CS 1112: Introduction To Programming

Nested Data Structures

Dr. Nada Basit // basit[*at*]Virginia[*dot*]edu

Spring 2024

Friendly Reminders

- Your ***safety*** and ***comfort*** is important!
  - If you choose to wear a mask you are welcome to do so
  - *We will interpret wearing a mask as being **considerate and caring** of others in the classroom (not that you are sick), and realize that some may choose to mask to **remain distanced***
- Be an ***active*** participant in your learning!
  You're welcome and ***encouraged*** to ask questions during class!
- If you feel ***unwell***, or think you are, please stay home
  - *We will work with you!*
  - Get some rest ☺
  - View the recorded lectures – *please allow 24-48 hours to post*
  - *Contact us!*

Announcements

- **PA05** is due by 11:00pm on 3/27 (***Tonight***)!
- New Quiz out this Friday…
- **PA06** is due by 11:00pm on 4/5 (Friday -- ***Note the day***)

Coming up…

- **Exam 2**: Monday, April 8, 2024 *(SDAC accommodations? Book time slot on April 8!)*
  - *In-class; exam on Sherlock (like last time)*
  - *Closed-book/closed-notes/closed-PyCharm/closed-everything!*
  - *Duration: 1 hour and 15 minutes (like last time)*

Quick Aside…

Splitting a String

- Strings have a method named split that divides the string into multiple pieces.
- By default, split divides strings based on whitespace.

```
str1 = 'Have a nice day'
list1 = str1.split()
print('type:', type(list1))
type: <class 'list'>
print('value:', list1)
value: ['Have', 'a', 'nice', 'day’]
```

- Notice that split returns a list of strings, the *original string has not been modified*.
- We can pass the split function an argument to specify a different **separator**.

```
color_str = 'red,orange,yellow,green,blue,purple'
color_list = color_str.split(',')
print(color_list)
['red', 'orange', 'yellow', 'green', 'blue', 'purple']
```

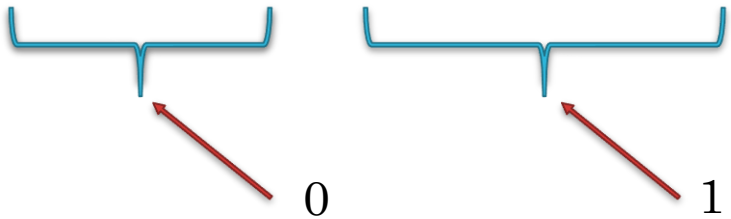
Multi-Assignment and Unpacking

```
x,y = 4,7
print(y) # 7

numbers = [1,2,3,4,5]
a,b,c,d,e = numbers
print(c)  # 3


def get_words():
    return 'desk', 'chair'
first_word, second_word = get_words()
print(second_word)  # chair
```

Nested Structures

Nested Lists

- Sometimes referred to as "2-dimensional lists" or "**list of lists**"
- Example: `my_list = [[8, 6, 7], [5, 3, 0, 9]]`



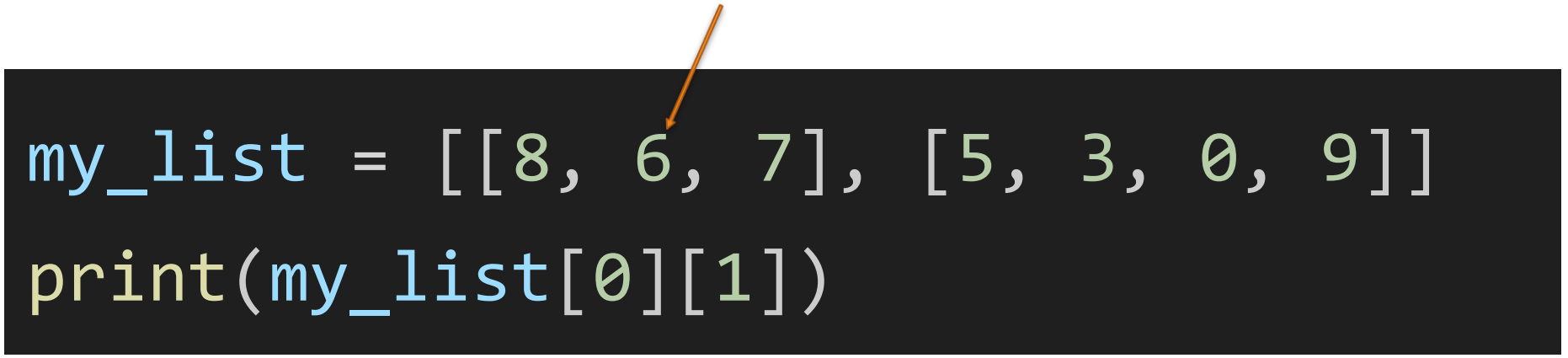
- To access a **sublist**:
```
my_list[0] # → [8, 6, 7]
```

- To access a **single element** in a sublist:
```
my_list[1][3] # → 9
```

Q1: What is printed?

```
my_list = [[8, 6, 7], [5, 3, 0, 9]]
print(my_list[0][1])
```

What is printed?



```
my_list = [[8, 6, 7], [5, 3, 0, 9]]
print(my_list[0][1])
```

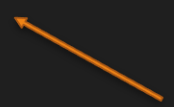
6

Q2: What is printed?

```
color_str = 'red,orange,yellow,green,blue,purple'
color_list = color_str.split()
print(color_list[-1])
```

What is printed?

