



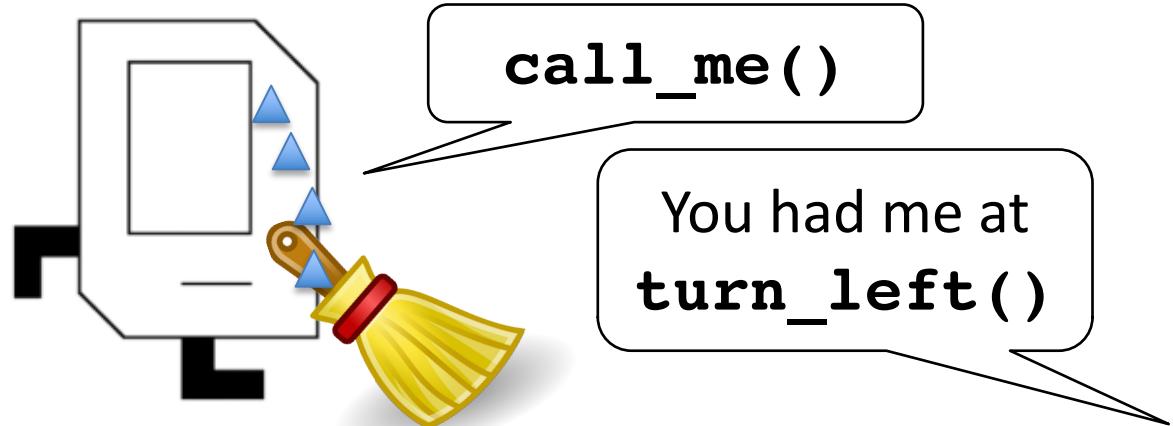
Introduction to Python

Chris Gregg

CS106A, Stanford University

Based on slides by Chris Piech and Mehran
Sahami

Housekeeping



- Class website: <http://cs106a.stanford.edu>
- Sections
 - Started this week – hopefully you enjoyed your first section!
- LaIR is now open. See class webpage for details
- Bye bye, Karel!



More on Programming Style

....

File: SteepleChaseKarel.py

Karel runs a steeple chase that is 9 avenues long.
Hurdles are of arbitrary height and placement.

....

To run a race that is 9 avenues long, we need to move forward or jump hurdles 8 times.

....

```
def main():
    for i in range(8):
        if front_is_clear():
            move()
        else:
            jump_hurdle()
```

Consistent
indentation

Comments for program
and *every* function

Decomposition principle:
Each function should solve
one step of problem

Pre-condition: Facing East at bottom of hurdle

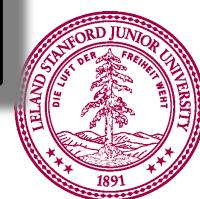
Post-condition: Facing East at bottom in next avenue after hurdle

....

```
def jump_hurdle():
    ascend_hurdle()
    move()
    descend_hurdle()
```

Short functions
(usually 1-15 lines)

Descriptive *names*
(snake_case)

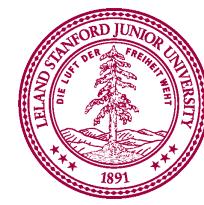
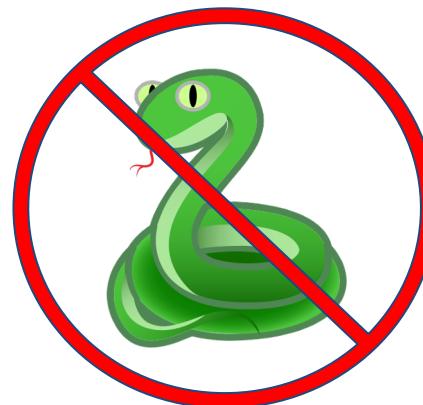


Welcome to Python

Guido van Rossum
(Creator of Python)

