

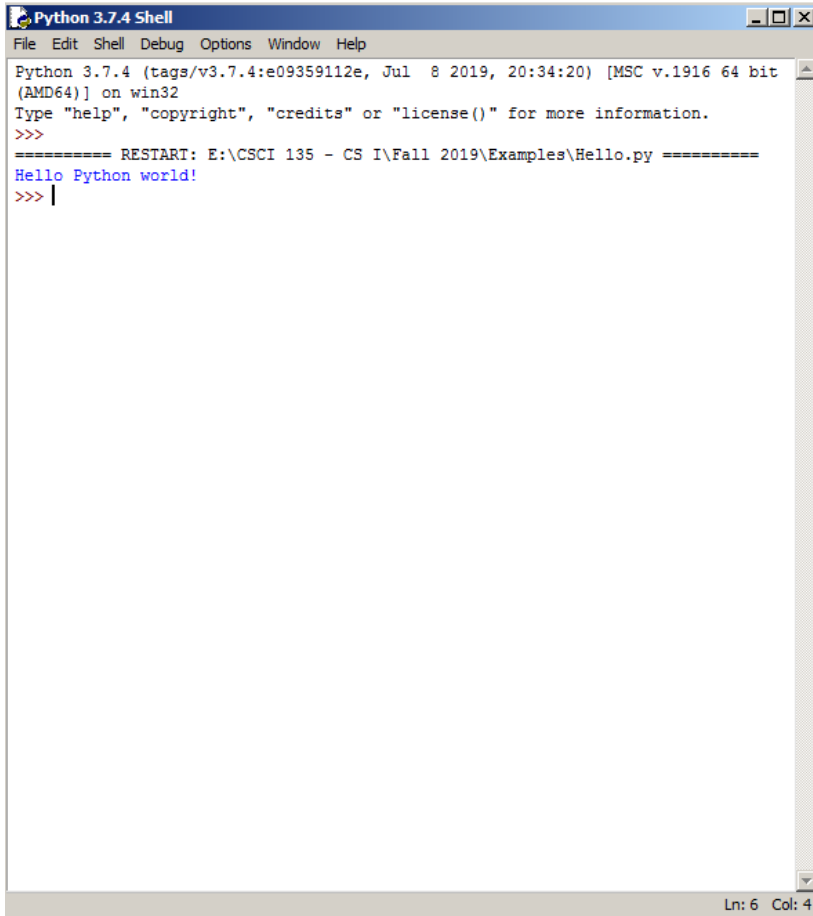
# GETTING STARTED; VARIABLES

---

# Outline

- Getting Started
  - Idle Shell and Editor
  - Command Window
  - Python Versions
- Variables
  - What is a Variable?
  - Variable Names
  - Working with Variables
  - Different Types of Variables
    - Simple Data Types
- Comments

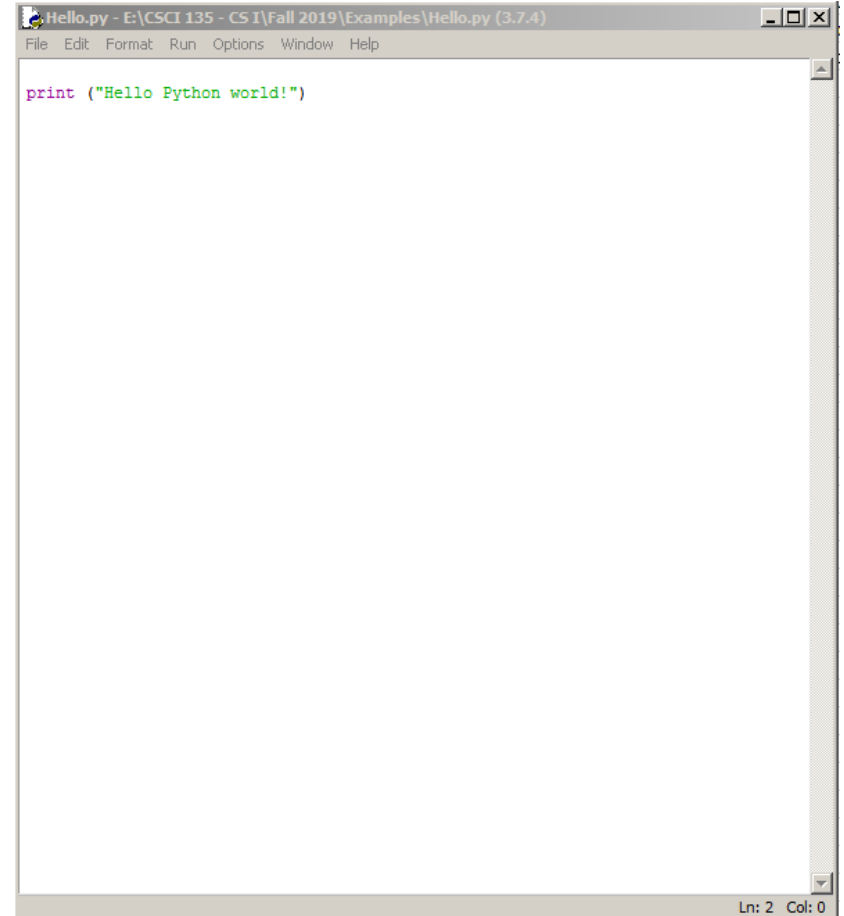
# Getting Started: Idle Shell and Editor



