Chapter 2: Writing Simple Programs

Jan 9, 2020

Today’s Outline

- Review basic programming: IPO, print( ), input( ) and eval( )
- Functions
- Variables and variable assignments
- For loops

Input, Process, Output (IPO)

A good idea for designing simple programs is to think of the input, process, and output that you will need to build the program.

Greeting Program

- Use Python to create a personalized greeting for a new user by asking their name

```
#input
name = input ("Please enter your first name: ")
#process
greeting = "Hello " + name
#output
print(greeting)
```

print( )

```
#process
greeting = "Hello " + name
#output
print(greeting)
```

We could also use one line to get the same output:

```
print("Hello" , name)
```

print( ) statement

```
#print onto a new line
print ("I would like to \ncreate a new line")
```

```
#print on the same line
print(3, 4, end=" "),
print(3 + 4)
```

Shakespeare Problem

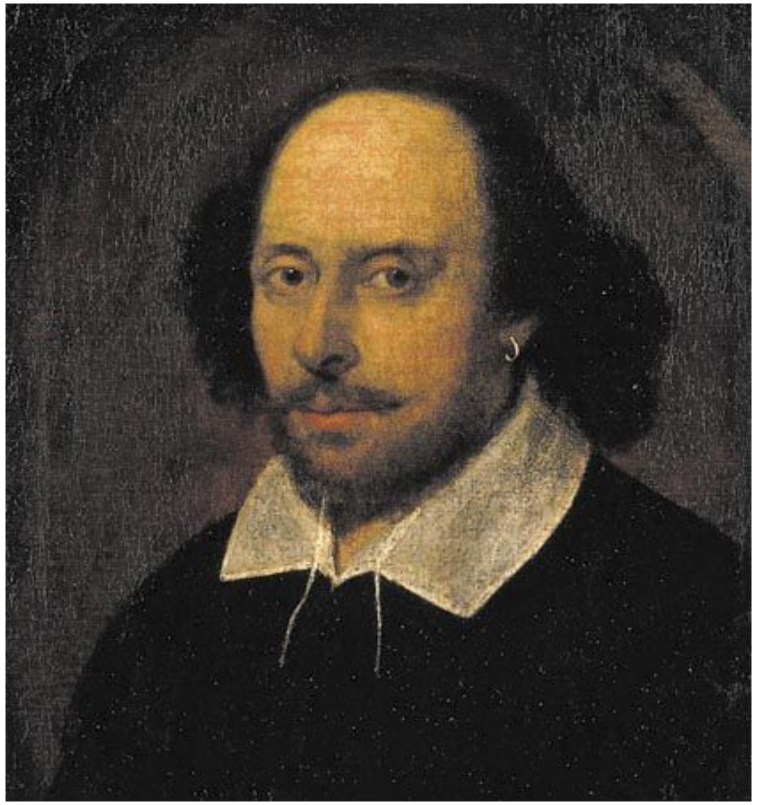
Print the following text to the screen so that it is formatted into four lines.

Shall I compare thee to a summer’s day?

Thou art more lovely and more temperate:

Rough winds do shake the darling buds of May,

And summer’s lease hath all too short a date:



eval ()

- The eval() function will evaluate any expression that is entered into it.

```
ans = eval (input ("Enter an expression: ") )
print(ans)
```

*** What’s the problem with this?

Code Injection Attack

- Theoretically, a user could enter an executable instruction that could be used to infiltrate and/or damage the programmer’s computer

- In Python, functions including eval( ), exec( ), compile ( ) are vulnerable to code injection attacks

- Safe example: __import__('os').getcwd()

- Unsafe example: __import__('os').system('rm -rf/file_path_to_maliciously_delete')

Code Injection Attack Mitigation

- Check the user input before using a function like eval ( )

- Possible ideas:
    - Check that all characters belong to a “whitelist” of safe character values (whitelist only includes numbers and operations)
    - Remove double underscores __ in the input
    - Remove / from the input
    - Use ast.literal_eval( ) instead of eval ( )

Functions

print ( ) , eval ( ) , and input ( ) are all built-in functions in Python.