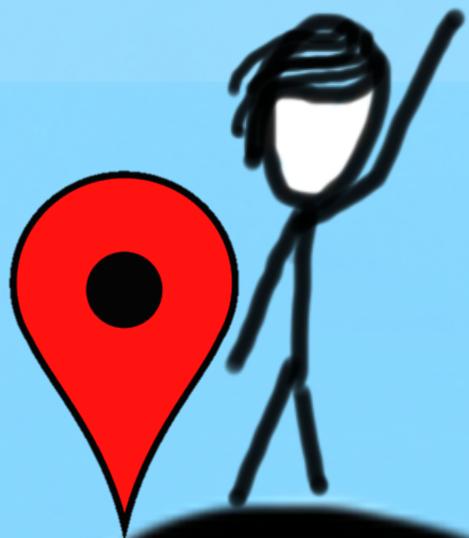


Monty Python's Flying Circus



Today's Goal

1. Introduction to Python
2. Understanding variables



Our First Python Program

```
"""
File: helloworld.py
-----
This is our first python program. It is customary to
have a programmer's first program write "hello world"
(inspired by the first program in Brian Kernighan and
Dennis Ritchie's classic book, 'The C Programming Language.')
"""
```

```
def main():
    print("hello, world!")
```

Our First Python Program

The screenshot shows the PyCharm IDE interface with the following details:

- File Menu:** File, Edit, View, Navigate, Code, Refactor, Run, Tools, VCS, Window, Help.
- Title Bar:** Lecture4-Python [...]\Lecture4-Python - ...\.helloworld.py
- Toolbar:** intro, Run, Stop, Refresh, Search.
- Project Explorer:** 1: Project, Lecture4-Python [pycharm_intro] (contains add2numbers.py, helloworld.py), External Libraries, Scratches and Consoles.
- Code Editor:** The file helloworld.py is open, showing the following code:

```
1     """
2     File: helloworld.py
3     -----
4     This is our first python program. It is customary to
5     have a programmer's first program write "hello world"
6     (inspired by the first program in Brian Kernighan and
7     Dennis Ritchie's classic book, 'The C Programming Language.')
8     """
9
10    def main():
11        print("hello, world!")
12
13
14    # This provided line is required at the end of a Python file
15    # to call the main() function.
16    if __name__ == '__main__':
17        main()
```
- Toolbars:** I: Structure, Favorites.
- Bottom Navigation:** R Console, Find, TODO, Terminal (highlighted with a red box), Python Console, Event Log.
- Status Bar:** 10:1, CRLF, UTF-8, 4 spaces, Python 3.8, Event Log icon.

Our First Python Program

The screenshot shows the PyCharm IDE interface with the following details:

- File Menu:** File, Edit, View, Navigate, Code, Refactor, Run, Tools, VCS, Window, Help.
- Title Bar:** Lecture4-Python [...]\\Lecture4-Python - ...\\helloworld.py
- Toolbar:** intro, Run, Stop, Refresh, Search.
- Project Explorer (Left):** 1: Project, Lecture4-Python [pycharm_intro] (expanded), add2numbers.py, helloworld.py, External Libraries, Scratches and Consoles.
- Code Editor (Center):** helloworld.py (selected). The code is:

```
"""
File: helloworld.py
-----
This is our first python program. It is customary to
have a programmer's first program write "hello world"
(inspired by the first program in Brian Kernighan and
Dennis Ritchie's classic book, 'The C Programming Language.')

"""

def main():
    print("hello, world!")

# This provided line is required at the end of a Python file
# to call the main() function.
if __name__ == '__main__':
    main()
```
- Terminal (Bottom):** Local (selected). Output:

```
Microsoft Windows [Version 10.0.18363.720]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\\Text\\Teaching\\CS106A\\CS106A-Spr19-20\\Lectures\\Lecture4\\Lecture4-Python>
```
- Bottom Navigation:** R Console, Find, TODO, Terminal (selected), Python Console, Event Log.
- Status Bar:** 10:1, CRLF, UTF-8, 4 spaces, Python 3.8.