The screenshot shows the Python 3.7.4 Shell window. The title bar reads "Python 3.7.4 Shell". The menu bar includes "File", "Edit", "Shell", "Debug", "Options", "Window", and "Help". The main text area contains the following text:

```
Python 3.7.4 (tags/v3.7.4:e09359112e, Jul 8 2019, 20:34:20) [MSC v.1916 64 bit  
(AMD64)] on win32  
Type "help", "copyright", "credits" or "license()" for more information.  
>>>  
===== RESTART: E:\CSCI 135 - CS I\Fall 2019\Examples\Hello.py =====  
Hello Python world!  
>>> |
```

The status bar at the bottom right indicates "Ln: 6 Col: 4".



The screenshot shows the Python 3.7.4 Editor window. The title bar reads "Hello.py - E:\CSCI 135 - CS I\Fall 2019\Examples\Hello.py (3.7.4)". The menu bar includes "File", "Edit", "Format", "Run", "Options", "Window", and "Help". The main text area contains the following code:

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

The status bar at the bottom right indicates "Ln: 2 Col: 0".

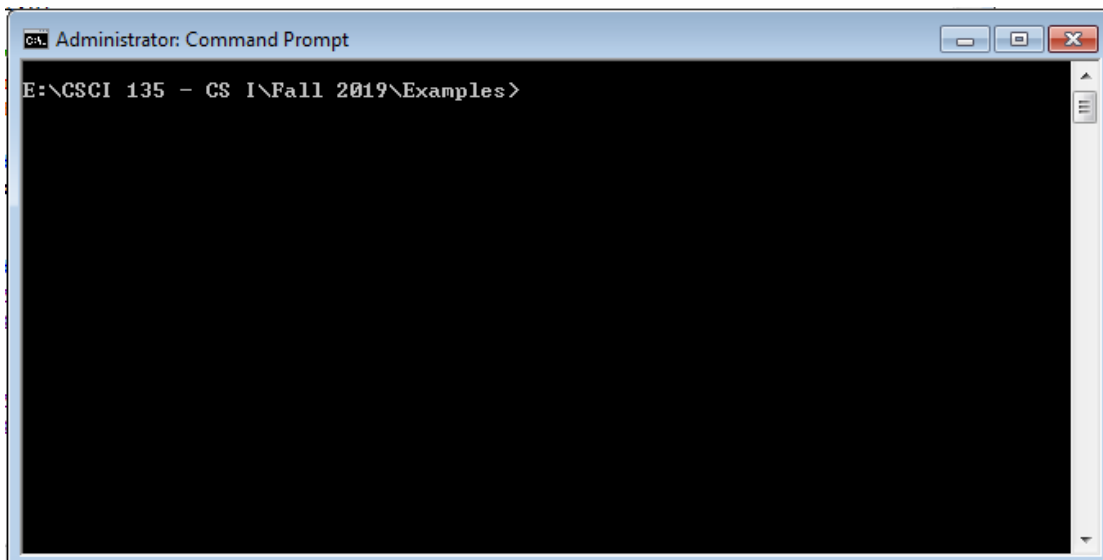
# Getting Started: Command Window

- **GUI** (Graphical User Interfaces)
  - Today: **predominant interaction method**
  - Windows, buttons, mouse
  - Advantages
    - **Easier for novices**
    - **No commands to remember**
    - **Rich input and output capabilities**

