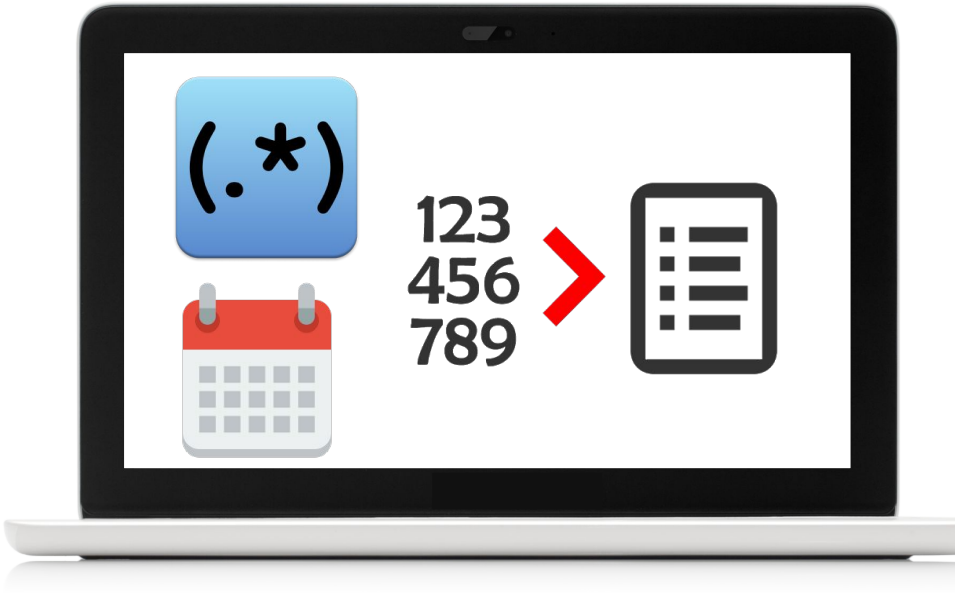
Standard Library

Please go through material for these topics. Complete the reading, exercises, and any videos linked. If the instructions ask to turn in any exercises, please do so through slack to your instructor.



Cognixia™

Outline

1. sys module
2. math module
3. random module
4. time module
5. datetime module
6. timeit module
7. os module
8. os.path module
9. csv module
10. json module
11. regular expressions
12. re module

Video Here: https://youtu.be/fzshUdX-WoQ

Instructions

- ➔ Please review all sections on the topics listed in the outline
  - ◆ Read through material, watch accompanying videos, run coding examples, etc.
- ➔ **Exercise are optional** and you are **NOT required to turn them in** unless otherwise stated
  - ◆ It is ***recommended you try them*** regardless to help understand the topics better
- ➔ The **open note quiz** at the end is **REQUIRED**
  - ◆ Have it ***turned in by the start of class the next day***
  - ◆ Ask your instructor if you have any questions regarding this
- ➔ Once you have turned in the quiz, feel free to leave for the day