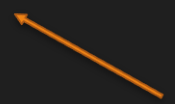
```
color_str = 'red,orange,yellow,green,blue,purple'
color_list = color_str.split()
print(color_list[-1])
```



red,orange,yellow,green,blue,purple

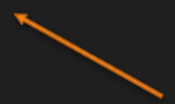
Q3: What is printed?

```
color_str = 'red,orange,yellow,green,blue,purple'
color_list = color_str.split(',')
print(color_list[-1])
```



What is printed?

```
color_str = 'red,orange,yellow,green,blue,purple'
color_list = color_str.split(',')
print(color_list[-1])
```



purple

What is printed?

```
color_str = 'red,orange,yellow,green,blue,purple'
color_list = color_str.split(',')
print(color_list[-1])   # Get the last element
```

purple

Let's Look At…

Nested structures including dictionaries, loops, tuples, and conditionals…

Python Demonstration

Let's jump on PyCharm!

nested_structures.py
nested_loops.py
nested_conditionals.py
nested_data_structures.py

```python
# Sample code

# Nested List Examples

# this is a nested list
lst = [[8, 6, 7], [5, 3, 0, 9]]

print(len(lst))  # this is only 2, since there are two elements in lst
                 # these elements just happen to *be* lists

print(lst[0])  # prints [8, 6, 7]
print(lst[1][3])  # prints 9, with is the element at index 3 of the list
of index 1

lst.append([5, 4, 3, 2, 1])  # append [5, 4, 3, 2, 1] to the list of
lists
print(lst)

#Loop through each item and print it
for i in lst:
   for j in i:
       print("sublist", i, "element", j)
```

```python
# Sample code

# Structured nested list

# My Cats
# Each sublist is [name, color, age, is_floofy]
my_cats = [["Stewart", "orange", 9, False],
           ["Colbert", "gray", 6, True],
           ["Morgan", "tuxedo", 9, False],
           ["Chloe", "calico", 0.5, False]]

# print all the cats names:
for cat in my_cats:
   print(cat[0])

# get the average age of my cats:
sum_age = 0
for cat in my_cats:
   sum_age += cat[2]
print(sum_age / len(my_cats))

# print all the information of only the floofy cats:
for cat in my_cats:
   if cat[3]:
       print(cat)
```

Review this code on your own

```
id1 = 142
animal1 = 'gorilla'
name1 = 'pete'

id2 = 253
animal2 = 'zebra'
name2 = 'marty'

zoo_list = [[142,'gorilla','pete'],[253,'zebra','marty']]
zoo_dictionary = {142:['gorilla','pete'],253:['zebra','marty']}
# These are 3 ways to loop through dictionaries
# .keys()  # uses this if we don't specify
# .values()
# .items()

for each in zoo_dictionary:
  animal_list = zoo_dictionary[each]
  print(animal_list[0])

# we can get both the key and the value this way
for k,v in zoo_dictionary.items():
  print(k)
  print(v)
```

```
str1 = 'Have a nice day'
list1 = str1.split()
print('type:',type(list1), ', value:',list1)

str1 = 'Have a nice day'
list1 = str1.split()
print('type:', type(list1))
print('value:', list1)

color_str = 'red,orange,yellow,green,blue,purple'
color_list = color_str.split(',')
print(color_list)
```

Review this code on your own

```
# Sample code - using a list of lists

# Create a list of lists:
state_list = []
for line in table.split("\n"):  # for each line
   #split on commas:
   split_line = line.split(",")  # this is now [Name, Postal,
Population]
   split_line[2] = int(split_line[2])  # converts the population to a
number
   state_list.append(split_line)

#Example ways to retrieve information
print(state_list)  # Print all the states
print(state_list[46])  # Print Virginia, I can't wait to ...
print(state_list[46][1]) # Print Virginia's Postal Code

# print the name all states with a population under 1 million
for state in state_list:
   if state[2] < 1000000:
       print(state[0], state[1])
```

```
# Dictionary of lists

state_dictionary = {}

for line in table.split("\n"):  # for each line
   #split on commas:
   split_line = line.split(",")  # this is now [Name, Postal, Population]
   population = int(split_line[2])  # converts the population to a number
   state_name = split_line[0]
   state_postal_code = split_line[1]
   state_dictionary[state_name] = [state_postal_code, population]

print(state_dictionary)  # Print all the states
print(state_dictionary["Virginia"])  # Print Virginia
print(state_dictionary["Virginia"][0])  # Print Virginia's Postal Code

# print the name all states with a population under 1 million
for state_name in state_dictionary:
   if state_dictionary[state_name][1] < 1000000:
       print(state_name, state_dictionary["Virginia"][0])
```

Review this code on your own

Reminder: CS Laptop Loaner Program

- This course requires students to have a **laptop**
- I realize that not everybody might have one (nor necessarily need one for their desired major / path…)
- If you do not have a laptop for any reason… ***not to worry!***
- The CS department's Systems staff has a notebook / laptop loaner program and will be able to loan you a notebook / laptop computer for the duration of the semester if you don't have one or if you cannot afford one.
  - Also available if your laptop is broken and under repair, we can arrange for you to receive a loaner laptop for a week or two until your own laptop is fixed

Interested? Link: https://www.cs.virginia.edu/wiki/doku.php?id=cs_laptop_loaner

I am happy to be your sponsor. Please let me know.