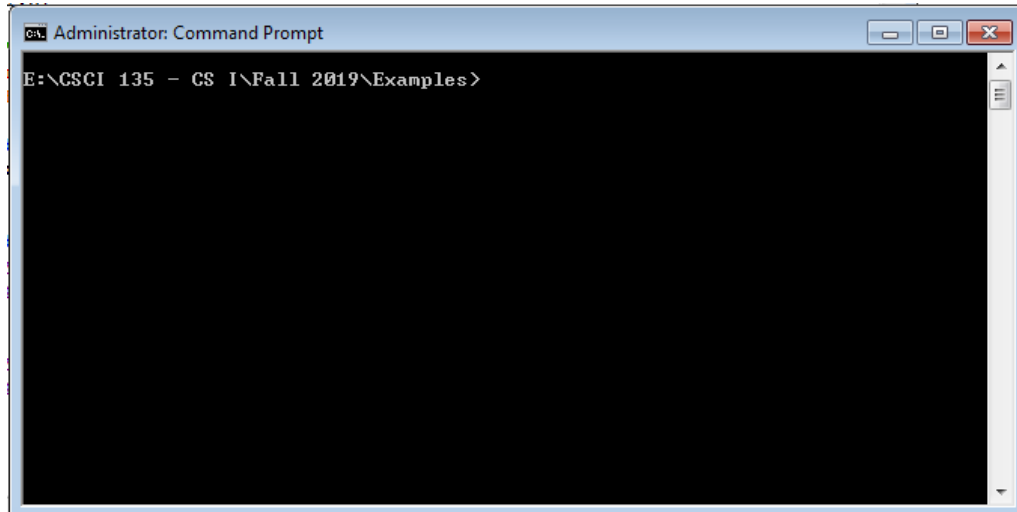


# Getting Started: Command Window

- **Command Line Interface (CLI)**
  - Originally the only option
  - Input by **typing commands**
  - Advantages:
    - Can be **faster for experts** than a GUI
    - Easier to **automate** tasks
    - Easier to **hook programs together**



# Getting Started: Command Window

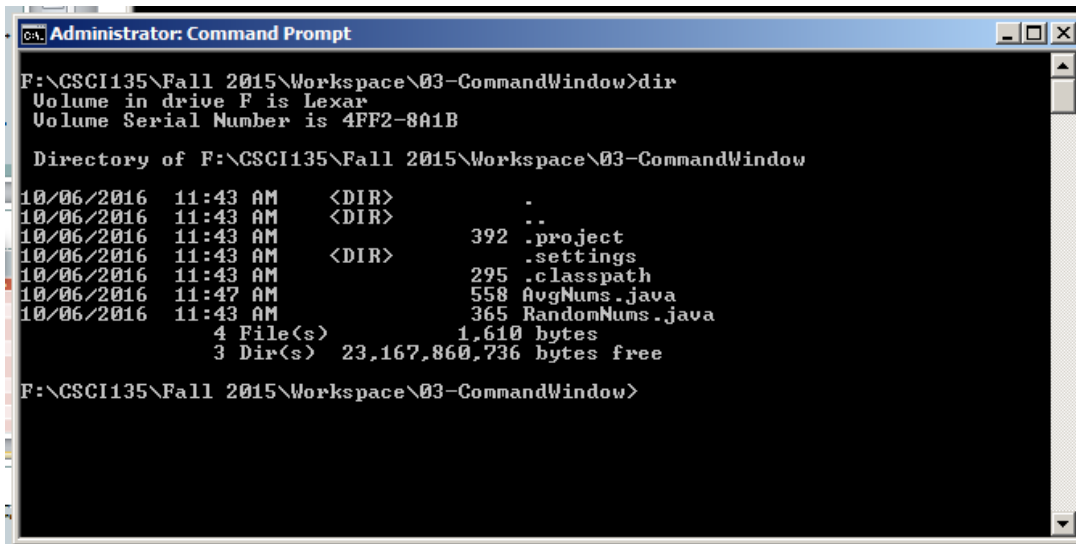


## *Windows 10*

*Cortana → type "cmd"*

*All Programs → Accessories →  
Command Prompt*

# Getting Started: Command Window



```
Administrator: Command Prompt
F:\CSCI135\Fall 2015\Workspace\03-CommandWindow>dir
Volume in drive F is Lexar
Volume Serial Number is 4FF2-8A1B

Directory of F:\CSCI135\Fall 2015\Workspace\03-CommandWindow

10/06/2016  11:43 AM    <DIR>          .
10/06/2016  11:43 AM    <DIR>          ..
10/06/2016  11:43 AM             392 .project
10/06/2016  11:43 AM    <DIR>          .settings
10/06/2016  11:43 AM             295 .classpath
10/06/2016  11:47 AM             558 AvgNums.java
10/06/2016  11:43 AM             365 RandomNums.java
            4 File(s)          1,610 bytes
            3 Dir(s)    23,167,860,736 bytes free

F:\CSCI135\Fall 2015\Workspace\03-CommandWindow>
```