Built-In Libraries

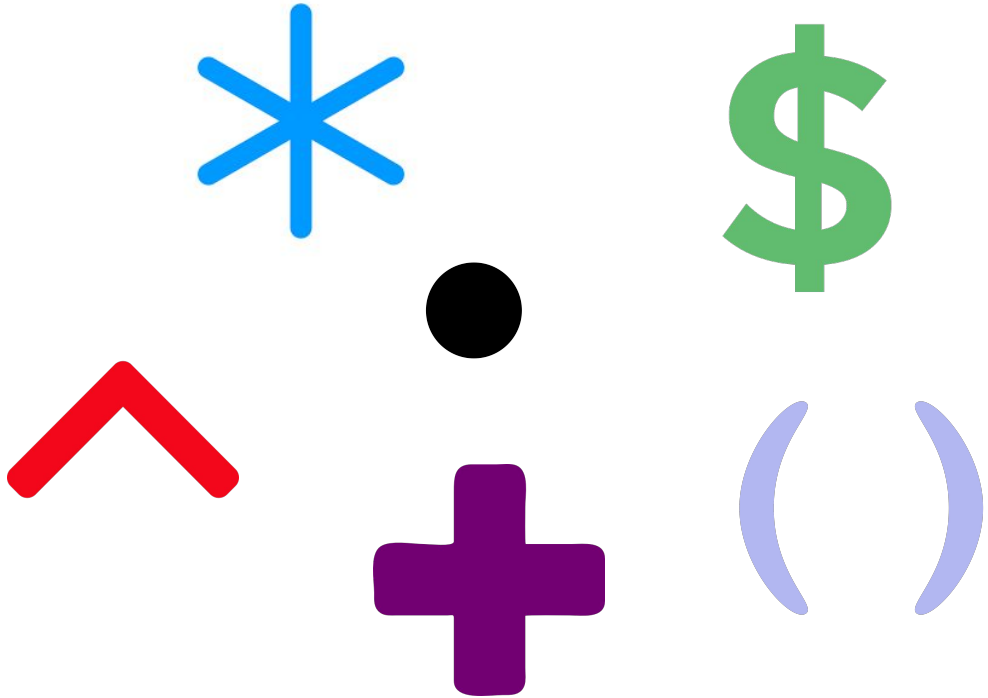
Standard Modules

`sys`	This module provides access to some variables used or maintained by the interpreter and to functions that interact strongly with the interpreter.
`math`	This module provides access to the mathematical functions defined by the C standard.
`random`	This module implements pseudo-random number generators for various distributions.
`time`	This module provides various time-related functions.
`datetime`	The datetime module supplies classes for manipulating dates and times.
`timeit`	This module provides a simple way to time small bits of Python code.
`os`	This module provides a portable way of using operating system dependent functionality.
`shutil`	The shutil module offers a number of high-level operations on files and collections of files.

sys Module

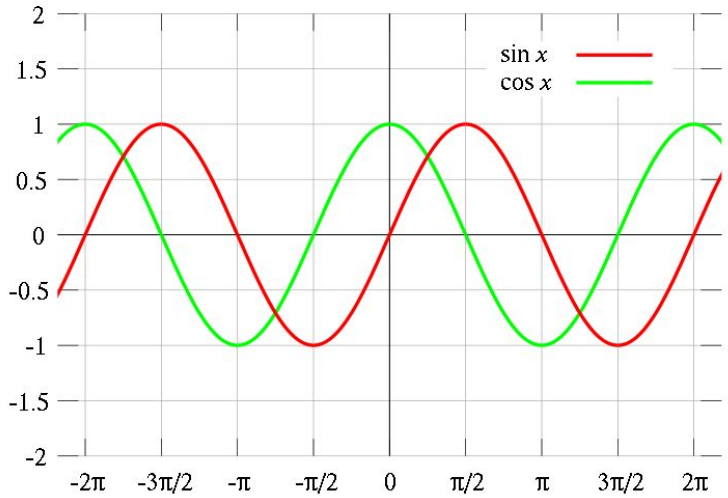
- Contains methods and variables to interact with the **runtime environment**
- Contains the Python version information
  - **sys.version**
- Interacts with the standard input/output
  - **sys.stdin**
  - **sys.stdout**
  - **sys.stderr**
- Reads the command line arguments
  - **sys.argv**
  - read as a List
- Can exit the program
  - **sys.exit(<exit message>)**