A function refers to another piece of code.

A function is “called” by using the function name and double brackets.

function_name ( )

function_name (parameter)

function_name (parameter_1, parameter_2)

Functions

A function can also output information to a variable:

```
function_output = function_name (parameter)

function_output1, function_output2 = function_name (parameter)
```

Function examples

```
print ("Hello world ")
```

function name parameter

```
username = input ("Please enter your first name: ")
```

function output function name parameter

How to get help with built_in functions

function_name.__doc__

help (function_name)

Google

Making custom functions

You can make a custom function in Python using def as follows:

```
def custom_function (parameters):
    function code line
    function code line
    return function_output
```

Custom Function Example 1

```
#Custom Function to greet a user

def greet (userName):
    greeting = "Hello " + userName
    print (greeting)

#Use the function in a program
name = input ("Please enter your first name: ")
greet (name)
```

Custom Function Example 2

```
#Custom Function to convert a temperature

def tempConvert (celsius):
    fahrenheit = celsius*9/5 + 32
    return fahrenheit

#Use the custom function in a program
temp_c = eval (input ("Enter a temperature in Celsius: "))
temp_f = tempConvert(temp_c)
print ("The temperature in Fahrenheit is: ", temp_f)
```

The main ( ) function

In Python, it is common practice to put the main portion of your code in a function called main(). If we do this, we can run the main program by using the command: main()

Note: For simple programs, it is not actually necessary to put everything in a main () function.

main ( ) example

```
def main():
    name = input ("Please enter your first name: ")

    greeting = "Hello " + name

    print(greeting)


#Call main () to tell Python to run the entire program

main ()
```

main ( ) example 2

```
def tempConvert (celsius):
    fahrenheit = celsius*9/5 + 32
    return fahrenheit


def main():
    temp_c = eval (input ("Enter a temperature in Celsius: "))
    temp_f = tempConvert(temp_c)
    print ("The temperature in Fahrenheit is: ", temp_f)


main()
```

Speed Limit Problem

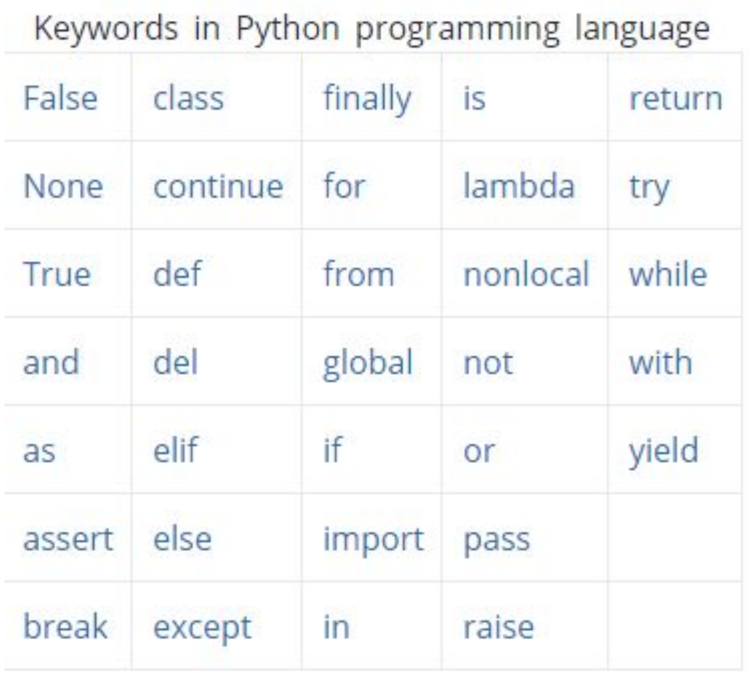
Design a python function that can convert a speed in miles per hour to km per hour. Hint: 1 mile = 1.6 km.

