Week 1

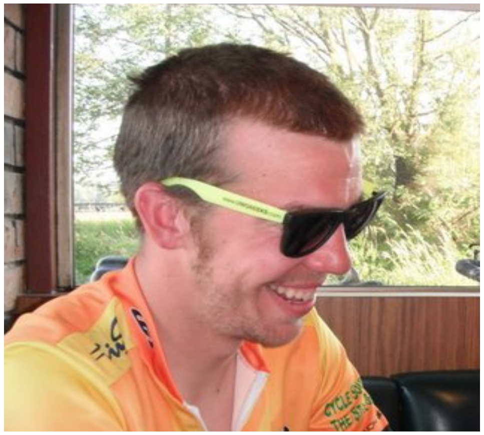
basic Python programs, defining functions

Special thanks to Scott Shawcroft, Ryan Tucker, and Paul Beck for their work on these slides.

About Us

- Stefanie Hatcher
  - Informatics
  - Grandmistress Leet Queen
  - hatchs@cs.washington.edu

- Will Beebe
  - Informatics
  - IPL Hobo
  - nedrager@cs.washington.edu



- Steve Geluso
  - Computer Science
  - Self-proclaimed Moon Mayor
  - geluso@cs.washington.edu

Words you will learn today

```
python
print()
#
def
```

Python!

- Created in 1991 by Guido van Rossum (now at Google)
  - Named for Monty Python
- Useful as a **scripting language**
  - **script**: A small program meant for one-time use
  - Targeted towards small to medium sized projects
- Used by:
  - Google, Yahoo!, Youtube
  - Many Linux distributions
  - Games and apps (e.g. Eve Online)

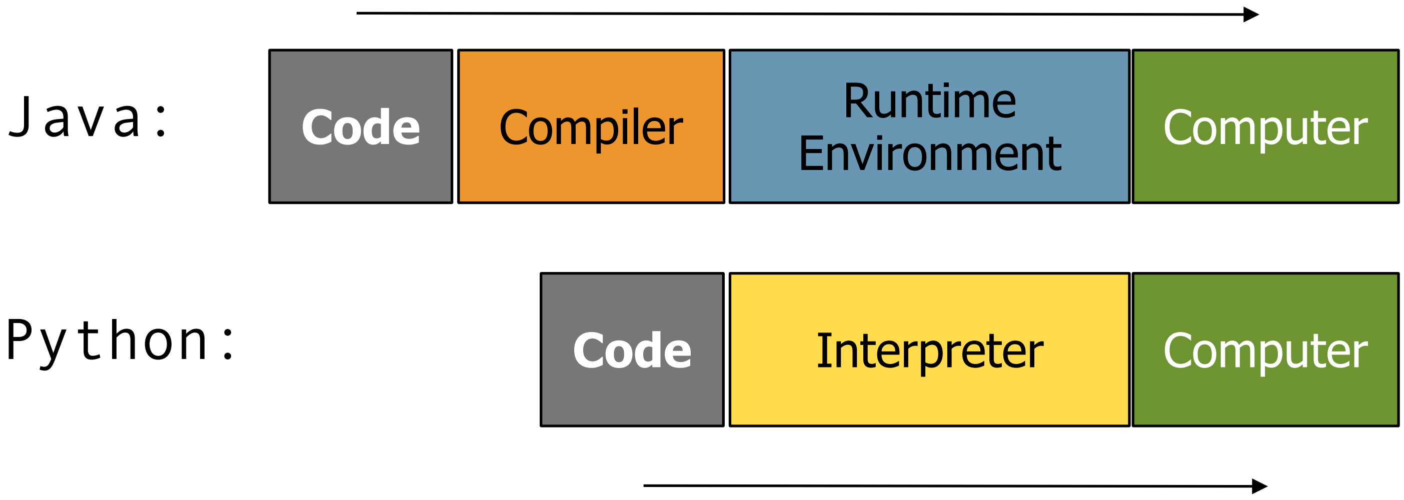
python™

Python is used everywhere!

BitTorrent™
NASA
LUCASFILM Ltd
Google
YouTube
SID MEIER'S CIVILIZATION V
python™

Interpreted Languages

- **Interpreted**
  - Not compiled like Java
  - Code is written and then directly executed by an **interpreter**
  - Type commands into interpreter and see immediate results



Installing Python

Windows:

- Download Python from http://www.python.org
- Install Python.
- Run **Idle** from the Start Menu.