Our First Python Program

The screenshot shows the PyCharm IDE interface. The top menu bar includes File, Edit, View, Navigate, Code, Refactor, Run, Tools, VCS, Window, Help, and Lecture4-Python [...\\Lecture4-Python] - ...\\helloworld.py. The left sidebar has sections for Project (1: Project), Favorites (2: Favorites), and Structure (3: Structure). The main editor window displays the contents of helloworld.py:

```
1  """
2  File: helloworld.py
3  -----
4  This is our first python program. It is customary to
5  have a programmer's first program write "hello world"
6  (inspired by the first program in Brian Kernighan and
7  Dennis Ritchie's classic book, 'The C Programming Language.')
8  """
9
10 def main():
11     print("hello, world!")
12
13
14 # This provided line is required at the end of a Python file
15 # to call the main() function.
16 if __name__ == '__main__':
17     main()
```

The terminal below shows the output of running the program:

```
Terminal: Local +  
Microsoft Windows [Version 10.0.18363.720]  
(c) 2019 Microsoft Corporation. All rights reserved.  
  
C:\Text\Teaching\CS106A\CS106A-Spr19-20\Lectures\Lecture4\Lecture4-Python>py helloworld.py
```

This is on a PC.
On Macs: **python3 helloworld.py**

The bottom navigation bar includes R Console, Find, TODO, Terminal (selected), Python Console, Event Log, and status indicators for 10:1 CRLF, UTF-8, 4 spaces, Python 3.8, and user profile.

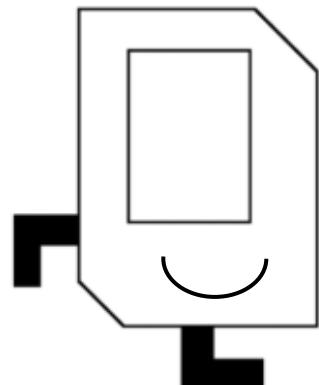
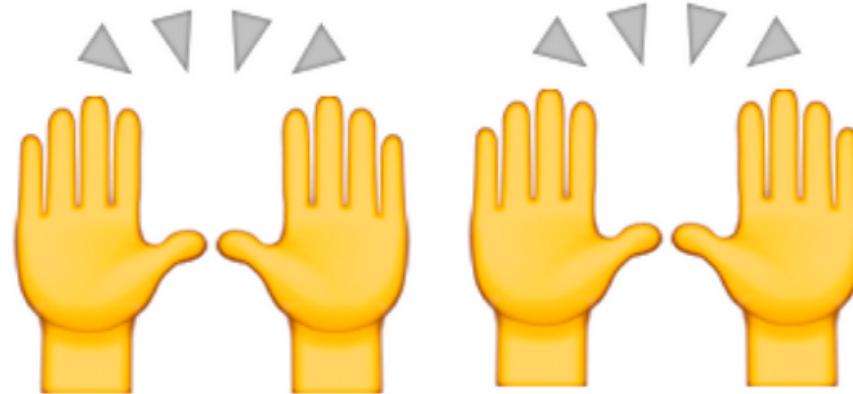
Our First Python Program

The screenshot shows the PyCharm IDE interface with the following details:

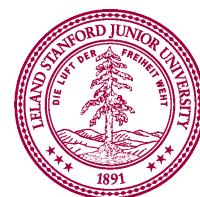
- File Menu:** File, Edit, View, Navigate, Code, Refactor, Run, Tools, VCS, Window, Help.
- Title Bar:** Lecture4-Python [...]\\Lecture4-Python - ...\\helloworld.py
- Toolbar:** intro, Run, Stop, Refresh, Search.
- Project Explorer (1: Project):** Shows the project structure with files: Lecture4-Python [pycharm_intro] (containing add2numbers.py and helloworld.py), External Libraries, and Scratches and Consoles.
- Code Editor:** Displays the content of helloworld.py:

```
1     """
2     File: helloworld.py
3     -----
4     This is our first python program. It is customary to
5     have a programmer's first program write "hello world"
6     (inspired by the first program in Brian Kernighan and
7     Dennis Ritchie's classic book, 'The C Programming Language.')
8     """
9
10    def main():
11        print("hello, world!")
12
13
14
15    # This provided line is required at the end of a Python file
16    # to call the main() function.
17    if __name__ == '__main__':
18        main()
```
- Terminal:** Local (selected), Microsoft Windows [Version 10.0.18363.720], (c) 2019 Microsoft Corporation. All rights reserved.
- Output Terminal:** C:\\Text\\Teaching\\CS106A\\CS106A-Spr19-20\\Lectures\\Lecture4\\Lecture4-Python>py helloworld.py
hello, world!
- Bottom Navigation:** R Console, Find, TODO, Terminal (selected), Python Console, Event Log.
- Status Bar:** 10:1 CRLF UTF-8 4 spaces Python 3.8.