Math Modules



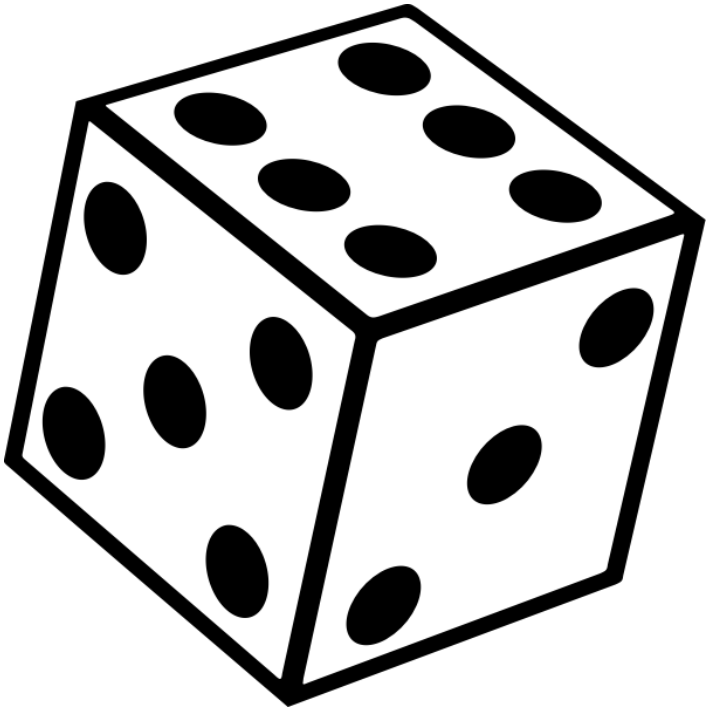
math Module

- Contains methods and variables to perform complex mathematical operations
- Trigonometric functions
  - **math.sin(<number)**
  - **math.cos(<number>)**
  - **math.tan(<number>)**
- Exponential functions
  - **math.log(<number>)**
  - **math.exp(<number>)**
- Useful constants
  - **math.e**
  - **math.pi**
  - **math.inf**



random Module

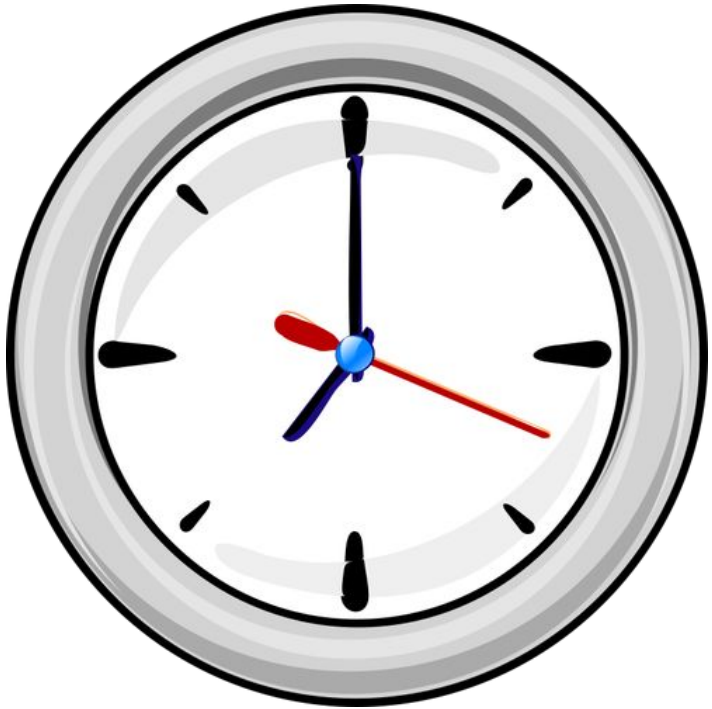
- Module for generating various types of random numbers
- **random.random()**
  - Random number between 0 and 1
- **random.randint(start, stop)**
  - Random number in the range
- **random.sample(<list>, <number>)**
  - Random element from the list
- **random.shuffle(<list>)**
  - The list in a random order
- random also contains distributions based on probability and statistics



Date and Time Modules

time Module

- Module for interacting with **epoch time**
  - Jan 1, 1970
- **time.time()**
  - Current time since the epoch
- **time.gmtime()**
  - Returns a structure with information about the current time
- **time.ctime(<seconds>)**
  - Current time as a string
  - Can be passed an epoch time
- **time.sleep(<seconds>)**
  - Pauses the execution of the program for a number of seconds