Looking at the contents of a folder  
Windows: dir

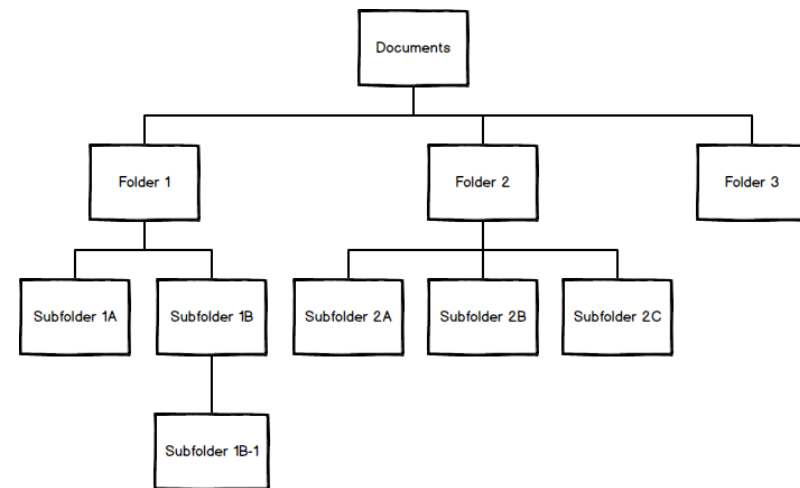
# Directory Structure

- “Folders”/Directories organized in a tree structure

- Root is at the top, branches below
- Files are stored in folders/directories
- On Windows, different devices have different letters
  - Primary hard drive is C:
  - At Tech, user directories are on D:
  - Flash drives are usually E: onward

- Navigating the tree

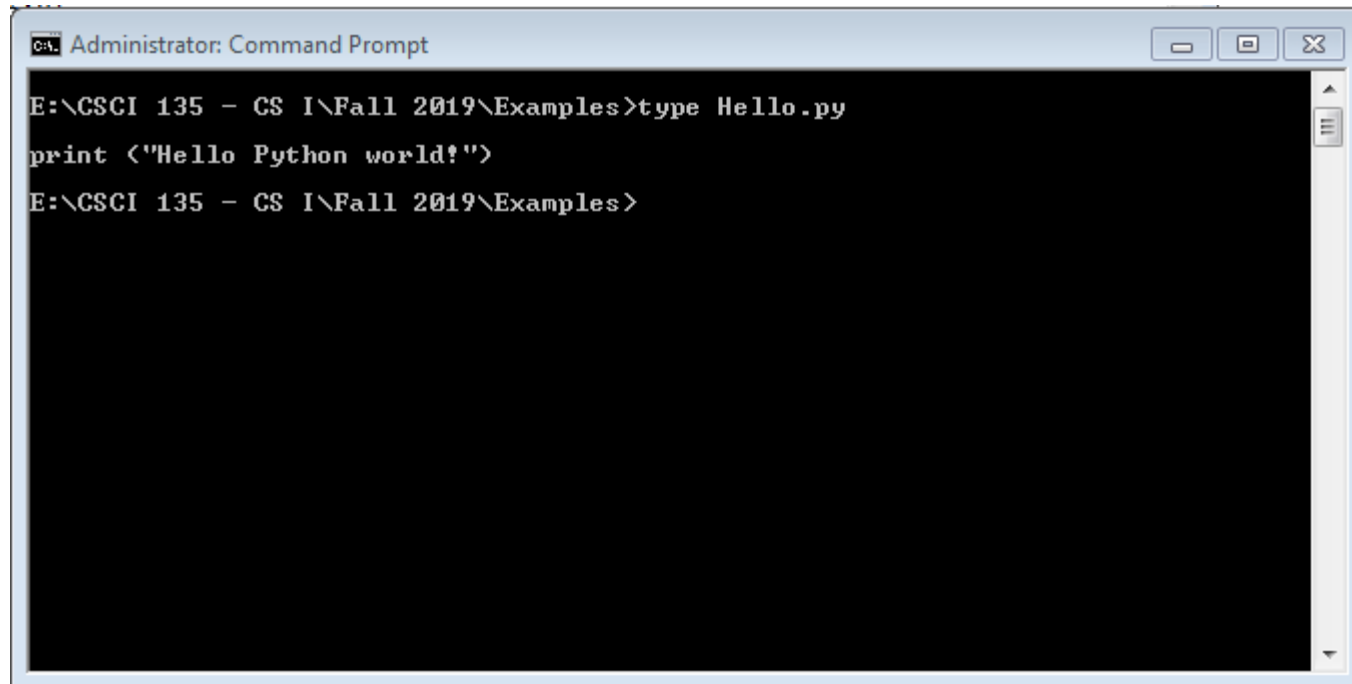
- To change to a directory:
  - Windows: `cd C:\Documents\Folder 1\Subfolder 1A`
  - Up one directory level: `cd ..`
  - The current directory: `.`
  - Where am I?
    - Windows: usually shown in the “prompt”