You're now all Python programmers!

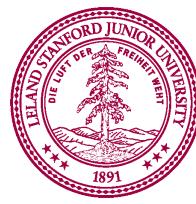


hey_that_looks_
like_what_I_
taught_them()



Another Program

```
def main():
    print("This program adds two numbers.")
    num1 = input("Enter first number: ")
    num1 = int(num1)
    num2 = input("Enter second number: ")
    num2 = int(num2)
    total = num1 + num2
    print(f"The total is {total}.")
```



Another Program

```
def main():
    print("This program adds two numbers.")
    num1 = input("Enter first number: ")
    num1 = int(num1)
    num2 = input("Enter second number: ")
    num2 = int(num2)
    total = num1 + num2
    print(f"The total is {total}.")
```

This program adds two numbers.



Another Program

```
def main():
    print("This program adds two numbers.")
    num1 = input("Enter first number: ")
    num1 = int(num1)
    num2 = input("Enter second number: ")
    num2 = int(num2)
    total = num1 + num2
    print(f"The total is {total}.")
```

This program adds two numbers.
Enter first number:



Another Program

```
def main():
    print("This program adds two numbers.")
    num1 = input("Enter first number: ")
    num1 = int(num1)
    num2 = input("Enter second number: ")
    num2 = int(num2)
    total = num1 + num2
    print(f"The total is {total}.")
```

num1 "9"

This program adds two numbers.

Enter first number: 9



Another Program

```
def main():
    print("This program adds two numbers.")
    num1 = input("Enter first number: ")
    num1 = int(num1)
    num2 = input("Enter second number: ")
    num2 = int(num2)
    total = num1 + num2
    print(f"The total is {total}.")
```

num1

9

This program adds two numbers.

Enter first number: 9



Another Program

```
def main():
    print("This program adds two numbers.")
    num1 = input("Enter first number: ")
    num1 = int(num1)
    num2 = input("Enter second number: ")  
    num2 = int(num2)
    total = num1 + num2
    print(f"The total is {total}.")
```

num1

9

This program adds two numbers.

Enter first number: 9

Enter second number:



Another Program

```
def main():
    print("This program adds two numbers.")
    num1 = input("Enter first number: ")
    num1 = int(num1)
    num2 = input("Enter second number: ")  
    num2 = int(num2)
    total = num1 + num2
    print(f"The total is {total}.")
```

num1

9

num2

"17"

This program adds two numbers.

Enter first number: 9

Enter second number: 17



Another Program

```
def main():
    print("This program adds two numbers.")
    num1 = input("Enter first number: ")
    num1 = int(num1)
    num2 = input("Enter second number: ")
    num2 = int(num2)
    total = num1 + num2
    print(f"The total is {total}.")
```

num1

9

num2

17

This program adds two numbers.

Enter first number: 9

Enter second number: 17



Another Program

```
def main():
    print("This program adds two numbers.")
    num1 = input("Enter first number: ")
    num1 = int(num1)
    num2 = input("Enter second number: ")
    num2 = int(num2)
    total = num1 + num2
    print(f"The total is {total}.")
```

num1

9

num2

17

total

26

This program adds two numbers.

Enter first number: 9

Enter second number: 17



Another Program

```
def main():
    print("This program adds two numbers.")
    num1 = input("Enter first number: ")
    num1 = int(num1)
    num2 = input("Enter second number: ")
    num2 = int(num2)
    total = num1 + num2
    print(f"The total is {total}.")
```

num1

9

num2

17

total

26

This program adds two numbers.

Enter first number: 9

Enter second number: 17

The total is 26.



print function

```
print("This program adds two numbers.")
```

- **print** command prints text to the terminal
- Text printed is between double quotes ("text")
 - Can also be between single quotes ('text')
 - Choice of quotes depends on text you are printing
 - Double quotes when text contains single quotes
`print("no, you didn't") → no, you didn't`
 - Single quotes when text contains double quotes
`print('say "hi" Karel') → say "hi" Karel`



input function

```
num1 = input("Enter first number: ")
```

- **input** command gets text input from the user
- Prints text specified in double/single quotes
 - Then waits for user input
 - Here, user input from **input** is put in a variable (**num1**)
 - The user input is considered text, even if user entered a number
- We'll talk more about **input** function later



What is a Variable?

x 10

- A **variable** is a place to store information in a program
- It associates a **name** with a **value**
- You can create a new variable by **assigning** a value:

x = 10



What is a Variable?