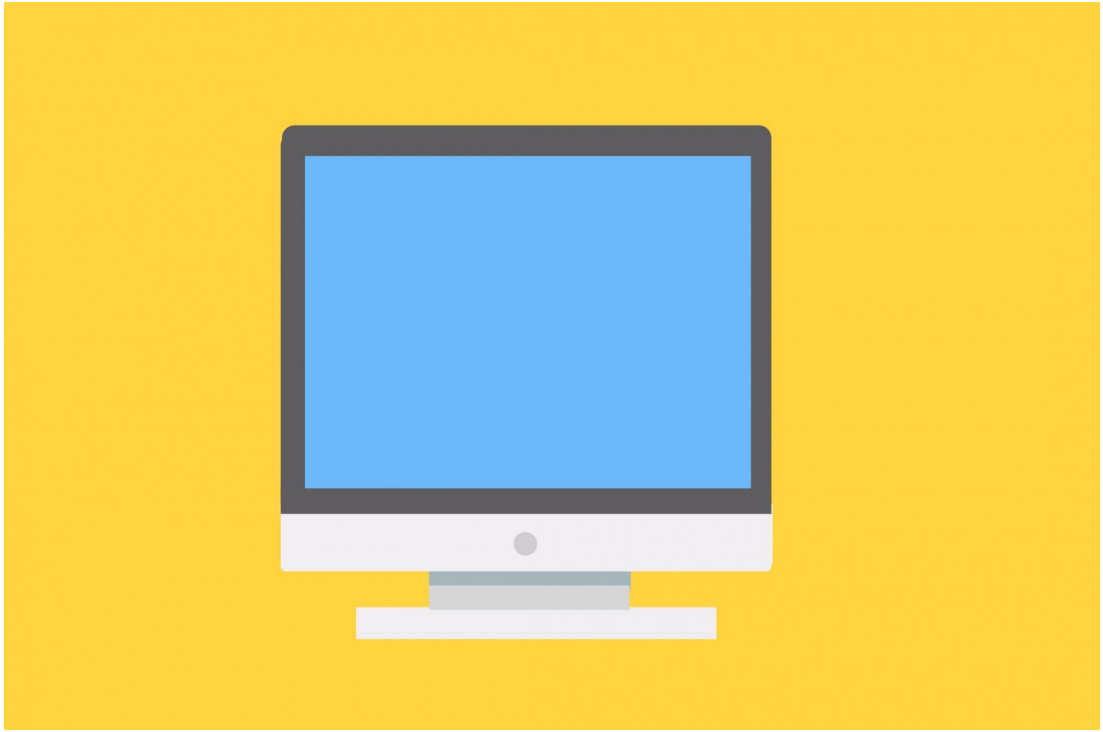
datetime Module

- Module for interacting with **dates** and **times**
- datetime objects contain date and time attributes
  - **.year**
  - **.hour**
- **datetime.datetime.now()**
  - Returns a datetime object for current date
- **datetime.datetime(<year>, <month>,<day>)**
  - Constructor that returns a new datetime object for the given date
- **strftime(<format>)**
  - Formats the date into a specified format

timeit Module

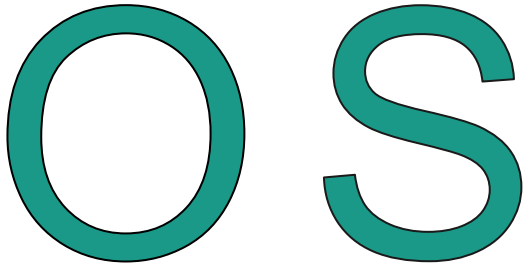
- Module for timing the execution of code snippets
- **timeit.timeit(stmt, setup, timer, number)**
- **stmt**
  - The code you want to time as a string
- **setup**
  - Initial code to be run before the test
- **Timer**
  - Defaults to timeit's internal timer
- **number**
  - The number of times the test is to be run

OS Operation Modules



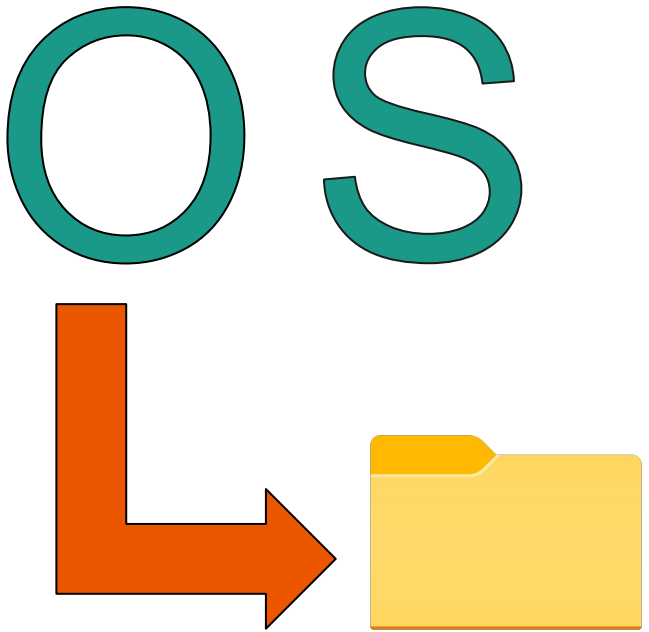
os Module

- Module for interacting with the **operating system**
- **os.getcwd()**
  - Returns the current working directory
- **os.chdir()**
  - Changes the working directory
- **os.mkdir()**
  - Creates a directory
- **os.listdir(<path>)**
  - Returns a list of directories
- **os.rmdir(<path>)**
  - Removes an empty directory
- **os.popen()**
  - Opens a data pipe to a source



