Python Assignment Two

Conor Sosh

# Basics

* What is an expression?
  + Pieces of code that return a value like 4 + 5, or even function calls

* What is a syntax error?
  + Mistyping, or inputting the wrong structure for your code, like not having a “:” after if or while statements, or not putting “ “ around your string variables

* What is PEP8?
  + PEP 8 is the style guide/conventions for python code. It is a document outlining the proper indentation, tabs/spaces, comment style and naming conventions.

* What does a linter do?
  + A linter is a tool used in writing code that can detect errors, style flaws, and syntax bugs that can occur while coding.

* What is the result of this expression: “\*” \* 10
  + \*\*\*\*\*\*\*\*\*\* (10 \*’s)

* What is CPython?
  + CPython is the original python implementation, created in C. It is used to distinguish between later versions of python that were implemented in other/different ways like Jython or PyPy

* How is CPython different from Jython?
  + CPython compiles Python code into bytecode, and interprets the bytecode in an evaluation loop. Jython compiles Python code into Java bytecode, and runs it on the JVM.

* How is CPython different from IronPython?
  + IronPython is a similar difference – it lets you run Python code on the Microsoft CLR

# Primitive Types

* What is a variable?
  + Variables are containers for storing data values. Strings, ints, lists, tuples can all be assigned to a variable.

* What are the primitive built-in types in Python?
  + Integer, Float, String, Boolean

* When should we use “”” (triple quotes) to define strings?
  + To define multi-line strings to prevent a string from stretching too long off of the screen

* Assuming (name = “John Smith”), what does name[1] return?
  + It returns “o”

* What about name[-2]?
  + It returns “t”

* What about name[1:-1]?
  + It returns “ohn Smit”

* How do you get the length of name?
  + print(len(name))

* What are the escape sequences in Python?
  + An escape sequence is a special set of characters all beginning with a \
  + \’ – Represents a single quote
  + \n – Represents a newline
  + \r – Represents a carriage return
  + \t – Represents a tab
  + \b – Represents a backspace
  + \b – Represents a formfeed
  + \ooo – Represents an octal value
  + \xhh – Represents a hex value
  + \\ - Represents a backslash
  + \uxxxx – Represents 16 bits hex value
  + \uxxxxxxxx – Represents 32 bits hex value
* What is the results of f”{2+2} + {10%3}”?
  + 4 + 1

* Given (name = “john smith”), what will name.title() return?
  + It capitalizes the first character of each word in the string
  + John Smith

* What does name.strip() do?
  + It removes the newline character from the string name

* What will name.find(“Smith”) return?
  + It returns the first location of “Smith” in the string
  + In this case: 5

* What will the value of name be after we call name.replace(“j”, “k”)?
  + Name will have the same value – John Smith
  + Strings are immutable, so replace returns a new string with the value kohn smith

* How can we check to see if name contains “John”?
  + If “John” in name:  
     print(“Found it”)

* What are the 3 types of numbers in Python?
  + Integers, Floats, and Complex

# Control Flow

* What is the difference between 10 / 3 and 10 // 3
  + 10 / 3 will return 3.333333 – the single / is used for floating point division
  + 10 // 3 will return 3 – the double / is used for “floor” division to get whole numbers

* What is the result of 10 \*\* 3?
  + This is 10^3 – it equals 1000

* Given (x = 1), what will be the value after we run (x += 2)?
  + x = 3

* How can we round a number?
  + You can use the built in round() function, which will always round up,
  + You can also write a function to truncate a number, like (n \* 1000) / 1000
* What is the result of float(1)?
  + 1.0

* What is the result of bool(“False”)?
  + True
  + Any string is True, except empty strings
  + Any number is True except 0
  + Any list, tuple, set, or dictionary is True, except empty ones

* What are the falsy values in python?
  + Empty strings
  + 0
  + Any empty collections like list, tuples, sets or dictionaries