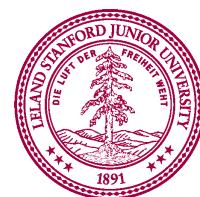


- A **variable** is a place to store information in a program
- It associates a **name** with a **value**
- You can create a new variable by **assigning** a value:

`x = 10`

- The value can change with a new assignment

`x = 5`



What is a Variable?

x 12

- A **variable** is a place to store information in a program
- It associates a **name** with a **value**
- You can create a new variable by assigning a value:

`x = 10`

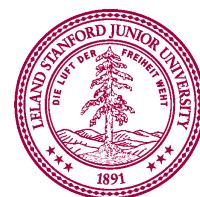
- The value can change with a new assignment

`x = 5`

- You can set the value using mathematical expressions

`x = 5 + 7`

- More about expressions next class



Variable Assignment

- You use the equal sign (=) to assign to a variable
 - The first time you assign a value to a variable, you create it
 - Subsequent assignments give the variable a new value
- Assignment is not the same as "equals" in math
 - Assignment: first evaluate right-hand side, then assign to the variable on the left-hand side
 - Consider the following code:

```
total = 5
total = total + 1
```
- Variables are only visible inside the function in which they are created (called "scope" of variable)
 - If you create a variable in `main()`, its only visible in `main()`
 - More on that next class



Variable Names

- Variable names must:
 - Start with a letter or an underscore (`_`)
 - Contain only letters, digits, or underscores
 - Cannot be a "built in" command in Python (e.g., `for`)
- Variable names are case sensitive
 - **Hello** is not the name as **hello**
- Variable names should:
 - Be descriptive of the value they refer to
 - E.g., `x` is only a good name if it's a coordinate
 - Be in snake case (e.g., `num_students`)



Suitcase Analogy

x 12

- When you store information in a variable, it becomes a Python *object*
 - Objects come in different sizes and types
- Think about a Python object as a suitcase stored in your computer's memory
 - Object take up different amounts of RAM depending on what you're storing.



You have space for millions on suitcases!

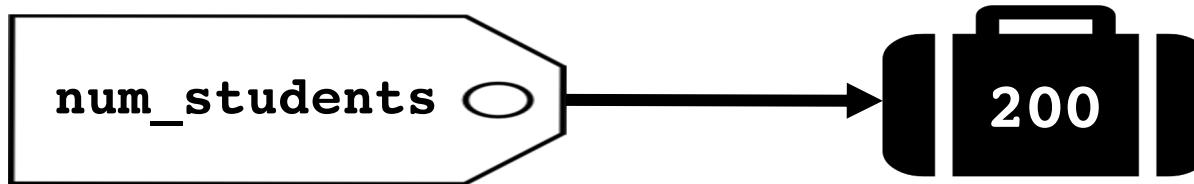


Suitcase Analogy

- Variable is a luggage tag that gives a *name* to suitcase

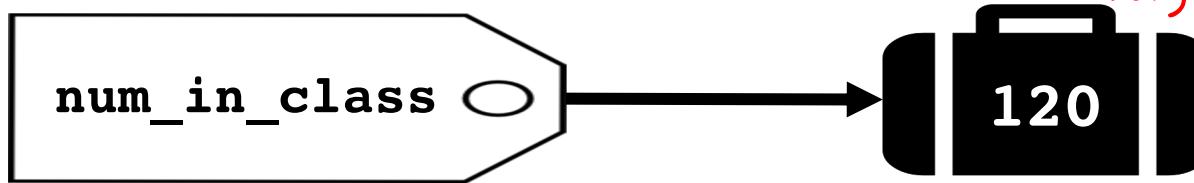
`num_students = 200`

- *Value* is what is stored in the suitcase
- Create the tag/suitcase the first time you assign to variable

