### Printing

print("Hello")

print("World")

print("!")

### Variables and Data Types

'''

Names are case-sensitive and may begin with:

letters, $, \_

After, may include

letters, numbers, $, \_

Convention says

Start with a lowercase word, then additional words are separated

by underscores

ex. my\_first\_variable

'''

name = "Mike" *# Strings*

age = 30 *# Integer*

gpa = 3.5 *# Decimal*

is\_tall = True *# Boolean -> True/False*

name = "John"

print("Your name is " + name)

print("Your name is", name)

### Casting and Converting

print( int(3.14) )

print( float(3) )

print( str(True) )

print( int("50") + int("70") )

### Strings

greeting = "Hello"

*#indexes: 01234*

print( len(greeting) )

print( greeting[0] )

print( greeting[-1] )

print( greeting.find("llo") )

print( greeting.find("z") )

print( greeting[2:] )

print( greeting[2:3] )

### Numbers

print( 2 \* 3 ) *# Basic Arithmetic: +, -, /, \**

print( 2\*\*3 ) *# Basic Arithmetic: +, -, /, \**

print( 10 % 3 ) *# Modulus Op. : returns remainder of 10/3*

print( 1 + 2 \* 3 ) *# order of operations*

print(10 / 3.0) *# int's and doubles*

num = 10

num += 100 *# +=, -=, /=, \*=*

print(num)

++num

print(num)

*# Math module has useful math methods*

import math

print( pow(2, 3) )

print( math.sqrt(144) )

print( round(2.7) )

### User Input

name = input("Enter your name: ")

print("Hello", name + "!")

num1 = int(input("Enter First Num: "))

num2 = int(input("Enter Second Num: "))

print(num1 + num2)

### Lists

lucky\_numbers = [4, 8, "fifteen", 16, 23, 42.0]

*# indexes 0 1 2 3 4 5*

lucky\_numbers[0] = 90

print(lucky\_numbers[0])

print(lucky\_numbers[1])

print(lucky\_numbers[-1])

print(lucky\_numbers[2:])

print(lucky\_numbers[2:4])

print(len(lucky\_numbers))

### 2 Dimensional Lists

numberGrid = [ [1, 2], [3, 4] ]

numberGrid[0][1] = 99

print(numberGrid[0][0])

print(numberGrid[0][1])

### List Functions

friends = []

friends.append("Oscar")

friends.append("Angela")

friends.insert(1, "Kevin")

*# friends.remove("Kevin")*

print( friends )

print( friends.index("Oscar") )

print( friends.count("Angela") )

friends.sort()

print( friends )

friends.clear()

print( friends )

### Tuples

lucky\_numbers = (4, 8, "fifteen", 16, 23, 42.0)

*# indexes 0 1 2 3 4 5*

lucky\_numbers[0] = 90

print(lucky\_numbers[0])

print(lucky\_numbers[1])

print(lucky\_numbers[-1])

print(lucky\_numbers[2:])

print(lucky\_numbers[2:4])

print(len(lucky\_numbers))

### Functions

def add\_numbers(num1, num2=99):

return num1 + num2

sum = add\_numbers(4, 3)

print(sum)

### If Statements

is\_student = False

is\_smart = False

if is\_student and is\_smart:

print("You are a student")

elif is\_student and not(is\_smart):

print("You are not a smart student")

else:

print("You are not a student and not smart")

*# >, <, >=, <=, !=, ==*

if 1 > 3:

print("number omparison was true")

if "dog" == "cat":

print("string omparison was true")

### Dictionaries

test\_grades = {

"Andy" : "B+",

"Stanley" : "C",

"Ryan" : "A",

3 : 95.2

}

print( test\_grades["Andy"] )

print( test\_grades.get("Ryan", "No Student Found") )

print( test\_grades[3] )

### While Loops

index = 1

while index <= 5:

print(index)

index += 1

### For Loops

for index in range(5):

print(index)

lucky\_nums = [4, 8, 15, 16, 23, 42]

for lucky\_num in lucky\_nums:

print(lucky\_num)

for letter in "Giraffe":

print(letter)

### Exception Catching

try:

answer = 10 / int(input("Enter Number: "))

except ZeroDivisionError as e:

print(e)

except:

print("Caught any exception")

### Classes and Objects

class Book:

def \_\_init\_\_(self, title, author):

self.title = title

self.author = author

def read\_book(self):

print("Reading", self.title, "by", self.author)

book1 = Book("Harry Potter", "JK Rowling");

*# book1.title = "Half-Blood Prince"*

print(book1.title)

book1.read\_book()

### Getters and Setters

class Book:

def \_\_init\_\_(self, title, author):

self.title = title;

self.author = author

@property

def title(self):

print("getting title")

return self.\_title

@title.setter

def title(self, value):

print("setting title")

self.\_title = value

@title.deleter

def title(self):

del self.\_title

def read\_book(self):

print("Reading", self.title, "by", self.author)

book1 = Book("Harry Potter", "JK Rowling");

*# book1.title = "Half-Blood Prince"*

print(book1.title)

book1.read\_book()

### Inheritance

class Chef:

def \_\_init\_\_(self, name, age):

self.name = name

self.age = age

def make\_chicken(self):

print("The chef makes chicken")

def make\_salad(self):

print("The chef makes salad")

def make\_special\_dish(self):

print("The chef makes bbq ribs")

class ItalianChef(Chef):

def \_\_init\_\_(self, name, age, countryOfOrigin):

self.countryOfOrigin = countryOfOrigin

super().\_\_init\_\_(name, age)

def make\_pasta(self):

print("The chef makes pasta")

def make\_special\_dish(self):

print("The chef makes chicken parm")

myChef = Chef("Gordon Ramsay", 50)

myChef.make\_chicken()

myItalianChef = ItalianChef("Massimo Bottura", 55, "Italy")

myItalianChef.make\_chicken()

print(myItalianChef.age);