

More on Variables Strings Data Types

**CS 8: Introduction to Computer Science, Spring 2019
Lecture #3**

Ziad Matni, Ph.D.
Dept. of Computer Science, UCSB

Administrative

- **Hw01 – due today**
- Hw02 – due next week
- Lab01 – due on Sunday by midnight (11:59 pm) on
Gradescope!
- Python IDLE
- Gradescope invites
- Linux workshop

Linux Workshop ?

- Gauging interest in this?

Prof. Matni's office hours:
Mon. 1 – 3 pm
SSMS 4409

4/9/19

CS 8 - CLOSED LABS - PHELPS 3525						
Day of Week	Start Time	End Time	TA On Duty	Mentors on Duty	Mentors on Duty	Mentors On Duty
MONDAY	8:00 AM	8:50 AM	Chong Liu	Yichen Shao	Jacqueline Mai	Daniel Shu
	9:00 AM	9:50 AM	Chong Liu	Yichen Shao	Jacqueline Mai	Daniel Shu
	10:00 AM	10:50 AM	Shane Masuda	Zhao Siqi	Jose Cuellar	
	11:00 PM	11:50 AM	Shane Masuda	Zhao Siqi	Jose Cuellar	
CS8 OPEN LABS (i.e. Office Hours) - PHELPS 3525						
Day of Week	Start Time	End Time	TA On Duty	Mentors on Duty	Mentors on Duty	Mentors On Duty
MONDAY	5:00 PM	5:30 PM	Anacaren Ruiz	Jacqueline Mai		
	5:30 PM	6:00 PM	Anacaren Ruiz	Jacqueline Mai		
	6:00 PM	6:30 PM		Jose Cuellar		
	6:30 PM	7:00 PM		Jose Cuellar		
	7:00 PM	7:30 PM		Jose Cuellar		
	7:30 PM	8:00 PM		Jose Cuellar		
	8:00 PM	8:30 PM		Zhao Siqi		
	8:30 PM	9:00 PM		Zhao Siqi		
TUESDAY	7:00 PM	7:30 PM	Anacaren Ruiz	Zhao Siqi	Daniel Shu	
	7:30 PM	8:00 PM	Anacaren Ruiz	Zhao Siqi	Daniel Shu	
	8:00 PM	8:30 PM		Zhao Siqi	Jacqueline Mai	
	8:30 PM	9:00 PM		Zhao Siqi	Jacqueline Mai	
WEDNESDAY	7:00 PM	7:30 PM	Shane Masuda	Yichen Shao	Jose Cuellar	
	7:30 PM	8:00 PM	Shane Masuda	Yichen Shao	Jose Cuellar	
	8:00 PM	8:30 PM	Shane Masuda	Yichen Shao		
	8:30 PM	9:00 PM	Shane Masuda	Yichen Shao		
THURSDAY	7:00 PM	7:30 PM	Chong Liu	Yichen Shao	Daniel Shu	
	7:30 PM	8:00 PM	Chong Liu	Yichen Shao	Daniel Shu	
	8:00 PM	8:30 PM	Chong Liu		Daniel Shu	
	8:30 PM	9:00 PM	Chong Liu		Daniel Shu	
FRIDAY	2:00 PM	2:50 PM		Jacqueline Mai		
	3:00 PM	3:50 PM		Jacqueline Mai		
	4:00 PM	4:50 PM				

Homework Etiquette

- Print the **PDF** for the homework **double sided**.
- Use **dark ink**.
- Write your name **CLEARLY**.
- Do **not staple** your homework.
- Write your name **on each page**.
- **Do not fold, cut or rip your assignment**.
- Keep the homework **stack neat**.

Lecture Outline

- Variables in Python
- Strings & Operations on Strings
- Intro to Lists & Tuple

Yellow Band = Class Demonstration! ☺

All Data in Python Has a *type()*

- But you can change its type
 - *Implicitly*, example: `x = 5` at first and then making `x = "hello"`
 - *Explicitly*, by forcing the type
- Introducing the built-in function ***type()***

More on “Type Casting”

- Let's try these out on IDLE and explain them:

```
>>> int(4.2)
>>> int (True)
>>> int (False)
>>> float(False)
>>> float ( true )
>>> float (4) / 5
>>> str ( 42 )
>>> int ("42")
```

Variable Names in Python

3 simple rules for choosing names:

- Can ONLY have **letters**, **digits**,
and **_** (underscores)
- Must NOT *begin with* a digit or non-alphabet character
(except underscore)
- Cannot use Python reserved **keywords**
 - Example: **def**, **int**, **False**, **True**, **print**, etc...