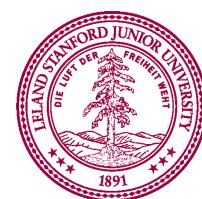
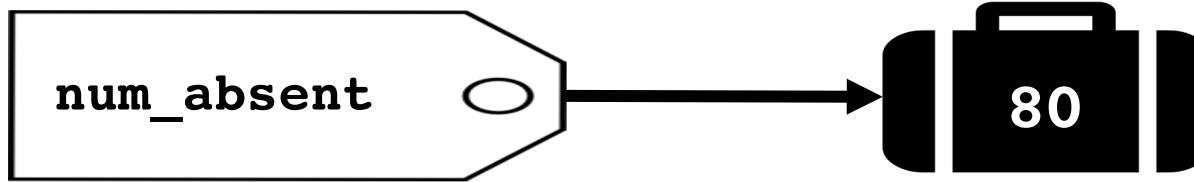


`num_in_class = 120`

Python handles the
baggage for you!



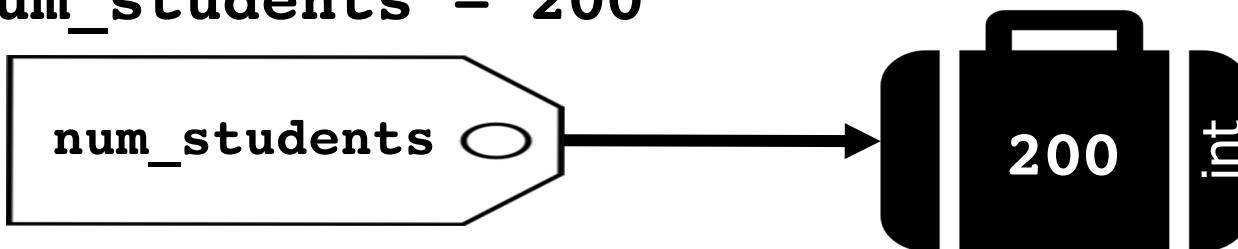
`num_absent = num_students - num_in_class`



Types

- Each suitcase knows what **type** of information it carries

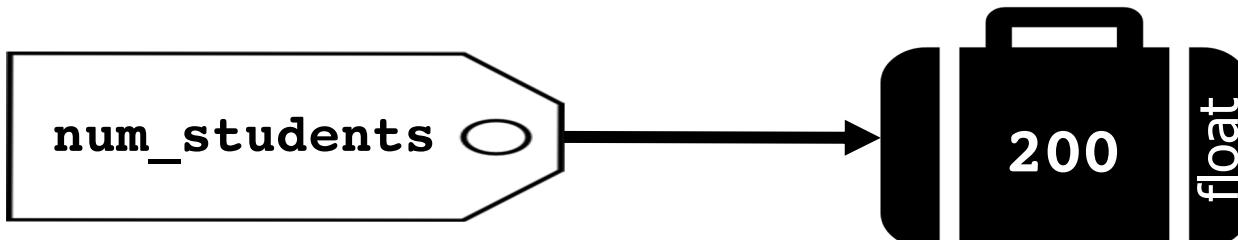
```
num_students = 200
```



- Value stored in suitcase is an integer (called an **int** in Python)
- Suitcase keeps track of **type** of data that is stored there

```
num_students = 200.0      # note decimal point
```

- Now, value stored is a real number (called a **float** in Python)



Some Types in Python

- **int:** integer value (no decimal point)

`x = 10` `y = -2`

- **float:** real number value (has decimal point)

`x = 5.0` `y = -3.7`

- **string:** text characters (between single/double quotes)

`x = "hello"` `y = '10'`

– Note: the string "5" is **not** the same as the integer 5

- **bool:** Boolean logical values (**True/False**)

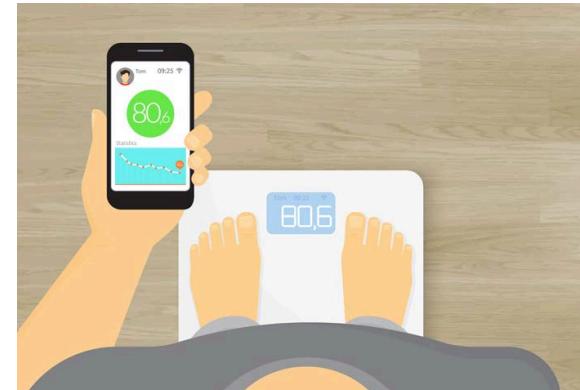
`x = True` `y = False`

- More on strings and bools in a few days



Why Do We Have int and float?