Note: For step by step installation instructions, see the course web site.

Mac OS X:

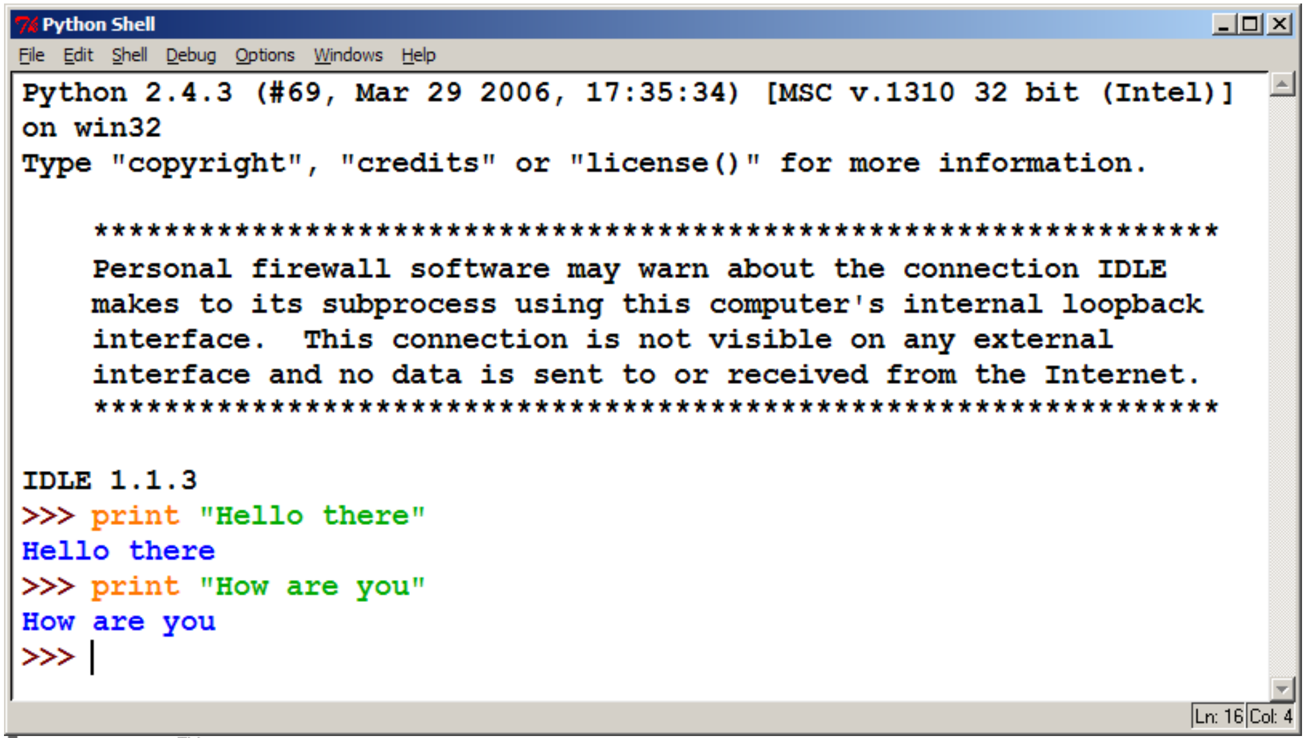
- Python is already installed.
- Open a terminal and run `python` or run Idle from Finder.

Linux:

- Chances are you already have Python installed. To check, run `python` from the terminal.
- If not, install from your distribution's package system.