# Getting Started: Command Window

- To display the contents of a file to the screen on Windows:  
> type Hello.py



```
Administrator: Command Prompt
E:\CSCI 135 - CS I\Fall 2019\Examples>type Hello.py
print <"Hello Python world!">
E:\CSCI 135 - CS I\Fall 2019\Examples>
```

# Running a Python Program

- Windows:
  - > `python HelloWorld.py`
- If it all runs correctly, you'll get the program results and a prompt
- If things go wrong, you will get an error and a cryptic description of what went wrong
  - Will give you some clues
  - Over time, these will become more understandable

# Summary of Helpful Commands

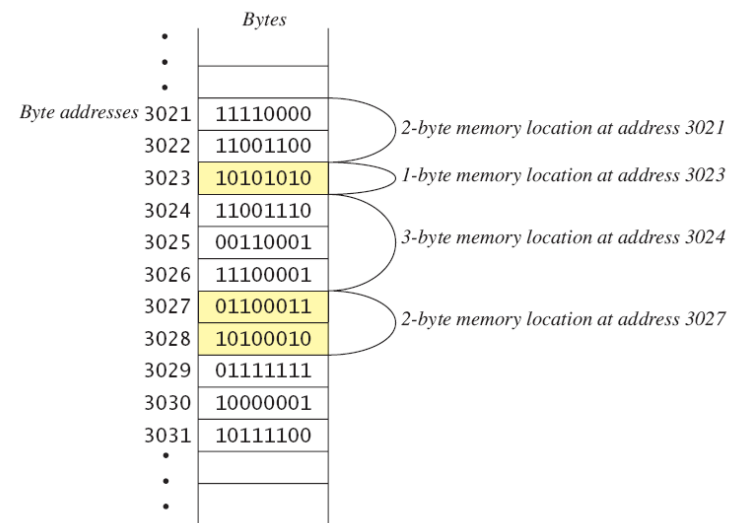
Action	Windows	Mac OS / Unix
Move into a folder	<code>cd myfolder</code>	<code>cd myfolder</code>
Move into parent folder	<code>cd ..</code>	<code>cd ..</code>
Move into a folder, absolute folder	<code>cd \Users\keith</code>	<code>cd /Users/keith</code>
List files in current folder	<code>dir</code>	<code>ls</code>
Compile and run a program in current folder	<code>python Prog.py</code>	<code>python Prog.py</code>
See what is in a text file	<code>type Prog.py</code>	<code>more Prog.py</code>
Auto-complete filenames	<code>&lt;tab key&gt;</code>	<code>&lt;tab key&gt;</code>
Previous command	<code>&lt;up arrow&gt;</code>	<code>&lt;up arrow&gt;</code>

# Getting Started: Python Versions

- Why do I have to type “python myProgram.py” in lab but not on my own computer?
- Why do I have to use the command window – on my computer, I can use command line arguments right from the editor?

# Variables: What is a Variable?

- *Variables* store data such as numbers and letters.
  - Think of them as places to store data.
  - They are implemented as memory locations.
- The data stored in a variable is called its *value*.
  - The value is stored in the memory location.
- Its value can be changed