- How much do I weigh?
 - Answer can be a real valued number
 - There is no "next" number
 - This would be a float
- How many children do I have?
 - Answer is an integer
 - There is a well-defined "next" number
 - This would be an int



Recall, Our Program

```
def main():
    print("This program adds two numbers.")
    num1 = input("Enter first number: ")
    num1 = int(num1)
    num2 = input("Enter second number: ")
    num2 = int(num2)
    total = num1 + num2
    print(f"The total is {total}.")
```



Recall, Our Program

```
def main():
    print("This program adds two numbers.")
    num1 = input("Enter first number: ")
    num1 = int(num1)
    num2 = input("Enter second number: ")
    num2 = int(num2)
    total = num1 + num2
    print(f"The total is {total}.")
```

This program adds two numbers.

- **print** command is displaying a **string**



Recall, Our Program

```
def main():
    print("This program adds two numbers.")
    num1 = input("Enter first number: ")
    num1 = int(num1)
    num2 = input("Enter second number: ")
    num2 = int(num2)
    total = num1 + num2
    print(f"The total is {total}.")
```

num1 "9"

This program adds two numbers.
Enter first number: 9

- **input** command gives you back a **string**
 - Even if the user types in a number



Recall, Our Program

```
def main():
    print("This program adds two numbers.")
    num1 = input("Enter first number: ")
    num1 = int(num1)
    num2 = input("Enter second number: ")
    num2 = int(num2)
    total = num1 + num2
    print(f"The total is {total}.")
```

num1

9

This program adds two numbers.

Enter first number: 9

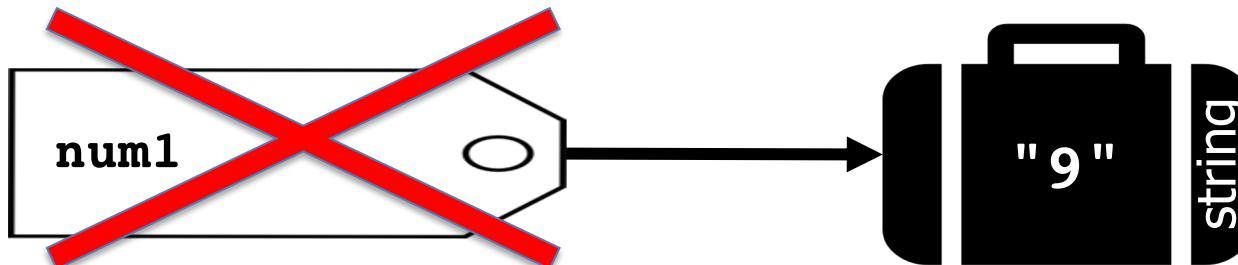
- Create **int** version of **string** and assign it back to **num1**



Show Me The Luggage!

- **input** command gives you back a **string**

```
num1 = input("Enter first number: ")
```

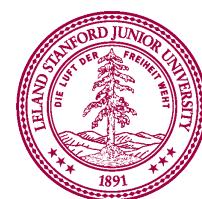
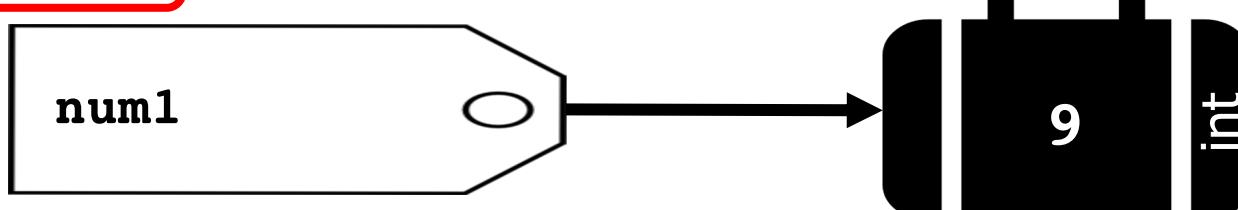


- We create an integer version of **num1**

```
num1 = int(num1)
```

- Create a new suitcase that has **int** version of **num1**
- Then assign the tag num1 to that piece of luggage

```
num1 = int(num1)
```



Recall, Our Program

```
def main():
    print("This program adds two numbers.")
    num1 = input("Enter first number: ")
    num1 = int(num1)
    num2 = input("Enter second number: ")
    num2 = int(num2)
    total = num1 + num2
    print(f"The total is {total}.")
```

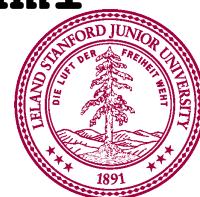
num1

9

This program adds two numbers.