The Python Interpreter

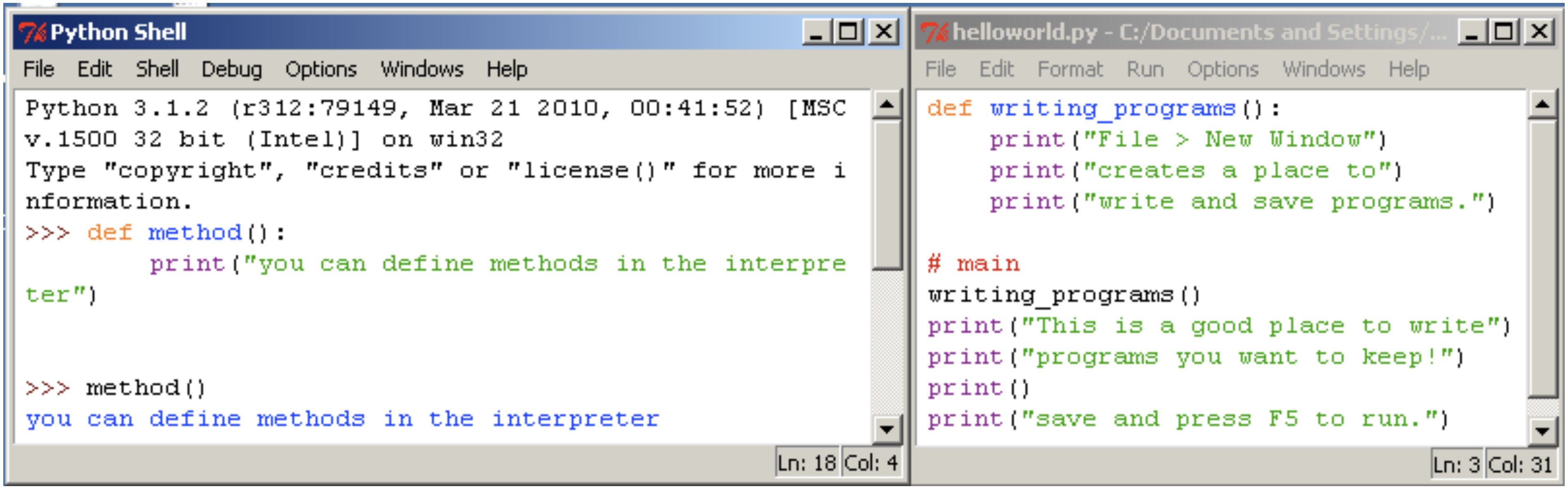
- Allows you to type commands one-at-a-time and see results
- A great way to explore Python's syntax
  - Repeat previous command: *Alt+P*



```
Python 2.4.3 (#69, Mar 29 2006, 17:35:34) [MSC v.1310 32 bit (Intel)]
on win32
Type "copyright", "credits" or "license()" for more information.

    ****************************************************************
    Personal firewall software may warn about the connection IDLE
    makes to its subprocess using this computer's internal loopback
    interface.  This connection is not visible on any external
    interface and no data is sent to or received from the Internet.
    ****************************************************************

IDLE 1.1.3
>>> print "Hello there"
Hello there
>>> print "How are you"
How are you
>>> 
```

How to run Python Windows

- Run IDLE to use the interpret
- Open a new window in IDLE to write and save programs