# Variables: Variable Names



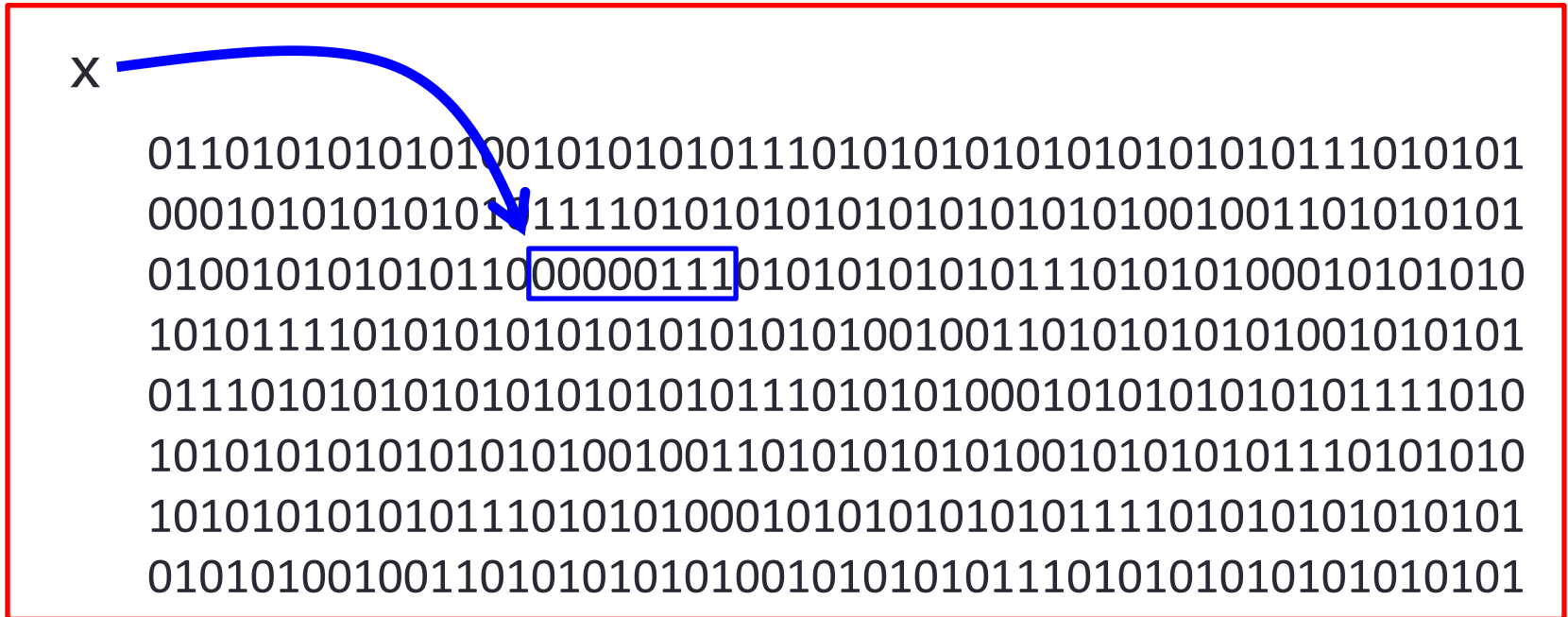
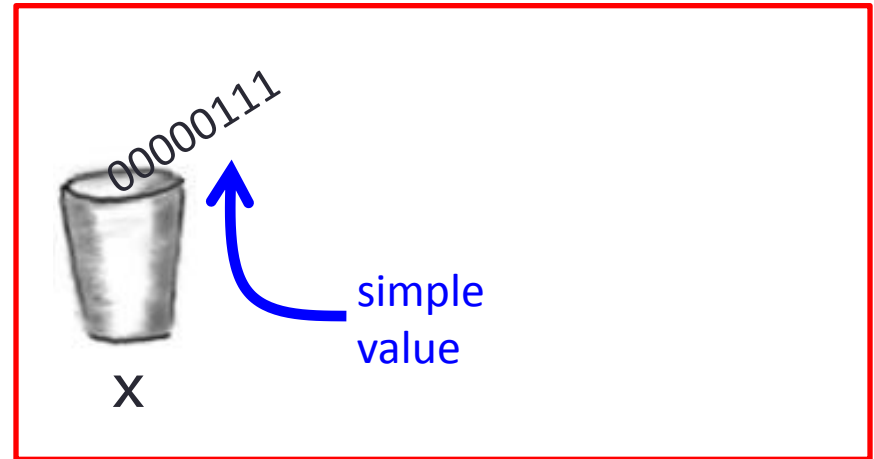
- A variable's name should suggest its use
  - e.g. **taxRate**, **count**, **sum**, etc.
  - That is, it should be meaningful
- Variable names can only contain letters, numbers and underscores
  - That means no spaces or punctuation characters.
  - The name must begin with a letter
  - You shouldn't use python keywords ("print" would not be a good name for a variable)
  - Python is case sensitive, so x and X are different variables
- One convention "lower camel case"
  - Begin with a lower case letter and then each new word is upper case
  - For example, totalWidgets

# Variables: Working with Variables

- To create a variable, simply name it and assign a value to it:
  - >>> myName = "Michele"
    - The Idle editor (or shell) will highlight text in green
  - >>> count = 0
    - There is no color highlighting here...
- Once you've created a variable, you can change its value:
  - >>> myName = "Rufus"
    - myName now has a different value
  - >>> count = count + 1
    - What?!? Is that legal?