* What is the result of 10 == “10”?
  + False

* What is the result of “bag” > “apple”?
  + True

* What is the result of not(True or False)?
  + False

* Under what circumstances does the expression 18 <= age < 65 evaluate to True?
  + When age is less than 65, and equal to or greater than 18

* What does range(1, 10, 2) return?
  + 1, 3, 5, 7, 9

* Name 3 iterable objects in Python
  + List, set, dictionary

# Functions

* What is the difference between a parameter and an argument?
  + Parameters are the variables required from by a function for it to actually function. For example a function add(a, b) – a and b are the parameters
  + Arguments are the actual values that are passed as parameters. In our example above, 4 and 6 would be valid arguments

* All functions in Python by default return what?
  + All functions by default return None

* What are keyword arguments and when should we use them?
  + Keyword arguments are a way of defining parameters in a different order than what the function originally specifies.
  + We typically use keyword arguments to make our function calls more readable or easier to understand, or when we can leave out certain arguments that have acceptable default values.

* How can we make a parameter of a function optional?
  + Set a default value for the parameter when we create the function like: add(x = 1, y)

* What happens when we prefix a parameter with asterisk?
  + It allows us to pass any number of arguments as parameters to that function

* What about two asterisks?
  + Two asterisks lets us define all keyword arguments

* What is scope?
  + Scope is the area where a given variable is considered valid, typically in a given function or class.
* What is the difference between local variables and global variables?
  + Local variables can only be used in the function/class they were defined in, where as global variables can be called and used at any point and time.

* Why is using the global statement a bad practice?
  + It breaks the typical rules and expectations of traditional OOP, and can give access to parts of your program that should otherwise be hidden.

# Coding Exercises

* Write a function that returns the maximum of two numbers

def find\_max(a, b):

if(a > b):

print(a)

return a

else:

print(b)

return b

* Write a function called fizz\_buzz that takes a number
  + If the number is divisible by 3, it should return “Fizz”
  + If it is divisible by 5, it should return “Buzz”
  + If it is divisible by both 3 and 5 it should return “FizzBuzz”
  + Otherwise, it should return the same number

def fizz\_buzz(num):

if(num % 15 == 0):

print("FizzBuzz")

return("FizzBuzz")

elif(num % 3 == 0):

print("Fizz")

return("Fizz")

elif(num % 5 == 0):

print("Buzz")

return("Buzz")

else:

print(num)

return(num)

* Write a function for checking the speed of drivers. This function should have one parameter: speed.
  + If speed is less than 70, it should print “Ok”
  + Otherwise, for every 5km over the speed limit (70), it should give the driver one demerit point, and print out the total number of demerit points.
  + If the driver gets more than 12 points, the function should print “License suspended”

def give\_demerits(speed):

if(speed < 70):

print("Ok")

else:

points = ((speed - 70) // 5)

print("Points: " + str(points))

if points > 12:

print ("License suspended")

* Write a function called showNumbers that takes a parameter limit. It should print out all the numbers between 0 and limit with a label to identify the even and odd numbers

def showNumbers(limit):

for num in range(limit + 1):

print(num, end = " ")

if (num % 2 == 0):

print("EVEN")

else:

print("ODD")

* Write a function that returns the sum of multiples of 3 and 5 between 0 and limit (parameter)

def sum\_multiples(limit):

sum = 0

for num in range(limit + 1):

if (num % 3 == 0) or (num % 5 == 0):

sum += num

print(sum)

return(sum)

* Write a function called show\_stars(rows) that prints out a number of asterisks. The number of asterisks in each row should be equal to the row number

def show\_stars(rows):

for i in range(rows + 1):

print()

for j in range(i):

print("\*", end = "")

* Write a function that prints all the prime numbers between 0 and limit where limit is a parameter

def print\_primes(limit):

for num in range(1, limit+1):

if (num > 1):

for i in range(2, num):

if (num % i) == 0:

break

else:

print(num)