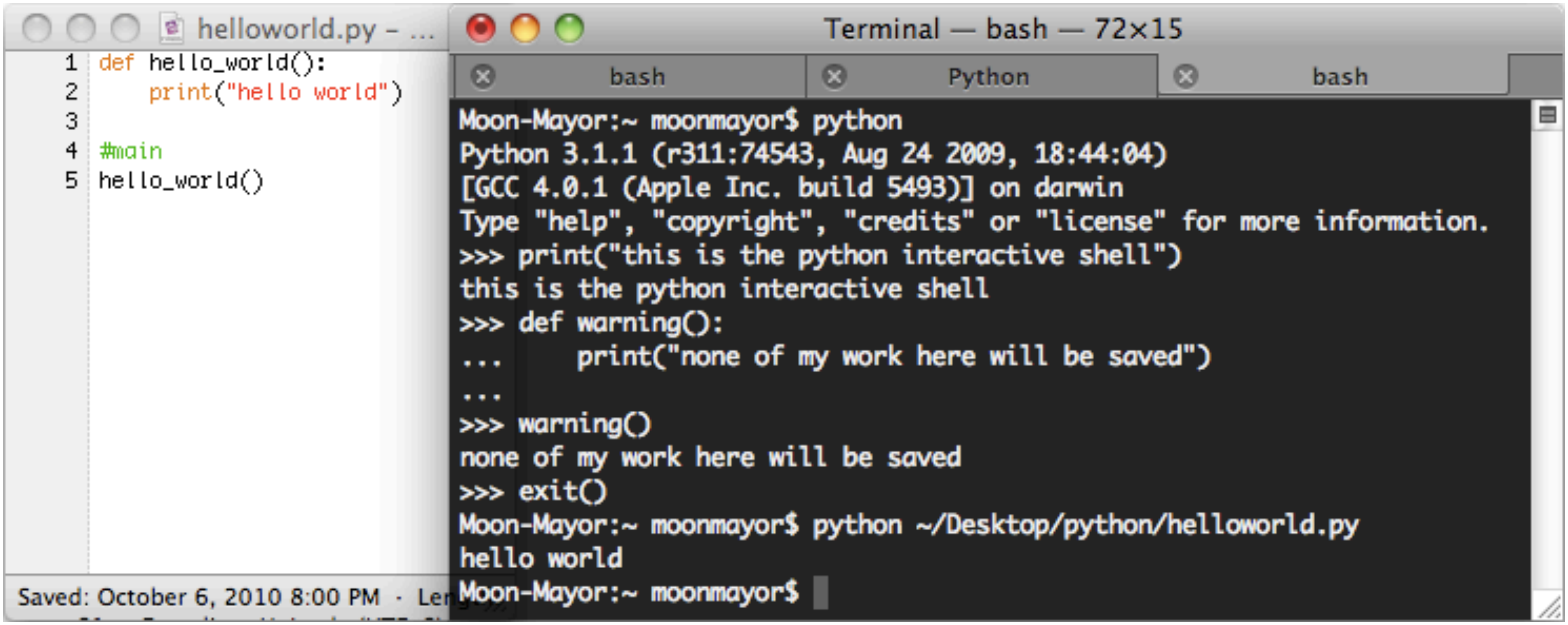

Python Shell
File Edit Shell Debug Options Windows Help
Ln: 18 Col: 4
helloworld.py - C:/Documents and Settings/...
File Edit Format Run Options Windows Help
Ln: 3 Col: 31

```
Python 3.1.2 (r312:79149, Mar 21 2010, 00:41:52) [MSC
v.1500 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more i
nformation.
>>> def method():
        print("you can define methods in the interpre
ter")


>>> method()
you can define methods in the interpreter
```

```
def writing_programs():
    print("File > New Window")
    print("creates a place to")
    print("write and save programs.")

# main
writing_programs()
print("This is a good place to write")
print("programs you want to keep!")
print()
print("save and press F5 to run.")
```

How to run Python Unix

- Start the interactive shell with `python`
- Run a program with `python /path/to/program.py`



```python
def hello_world():
    print("hello world")

#main
hello_world()
```

```
Moon-Mayor:~ moonmayor$ python
Python 3.1.1 (r311:74543, Aug 24 2009, 18:44:04)
[GCC 4.0.1 (Apple Inc. build 5493)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> print("this is the python interactive shell")
this is the python interactive shell
>>> def warning():
...     print("none of my work here will be saved")
...
>>> warning()
none of my work here will be saved
>>> exit()
Moon-Mayor:~ moonmayor$ python ~/Desktop/python/helloworld.py
hello world
Moon-Mayor:~ moonmayor$
```

Chapter 1 Review

- Console output: `System.out.println();`
- Methods: `public static void` **`name`**`() { ... }`

Hello2.java

```
public class Hello2 {
    public static void main(String[] args) {
        hello();
    }

    public static void hello() {
        System.out.println("Hello, world!");
    }
}
```