# Creating and Initializing a Simple Variable

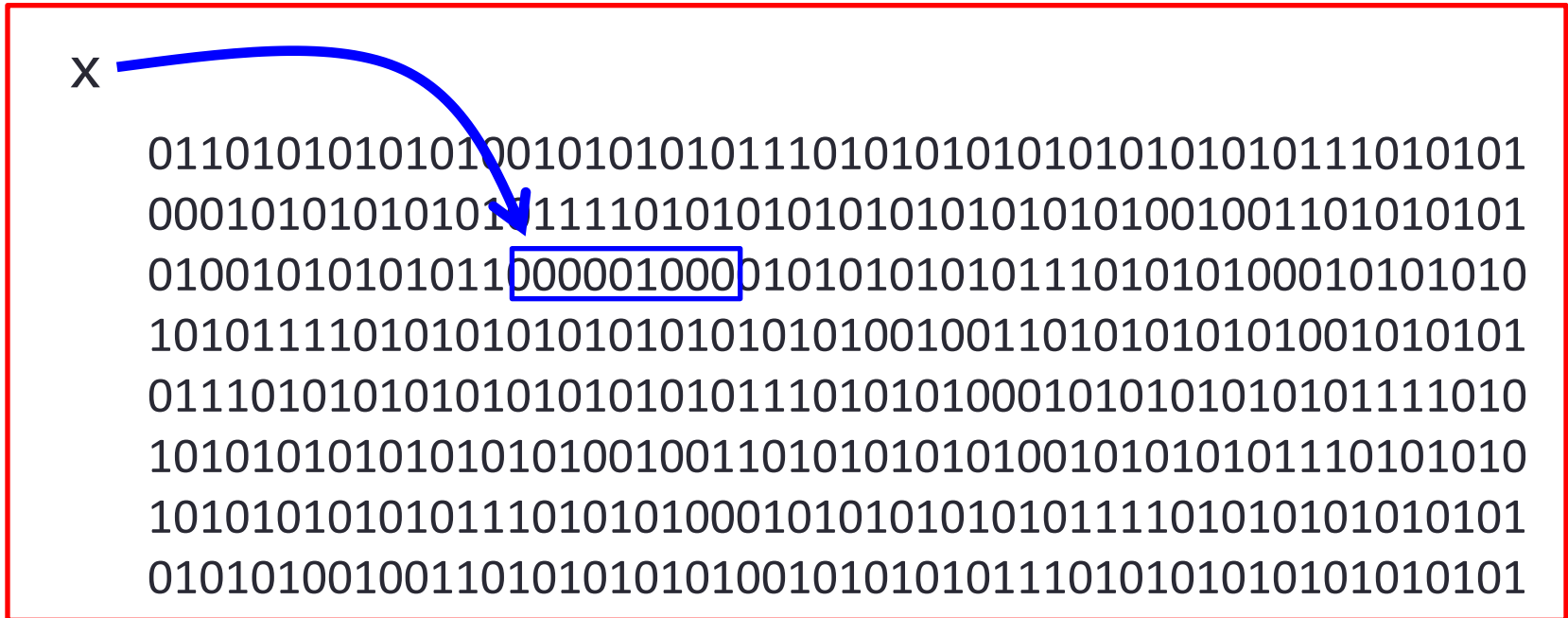
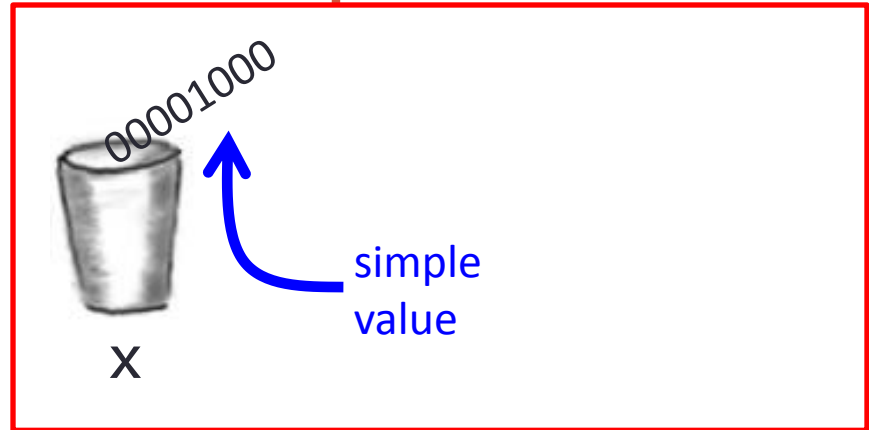
```
x = 7;
```