Enter first number: 9

- Create **int** version of **string** and assign it back to **num1**



Recall, Our Program

```
def main():
    print("This program adds two numbers.")
    num1 = input("Enter first number: ")
    num1 = int(num1)
    num2 = input("Enter second number: ") num2 = input("Enter second number: ")
    num2 = int(num2)
    total = num1 + num2
    print(f"The total is {total}.")
```

num1

9

This program adds two numbers.

Enter first number: 9

Enter second number:



Recall, Our Program

```
def main():
    print("This program adds two numbers.")
    num1 = input("Enter first number: ")
    num1 = int(num1)
    num2 = input("Enter second number: ") num2 = input("Enter second number: ")
    num2 = int(num2)
    total = num1 + num2
    print(f"The total is {total}.")
```

num1

9

num2

"17"

This program adds two numbers.

Enter first number: 9

Enter second number: 17



Recall, Our Program

```
def main():
    print("This program adds two numbers.")
    num1 = input("Enter first number: ")
    num1 = int(num1)
    num2 = input("Enter second number: ")
    num2 = int(num2)
    total = num1 + num2
    print(f"The total is {total}.")
```

num1

9

num2

17

This program adds two numbers.

Enter first number: 9

Enter second number: 17



Recall, Our Program

```
def main():
    print("This program adds two numbers.")
    num1 = input("Enter first number: ")
    num1 = int(num1)
    num2 = input("Enter second number: ")
    num2 = int(num2)
    total = num1 + num2
    print(f"The total is {total}.")
```

num1

9

num2

17

total

26

This program adds two numbers.

Enter first number: 9

Enter second number: 17



Recall, Our Program

```
def main():
    print("This program adds two numbers.")
    num1 = input("Enter first number: ")
    num1 = int(num1)
    num2 = input("Enter second number: ")
    num2 = int(num2)
    total = num1 + num2
    print(f"The total is {total}.")
```

num1

9

num2

17

total

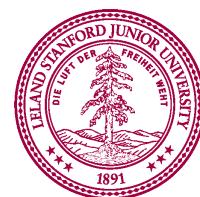
26

This program adds two numbers.

Enter first number: 9

Enter second number: 17

The total is 26.



What's Going on With `print`

- Why is there an `f`, and what is this business with the curly braces?

```
print(f"The total is {total}.")
```
- The `f` denotes a *formatted string*. If you want to print variables inside a string, you need to prepend the double quotes with “`f`”. Variables you want to print are put inside curly-braces.
- This is a new way to print to the terminal in Python – the language has changed! If you’ve programmed in Python before, you might not have seen this method.
- There are at least *three* other ways to print variables, too, but we’re going to use f-strings in CS106A.
- The basic idea is simple: if you want to print a variable, you put the variable inside curly-quotes in a string, and it just gets printed.



Recall, Our Program

```
def main():
    print("This program adds two numbers.")
    num1 = input("Enter first number: ")
    num1 = int(num1)
    num2 = input("Enter second number: ")
    num2 = int(num2)
    total = num1 + num2
    print(f"The total is {total}.")
```

num1

9

num2

17

total

26

This program adds two numbers.

Enter first number: 9

Enter second number: 17

The total is 26.



Side note about `print`

- You can `print` numbers by themselves directly
 - Only need to create and f-string version when printing other text (strings) with them

```
def main():
    x = 10
    y = 3.5
    print(x)
    print(y)
    print(f"x = {x}")
```

```
10
3.5
x = 10
```



You just wrote your first
Python program and learned
about variables!

Today's Goal

1. Introduction to Python
2. Understanding variables



add2numbers.py