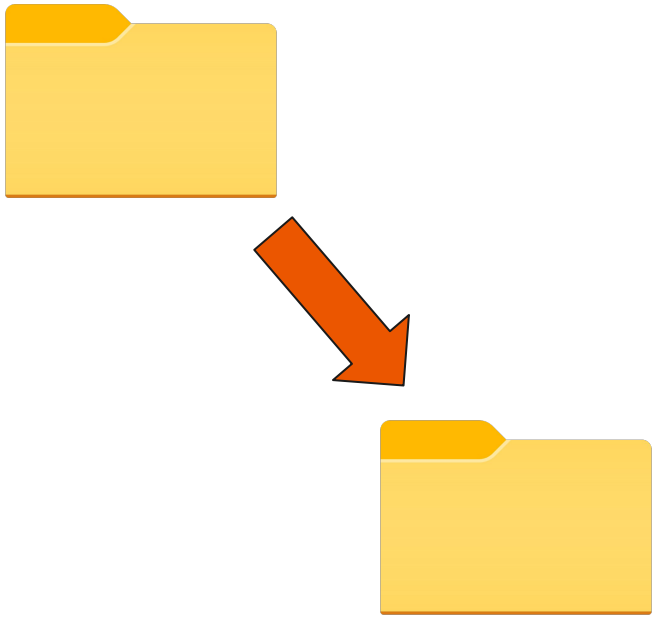
os.path Module

- Submodule of **os**
- Module for interacting with **file paths**
- **os.path.basename(<path>)**
  - Returns the base name of a path
- **os.path.dirname(<path>)**
  - Returns the parent directory of the base
- **os.path.isdir(<path>)**
  - Checks if the path leads to a directory
- **os.path.join(<path>,*<paths>)**
  - Returns a new path with the base path and the list of arguments to be added to the path

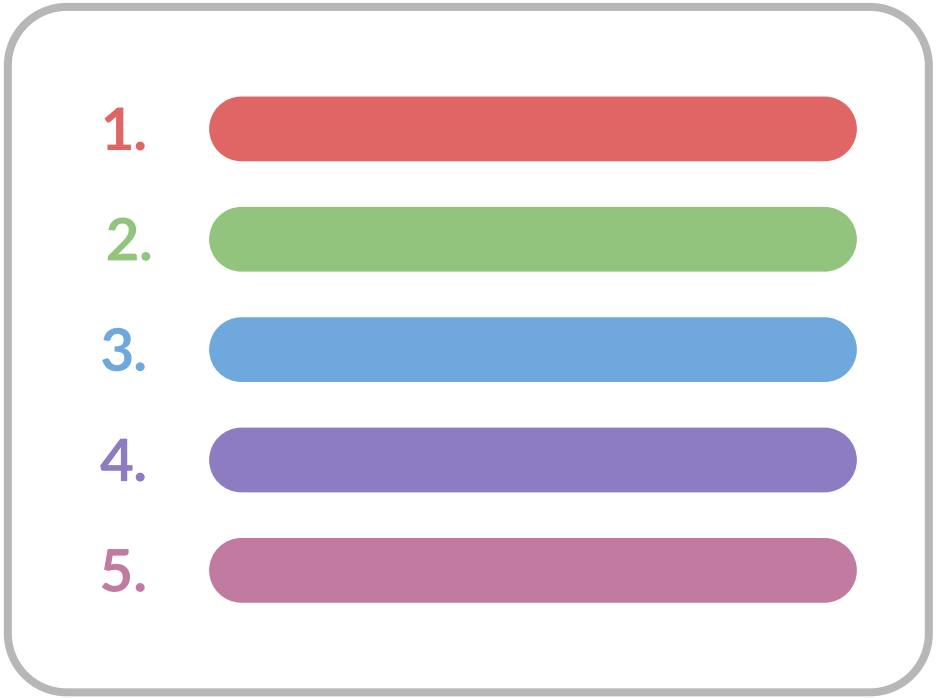


shutil Module

- ➔ Module for high-level operations on **files**
- ➔ **shutil.move(<source>,<destination >)**
  - ◆ Moves a file from a source to a destination
- ➔ **shutil.copy(<source>,<destination >)**
  - ◆ Copies a file at a source to a destination
- ➔ **shutil.which(<executable>)**
  - ◆ Returns the path to the executable that will be runif that command would be run