UserName ☺
Age1
Age2
_Deviation

2Good2BTrue ☹
\$\$MaMoney!!
<0_0>
#YOLO

Variable Names in Python: Other Conventions

- Choose brief, but *meaningful* names
- Most programmers prefer lower case use (Example: **total** vs. **TOTAL**)
- Use either “**camel case**” or **underscore** to separate words
 - Camel Case is using capital letters to separate words, like **NumOfCats**
 - Underscoring is using underscores to separate words, like **num_of_cats**
 - **Be consistent**: use one or the other throughout your program
- All the above applies to function names, module names, etc...

Objects

- An *object* in Python is anything that has:
 - (1) an identity or name
 - (2) a type
 - (3) a value
- Example: `pi = 3.14159`
 - Name: `pi`
 - Type: floating point
 - Value: `3.14159`

Demo

Let's try this code out – what do you think it'll do?

```
pi = 3.14159  
radian_angle = 0.7853975  
degree_angle = radian_angle*180/pi  
print(degree_angle)
```

Let's try it out!

The Equals Operators

=

Assign it to

==

Is it equal to?

!=

Is it not equal to?

The Equals Operators

bebe = 22

The *int* variable **bebe** now has the value **22** (i.e. it is *assigned* the value 22)

bebe == 22

A statement that has a Boolean answer:

Is **bebe** equal to **22**? The answer is yes, in other words, the answer is Boolean **True**.

*Note that **bebe** is not changed – it's still 22*

bebe != 22

Is **bebe** not equal to 22?

The answer is **False**. Again, **bebe** is unchanged.

Let's try it out!

Assignment vs. Comparison

What happens when I do this?

	<i>a</i>	<i>b</i>
$a = 1.1$	1.1	
$b = a$	1.1	1.1
$a = 2.2$	2.2	1.1

Assignment vs. Comparison

What happens when I do this?

	<i>z</i>	<i>y</i>	<i>x</i>
$z = 6$	6		
$y = 8$	6	8	
$z = z + y$	14	8	
$x = (z == 14)$	14	8	True

Assignment vs. Comparison

What happens when I do this?

	<i>m</i>	<i>n</i>
$m = 42$	42	
$n = 2 - m$	42	-40
$n == 0$	42	-40
$m = (m == 40)$	False	-40

Input and Output

- We'll make use of 2 built-in functions in Python:
 - **print()** to print out to the screen
(called *standard output*)
 - **input()** to get input from the keyboard
(called *standard input*)

Input and Output

- To output data, use `print()`
`print("Hello all you happy people!")`

- To get data and put it in a variable, use `input()`

`name = input()`

OR

`name = input("Name. Give it. Now: ")`

Let's try it out!

Strings

- Collection of *characters*
- A string literal is enclosed in quotes
 - Use either double-quotes ("") or single quotes ('')

Examples:

```
name = "#JimboJones@UCSB? Wow!"  
nombre = 'Lisa Simpson'
```

Special Characters in Strings

- What would you do if you wanted a string to be:
I said "hello!"
- Answer: use the special character indicator \
 - The back-slash

Example:

```
message = "I said \"hello!\""
```

Demo!

Strings as Objects

- Strings are **objects** of a Python *class* named **str**
- Lots of built-in functions work for string *objects*
- **Class** = an general “blueprint”
- **Object** = a particular “instant” of a class

Operations on Strings

- **Concatenation**
 - Merging multiple strings into 1
 - Use the `+` operator
 - `"say my" + " " + "name"` will become `"say my name"`
- **Repetition**
 - Easy way to multiply the contents of a string
 - Use the `*` operator
 - `"ja " * 3` is `"ja ja ja "` (*why is there a space at the end?*)

Demo!

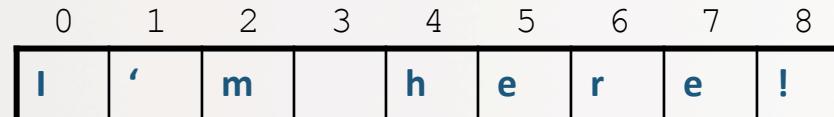
Indexing

- Every character in a string has an index associated with it

0	1	2	3	4	5	6	7	8
I	'	m		h	e	r	e	!

- In Python, indexing always starts at **0**.
 - So the 1st character in the string is character #0
 - Indexing is called out with square brackets [n]

Indexing



- If `name = "I'm here!"` then:

`name[0] = "I"`

`name[3] = " "`

`name[5] = "e"`

`name[15]` is undefined (error)

YOUR TO-DOS

- Finish reading **Chapter 2**
- Start reading **Chapter 3**
- Start on **HW2** (due next **Tuesday**)
- Finish up **Lab1**

- Remember office hours! ☺

- Embrace randomness

</LECTURE>