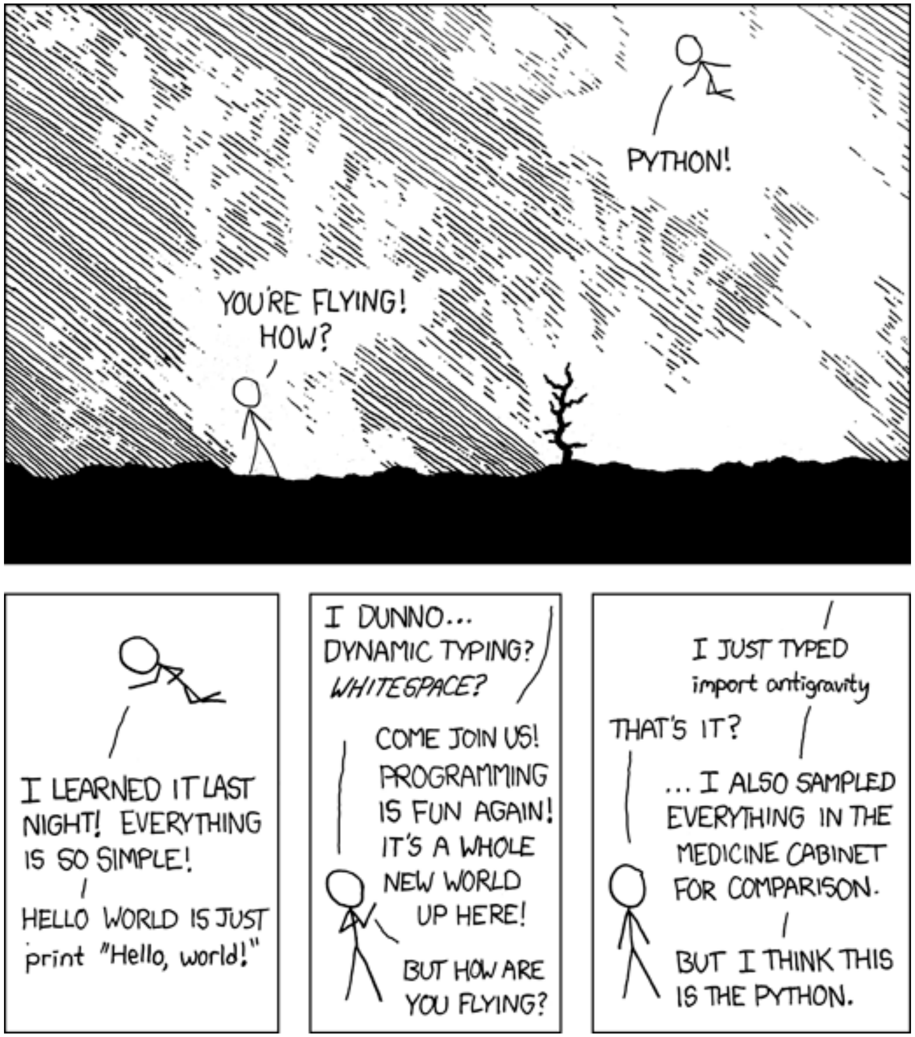
Our First Python Program

- Python does not have a `main` method like Java
  - The program's main code is just written directly in the file
- Python statements do not end with semicolons

hello.py

```
print("Hello, world!")
```

A Brief Review


PYTHON!
YOU'RE FLYING! HOW?
I LEARNED IT LAST NIGHT! EVERYTHING IS SO SIMPLE!
HELLO WORLD IS JUST print "Hello, world!"
I DUNNO... DYNAMIC TYPING? WHITESPACE?
COME JOIN US! PROGRAMMING IS FUN AGAIN! IT'S A WHOLE NEW WORLD UP HERE!
BUT HOW ARE YOU FLYING?
I JUST TYPED import antigravity
THAT'S IT?
... I ALSO SAMPLED EVERYTHING IN THE MEDICINE CABINET FOR COMPARISON.
BUT I THINK THIS IS THE PYTHON.

Python 2.x vs Python 3.x

- – We will be using Python 3 for this course
- – Sometimes we may refer to Python 2
- – The differences are minimal

How to	Python 2.x	Python 3.x
print text	`print “text”`	`print(“text”)`
print a blank line	`print`	`print()`

The `print` Statement

`print(”`**text**`")`
`print()` (a blank line)

- Escape sequences such as `\"` are the same as in Java
- Strings can also start/end with `'`

swallows.py

```
print("Hello, world! ")
print()
print("Suppose two swallows \"carry\" it together.")
print('African or "European" swallows?')
```

Comments

# comment text (one line)
# must start each line of comments with the pound sign

swallows2.py

```
# Suzy Student, CSE 142, Fall 2097
# This program prints important messages.
print("Hello, world!")
Print()                          # blank line
print("Suppose two swallows \"carry\" it together.")
Print('African or "European" swallows?')
```

Functions

- **Function**: Equivalent to a static method in Java.
- Syntax:

```
def name():
    statement
    statement
    ...
    statement
```