File Modules



csv Module

- ➔ Python has a built-in module for handling **Comma Separated Values** files
- ➔ **csv.reader(<file>)** creates an iterable object and each row can be extracted

json Module

- **Javascript Object Notation** (**JSON**) is a human and machine readable format
- Python has a built in **json** module to parse to and from **JSON** format
  - **.dumps(<object>)**
  - **.loads(<str>)**

Student Exercise

- ➔ Persist the data from your Employee program
- ➔ Incorporate the **datetime** module to record employees' birthday information
- ➔ Create a file called **employees.csv**
  - ◆ write the data for each employee as a row
  - ◆ Import the data from **employees.csv** at runtime
- ➔ Make a file called **employees.json**
  - ◆ Write the employee data o this file as well
  - ◆ If **employees.csv** doesn't exist, load data from **employees.json**

Regular Expressions

The universal language for character pattern matching

REGEX: Pattern Matching

A **regular expression** is a string that describes a search pattern. They can be used to find certain substring patterns in a string or used to validate user input.

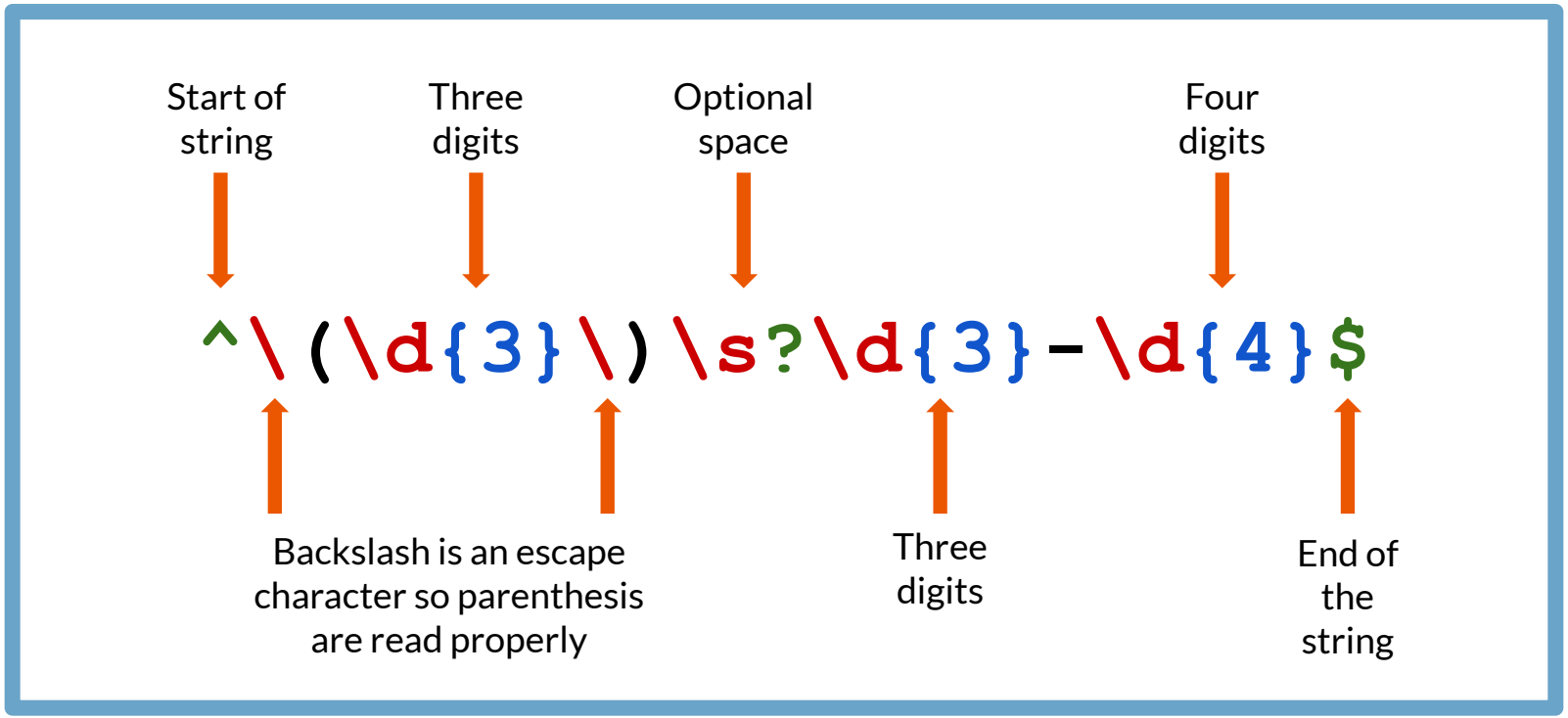
```
^\(\d{3}\)\s?\d{3}-\d{4}$
```

```
(012)345-6789
```