Variables

We already learned that variables can be numbers (x = 3) or “strings” of characters (greeting = "Hello World!")

Variable Name Rules

- Variable names are case-sensitive
- Variable names must begin with a letter or an underscore (_)
- Variables names should not be Python keywords

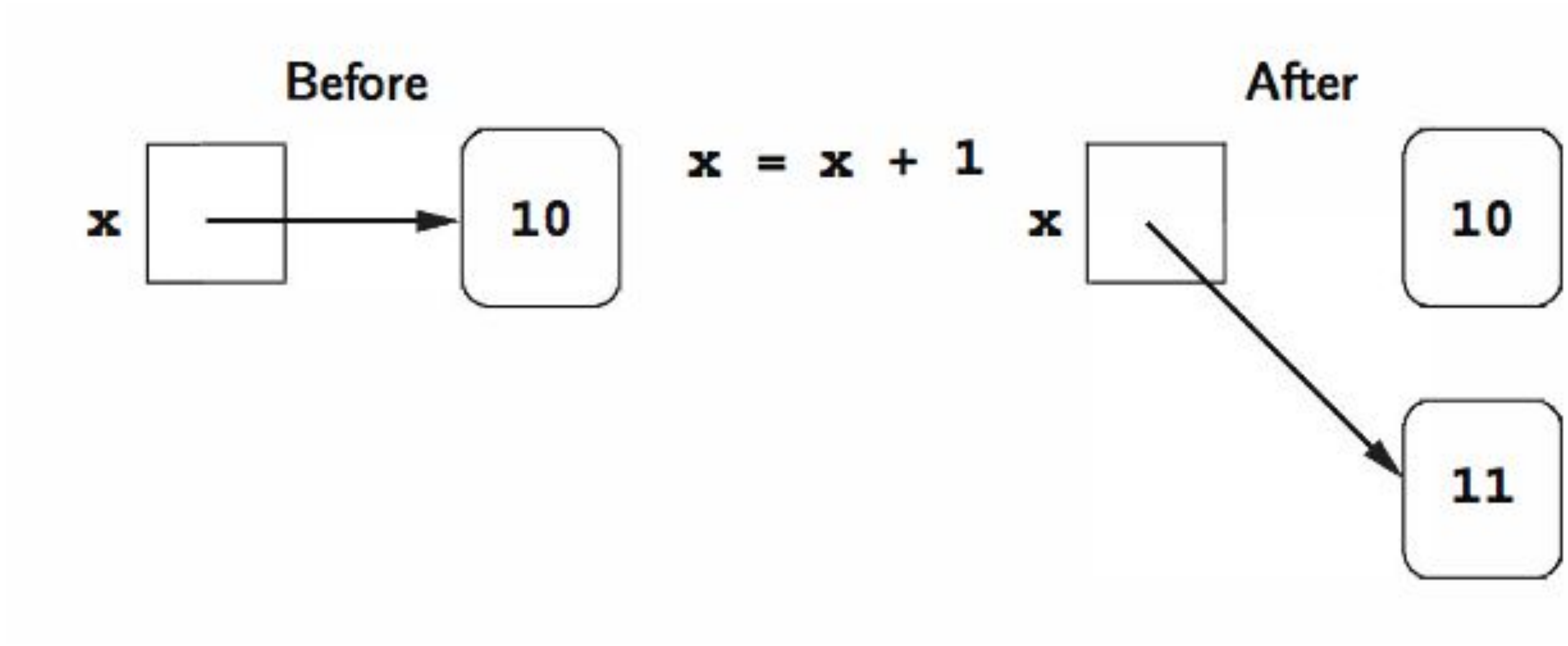


Keywords in Python programming language

False	class	finally	is	return
None	continue	for	lambda	try
True	def	from	nonlocal	while
and	del	global	not	with
as	elif	if	or	yield
assert	else	import	pass	
break	except	in	raise	

https://www.programiz.com/python-programming/keyword-list

Variables in Python



Simultaneous Variable Assignments

```
x = 3
y = 4
sum, diff = x+y, x-y
print(sum)
print(diff)
```

What will the output be?

