])
print(aString[3] + aString[-2])
print(aString[0:] * 3)

aString[0] = "c"
aString[0:2] = "cs"
aString = "c"
aString = "cs"
```
Strings are immutable

This means that once a String is created, you cannot change the value of any one character in the string.

Task: Locate the error(s) in the below code

```python
aString = "CSCI141 WWU"
print(aString[3])
print(aString[3] + aString[-2])
print(aString[0:] * 3)

aString[0] = "c"
aString[0:2] = "cs"

aString = "c"
aString = "cs"
```

These are all allowed because recall that slice and retrieving a single character does NOT change the value of the character(s) that is/are retrieved.
Strings are immutable

This means that once a String is created, you cannot change the value of any one character in the string.

Task: Locate the error(s) in the below code

```python
aString = "CSCI141 WWU"
print(aString[3])
print(aString[3] + aString[0])
print(aString[0:2] * 3)
```
Strings are immutable

This means that once a String is created, you cannot change the value of any one character in the string.

Task: Locate the error(s) in the below code

| aString = "CSCI141 WWU"
| print(aString[3])
| print(aString[3] + aString[-2])
| print(aString[0:2] * 3)

These lines are trying to change the value of a character in an already created String, which is NOT allowed.

These are creating new String objects and reassigning the variables to point to Those new objects (on the board explanation).

Recall that slice and retrieving a single character does NOT change the value of the character(s) that is/are retrieved.
Traversing strings

```python
word = "Santa"
newWord = ""
for char in word:
    newWord = newWord + char
print (newWord)
```

```python
word = "Santa"
newWord = ""
currentChar = 0
while (currentChar < 2):
    newWord = newWord + word[currentChar]
    currentChar = currentChar + 1
print (newWord)
```

In addition to using `for` and `while`, how else can a string be traversed?
Traversing strings

In addition to using for and while, how else can a string be traversed?

Recall the `range` function, in combination with `for`

```python
for x in range(0, 44):
    print(x)
```

What is the output of `range`, and how is `for` and `in` used in combination with `range`'s output?
Traversing strings

Use a for loop in combination with a string and in will iterate over each character in the string

```python
word = "Santa"
newWord = ""
for char in word:
    newWord = newWord + char
print(newWord)
```

```python
word = "Santa"
newWord = ""
currentChar = 0
while (currentChar < 2):
    newWord = newWord + word[currentChar]
    currentChar = currentChar + 1
print(newWord)
```

Q: What is the output of the above code?
Traversing strings

Use a for loop in combination with a string and in will iterate over each character in the String

```python
word = "Santa"
newWord = ""
for char in word:
    newWord = newWord + char
print(newWord)
```

```python
word = "Santa"
newWord = ""
currentChar = 0
while (currentChar < 2):
    newWord = newWord + word[currentChar]
    currentChar = currentChar + 1
print(newWord)
```

The iterator variable `aChar` will be assigned individual characters from “Fred”, one-by-one, until all characters have had their turn
Character Classification

A question that has come up before ...

What if you want to check if a user has input, via the keyboard, a number or a character?

```
anInput = "CSCI141"
```

Write code that checks each character of the variable `anInput`, and prints if it is a letter or an integer character

(on the board exercise)
Character Classification

A question that has come up before …

What if you want to check if a user has input, via the keyboard, a number or a character?

anInput = "CSCI141"

Write code that checks each character of the variable anInput, and prints if it is a letter or an integer character

```python
for char in anInput:
    if (char == "a" or char == "b" or char == "c")...
        print (char + " is a letter")
    if (char == "1" or char == "2" or char == "3")...
        print (char + " is a digit")
    if (char == "A" or char == "B" or char == "C")...
        print (char + " is a letter")
```
Character Classification

A question that has come up before ...

What if you want to check if a user has input, via the keyboard, a number or a character?

```python
anInput = "CSCI141"

Write code that checks each character of the variable `anInput`, and prints if it is a letter or an integer character.

```for char in anInput:
    if (char == "a" or char == "b" or char == "c") ...:
        print (char + " is a letter")
    if (char == "1" or char == "2" or char == "3") ...:
        print (char + " is a digit")
    if (char == "A" or char == "B" or char == "C") ...:
        print (char + " is a letter")
```

There Must be a better way...
Character Classification

A question that has come up before ...

What if you want to check if a user has input, via the keyboard, a number or a character?

anInput = "CSCI141"

Write code that checks each character of the variable anInput, and prints if it is a letter or an integer character

string.ascii_lowercase is a string that contains all the lower case letters
string.ascii_uppercase is a string that contains all upper case letters
string.digits is a string that contains all the digits

Q: How does this help? (hint : this week’s lab .. in and not in)
Character Classification

A question that has come up before ...