# Changing the Value of a Simple Variable

```
x = 7;  
x = x + 1;
```



# Variables: Different Types

- Simple Data Types

- Strings
  - Represent text
- Numbers
  - Integers and floating point number
    - Integers have no decimal point – they are whole numbers (e.g. 10)
    - Floating point numbers do have a decimal point (e.g. 3.1415)
- Booleans
  - Logical data type
    - Either True or False

- You can find the type of a variable in the Python Idle shell by using the “type” command:

```
>>> x = 10
>>> type(x)
<class 'int'>
```

# Data Types: Constants

- Sometimes you have a value that should not change
  - e.g. pi, my favorite number, the speed of light
- Values that shouldn't change are called *constants*.
- Floating-point constants can be written
  - With digits after a decimal point or
  - Using *e notation*.
- Naming convention
  - All upper case, use `_` between words

```
>>> SPEED_LIGHT = 3.0e8
```



# Variables and Data Types

- Variables
  - **Stores information** your program needs
  - Each has a **unique name**
  - Each has a specific **type** that Python infers

Python simple type	what it stores	example values	operations
<b>int</b>	integer values	42 1234	add, subtract, multiply, divide, remainder, compare, increment, decrement
<b>float</b>	floating-point values	9.95 3.0e8	add, subtract, multiply, divide, remainder, compare
<b>str</b>	sequence of characters	"Hello world!" 'I love this!'	concatenate, and more
<b>bool</b>	truth values	True False	and, or, not

# Changing the Data Type

- You can't do math with text
  - Input comes in as text
  - To change a text variable, say x, to an integer:  

```
>>> int(x)
```
  - To change a text variable, say x, to a floating point number:  

```
>>> float(x)
```
  - Can we change x to a boolean?  

```
>>> bool(x)
```

Sure! (Results may be surprising, though)
  - Can we change a number to text?  

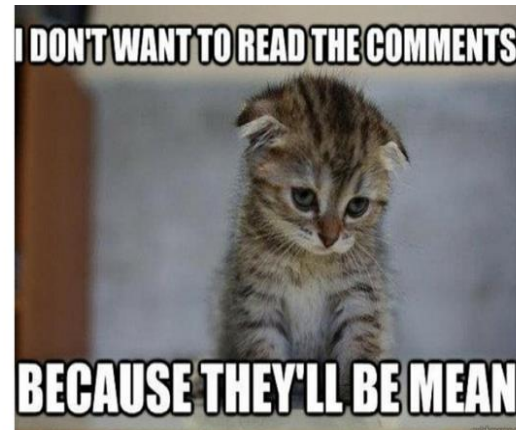
```
>>> str(10)
```

# Comments

- The best programs are self-documenting.
  - Clean style
  - Well-chosen names
- Comments are written into a program as needed to explain the program.
  - They are useful to the programmer, but they are ignored by the compiler.
  - You must always include a header comment with your name and a short description of the program

## # comment to end of line

- The Idle editor will highlight these in red



# Summary

- Getting Started
  - Idle Shell and Editor
  - Command Window
  - Python Versions
- Variables
  - What is a Variable?
  - Variable Names
  - Working with Variables
  - Different Types of Variables
    - Simple Data Types
- Comments



# Your Turn

- Open the Idle shell and try the following commands interactively:

```
thisClass = "CSCI 135"  
type(thisClass)  
thisClass
```

```
count = 0  
type(count)  
count = count + 1  
count
```

```
isOK = True  
type(isOK)  
int(isOK)
```

```
str(isOK)  
str(count)  
int(thisClass)
```

```
count = count + 0.5  
type(count)  
count
```

On normal “Your Turn” class assignments, I would have you turn in your work to Moodle for extra credit. For this one, there is nothing to turn in – I just want you to experiment with different data types and variables in the Python shell.

# why do you think this was the answer?

# makes sense, right?

# your knew that would happen, didn't you?

# did the type change from what it was before?