^	If first in the regex, denotes the start or as a negation	`^Hello` → string must start with word “Hello” `[^a]` → string cannot have the character “a”
$	Denotes the end of a string	`Hello$` → string must end with word “Hello”
*	Zero or more occurences	`ba*b` → string has two b’s with zero or more a’s between them
+	One or more occurences	`b(ac)+b` → string has two b’s with one “ac” r multiple “ac” strings between them
?	Zero or one occurrence	`bc?b` → string has two b’s with nothing between them or a single c
{}	Matches a string in the range specified	`abc{2, 5}` → string starts with ab and follows with two to five c’s
() and |	Used to provide section of choices	`b(a|i|e)t` → string is either bat, bit, or bet
[]	Same as above	`b[aie]t` → string is either bat, bit, or bet

.	Any character, can be a alphanumeric or a symbol	`b.c` → string can bac, b5c, b@c, etc.
[0-9]	Digit from 0 t 9	`[0-9][0-9]` → string from 00 to 99
[a-z]	Lowercase letter, switch to capital A and Z for all uppercase letters	`[a-z]+` → string with a lowercase characters repeated one or more times
[a-zA-Z]	A letter, lowercase or uppercase	`[a-zA-Z]{2}` → a two character string with any letter (wE, BP, ee...)
\w	A word character: letter, number, or an underscore	`\w+@gmail.com` → hello@gmail.com, 123@gmail.com, b_tr@gmai.com
\d	A digit	`\d*` → string with zero or more digits (4, 48, 489..)
\s	Whitespace character that includes tabs and line breaks	`hello\sworld` → string has a whitespace character like a tab or line break between the two words

Additional resources here and here.


Start of string
Three digits
Optional space
Four digits
^\(\d{3}\)\s?\d{3}-\d{4}$
Backslash is an escape character so parenthesis are read properly
Three digits
End of the string

(XXX) XXX-XXXX

re Module

- Python has the built-in **re** module to handle **regex**
- **.search(<regex>, <str>)**
  - Searches a string for a pattern and returns a match object
- **.findall(<regex>,<str>)**
  - Returns a list of all matches
- **.sub(<regex>,<replace>, <str>)**
  - Replaces all instances of a match with the given string

```
import re
message = "hello world"
match = re.search("or",message)
if match:
   print("regex pattern found")
else:
   print("no match")
```