What if you want to check if a user has input, via the keyboard, a number or a character?

anInput = "CSCI141"

Write code that checks each character of the variable anInput, and prints if it is a letter or an integer character

```python
import string

for char in anInput:
    if (char in string.ascii_lowercase):
        print (char + " is lower case letter")
```
Take home exercise

We can now more easily solve the most recent take home exercise ...

Task: Write a function, compareTwoStrings, that has two parameters. You can assume that when invoked, the compareTwoStrings function’s both argument are of type String containing only letters.

The function should determine if among the two arguments there are none, a single, or more then a single upper case letter.

- **If no upper case letters**, the function should return a concatenation of both arguments, except all upper case
- **If a single upper case letter**, the function should return a concatenation of both arguments, except all (original) lower case letters are printed upper case, and the single upper case letter is lower case.
- **If more than a single upper case letter** is detected, the function should output the string “ERROR”

Live demo
Recall that Strings and Lists are “different” than other data types, because they are made up of “smaller” pieces.

Q: How are lists different from Strings?
A list is a collection of items, and each item is an element of the list. Just like Strings, each element has an index.
A list is a collection of items, and each item is an element of the list. Just like Strings, each element has an index.

Unlike strings, elements in a list can be of different types

```
[10, 20, 30, 40]
["spam", "bungee", "CS"]
["hello", 2.0, 5, [10, 20]]
```

What do these lists “look” like?
Lists

A list is a collection of items, and each item is an element of the list. Just like Strings, each element has an index.

Unlike strings, elements in a list can be of different types.

This is a list made up of 4 elements, all of them are integers.

```
[10, 20, 30, 40]
["spam", "bungee", "CS"]
["hello", 2.0, 5, [10, 20]]
```
A list is a collection of items, and each item is an element of the list. Just like Strings, each element has an index.

Unlike strings, elements in a list can be of different types:

```
[10, 20, 30, 40]
["spam", "bungee", "CS"]
["hello", 2.0, 5, [10, 20]]
```

This is a list made up of 3 elements, all of them are Strings:

```
“spam”  “bungee”  “CS”
```
Lists

A list is a collection of items, and each item is an element of the list. Just like Strings, each element has an index.

Unlike strings, elements in a list can be of different types

Q: What is this list made up of?

```
[10, 20, 30, 40]
["spam", "bungee", "CS"]
["hello", 2.0, 5, [10, 20]]
```
Lists

A list is a collection of items, and each item is an element of the list. Just like Strings, each element has an index.

Unlike strings, elements in a list can be of different types.

This is a list made up of 4 elements, of different types, and the fourth elements is ANOTHER list:

```
[10, 20, 30, 40]
["spam", "bungee", "CS"]
["hello", 2.0, 5, [10, 20]]
```

```
<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;hello&quot;</td>
<td>2.0</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>20</td>
<td></td>
</tr>
</tbody>
</table>
```
Lists

To create a list, you create a variable name, use the assignment operator, and then specify a list (we’ve already seen this), using the [ ] brackets.

```
vocabulary = ["iteration", "selection", "control"]
```

Task: Draw a diagram to show the variable `vocabulary`, a list with 3 entries, and the strings `iteration`, `selection`, and `control`.

Live demo
(you will be asked to draw a similar diagram on the final exam)
Lists

To create a list, you create a variable name, use the assignment operator, and then specify a list (we’ve already seen this), using the [ ] brackets.

```python
vocabulary = ['iteration', 'selection', 'control']
```

The len function returns the number of elements in the list
Accessing elements is done the same way as accessing characters in a string

Write code that prints the items of the list `vocabulary`, one-by-one

Live demo
To create a list, you create a variable name, use the assignment operator, and then specify a list (we’ve already seen this), using the [ ] brackets.

```python
vocabulary = ["iteration", "selection", "control"]
```

The `len` function returns the number of elements in the list.

Accessing elements is done the same way as accessing characters in a string.

Write code that prints the items of the list `vocabulary`, one-by-one.

```
for item in vocabulary:
    print(item)

for x in range(0, len(vocabulary)):
    print(vocabulary[x])
```

Live demo
Unlike strings, lists are mutable. What does this mean?
Unlike strings, lists are mutable. What does this mean?

```python
myList = ['1', 3, '34']
myList.append('43')
for item in myList:
    print(item)
```

The `append` function is used to add an item to the list. What does the above code accomplish?

(on the board explanation and walk through)
Up Next

Lists and Files