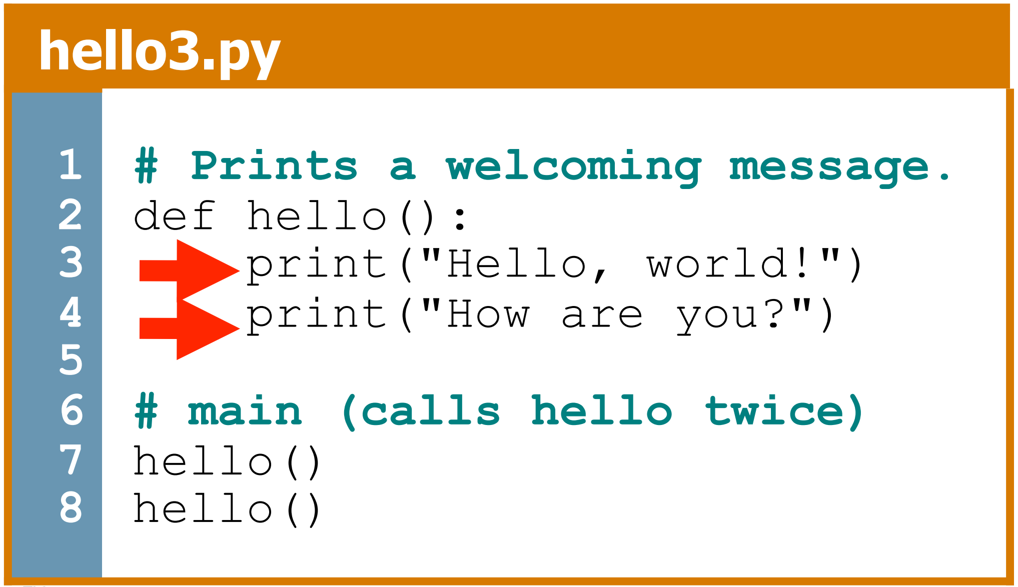
hello2.py

```python
# Prints a helpful message.
def hello():
    print("Hello, world!")

# main (calls hello twice)
hello()
hello()
```

  - Must be declared above the 'main' code
  - Statements inside the function must be indented

Whitespace Significance

- Python uses indentation to indicate blocks, instead of `{ }`
  - Makes the code simpler and more readable
  - In Java, indenting is optional. In Python, you **must** indent.
  - You may use either tabs or spaces, but you **must** be consistent



hello3.py

```
# Prints a welcoming message.
def hello():
    print("Hello, world!")
    print("How are you?")

# main (calls hello twice)
hello()
hello()
```

Tabs or spaces

shell

```
# Prints a helpful message.
>>> def indent_reminder():
... print("Remember, you must indent!")
  File "<stdin>", line 2
    print("Remember, you must indent!")
        ^
IndentationError: expected an indented block
>>> def indent_reminder():
...     print("Remember, you must indent!")
...
>>> indent_reminder()
Remember, you must indent!
>>>
```

Tabs or spaces

shell

```
# Prints a helpful message.
>>> def indentation_errors():
...     print("this was indented using a tab")
...     print("this was indented using four spaces")
  File "<stdin>", line 3
    print("this was indented using four spaces")
                                               ^
IndentationError: unindent does not match any outer
indentation level
>>> def indentation_errors():
...     print("this was indented using a tab")
...     print("so this must also use a tab")
...
>>> def more_indentation_tricks():
...     print("If I use spaces to indent here.")
...     print("then I must use spaces to indent here.”)
>>>
```

Exercise

- Rewrite the Figures lecture program in Python. Its output:

```
  ______
 /      \
/        \
\        /
 \______/

\        /
 \______/
+--------+

  ______
 /      \
/        \
|  SPAM  |
\        /
 \______/

  ______
 /      \
/        \
+--------+
```

Exercise Solution

```
def egg():
    top()
    bottom()
    print()

def cup():
    bottom()
    line()
    print()

def stop():
    top()
    print("|  SPAM  |”)
    bottom()
    print()

def hat():
    top()
    line()
    print()
```

```
def top():
    print("  ______”)
    print(" /      \\”)
    print("/        \\”)

def bottom():
    print("\\        /”)
    print(" \\______/”)

def line():
    print("+--------+”)

# main
egg()
cup()
stop()
hat()
```