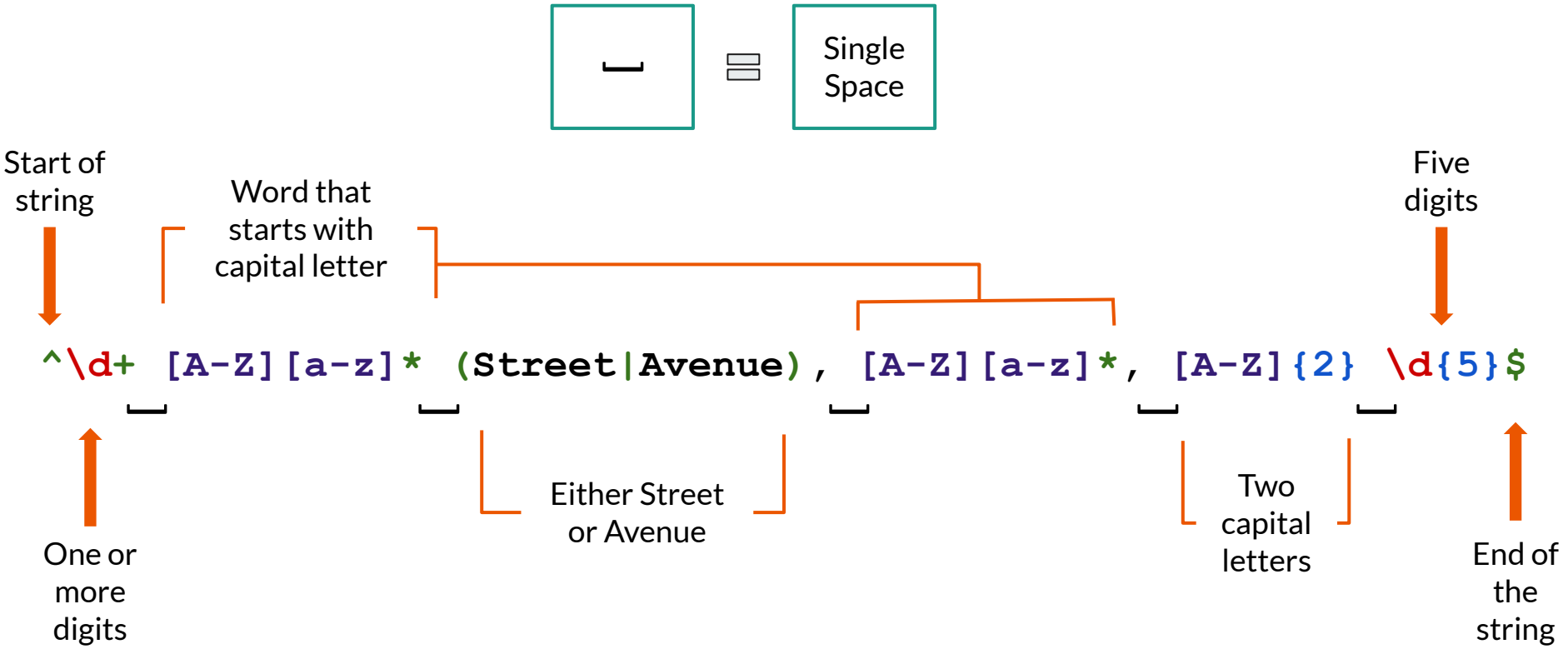
Write a regex expression for an address. For this exercise we will assume that an address is formatted like this:

1 Some Street, City, ST 12345

The street just has to lead with a number. Street name should be two words with only letters. The second word of the street name should be either “Street” or “Avenue”. The city should be one word. The state should be two capital letters, doesn't have to be a real state. The zip can just be exactly 5 digits long. Have a comma before and after the city.

25 Water Avenue, Bee, BZ 45092 ✓

9 South 27th Place, A City, CAB 12378-0123 ✗

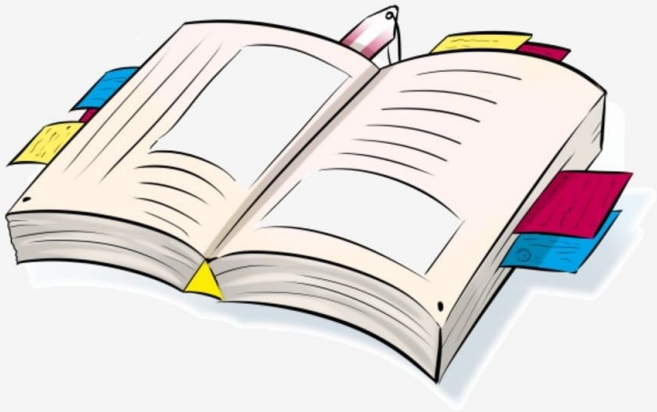

␣
Single Space
Start of string
Word that starts with capital letter
Five digits
^\d+ [A-Z][a-z]* (Street|Avenue), [A-Z][a-z]*, [A-Z]{2} \d{5}$
One or more digits
Either Street or Avenue
Two capital letters
End of the string

Student Exercise

➔ Construct a Python program that will examine each test string and match it against your **regex pattern**

Open Book Quiz on Regex, Enums, and Dates

- ➔ Click Here for Quiz Link
- ➔ This is an **open note**, multiple choice quiz
- ➔ Have it completed by the **start of class tomorrow at 10AM EST**
- ➔ If there are ***any questions, ask your instructor during this time or during office hours***, as they may not able available after hours



Cognixia™