What if we want to swap variables x and y?

```
num_eggs = 3
num_toast = 10
```

Swap the variables so that we have 10 eggs and 3 toast.

What if we want to swap variables x and y?

```
#Method One
num_eggs = 3
num_toast = 10

old_num_eggs = num_eggs
num_eggs = num_toast
num_toast = old_num_eggs
```

What if we want to swap variables x and y?

```
#Method One
num_eggs = 3
num_toast = 10

old_num_eggs = num_eggs
num_eggs = num_toast
num_toast = old_num_eggs
```

```
#Method Two
num_eggs = 3
num_toast = 10

num_eggs, num_toast = num_toast, num_eggs
```

Simultaneous Variable Assignments

```
num_toast, num_eggs = eval(input("Enter # of slices of toast followed by # of eggs: "))

print("There are", num_toast, "slices of toast and", num_eggs, "eggs.")
```

Change Calculator Problem

Design a program that prompts the user to enter the number of quarters they own followed by the number of dimes they own, and outputs the total amount of money the user owns.

Change Calculator Problem

Design a program that prompts the user to enter the number of quarters they own followed by the number of dimes they own, and outputs the total amount of money (in dollars) that the user owns.

Input: # of dimes, # of quarters

Process: (# of dimes)*0.1 + (# of quarters)*0.25

Output: Total money in $

For loops

For loops are used to repeat a part of a program a specified number of times.

```
for i in range(10):
    [code to perform 10 times]
```

*For coding in Python, remember that the lines of code that you want to repeat must be indented underneath your for loop statement.

Equivalent Codes

```
for i in range(10):
    print ("Hello")
```

=

```
print ("Hello")
print ("Hello")
print ("Hello")
print ("Hello")
print ("Hello")
print ("Hello")
print ("Hello")
print ("Hello")
print ("Hello")
print ("Hello")
```

for loop

Example:

```
for i in range(10):
  print(i)
```

For loops with lists

A **list** is a sequence of values. To make a list, we can place the values in square brackets:

```
lottery_numbers = [48, 3, 15, 7, 23, 55, 10]

fruits = ["strawberry", "cherry", "grape"]
```

For loops with lists

For loops can be used to repeat code for any specified values by using a list.

```
lottery_numbers = [48, 3, 15, 7, 23, 55, 10]
for num in lottery_numbers:
  print (num)


for i in [1, 3, 5, 6, 8]:
  print(i)


for fruits in ["strawberry", "cherry", "grape"]:
  print (fruits)
```

For loops with range (   )

```
for i in range (start, end, spacing):
    [code to execute]
```

```
for i in range (0, 10, 2):
 print(i)
```

4: For loop Problems:

1. Write a program that uses a for loop to print the sentence “Hello, Python!” twenty times.

2. Write a program that prints the square of even integers from 0 to 20.

Chaos Example Code

```
# File: chaos. py
# A simple program illustrating chaotic behavior.
print("This program illustrates a chaotic function")
x = eval(input("Enter a number between 0 and 1: "))
for i in range(10):
    x = 3.9 * x * (1 - x)
    print(x)
```

Chaos Example Code

1. Modify the Chaos code to print out 20 values instead of 10.

2. Modify the Chaos code so that the number of values to print is determined by the user.

Interest Problem

I am going to invest $5000 in a GIC with a compounding interest rate of 2.55%. Use a for ( ) loop to calculate the total amount of money I will have after 5 years.

Interest Problem - Part 2

Modify the program so that it can calculate the investment amount after 5 years for any principal investment, and any interest rate specified by the user.

Pizza Problem

Python Pizzaria sells 3 sizes of pizza as follows:

Small (6 inch radius) - $10

Medium (8 inch radius) $15

Large (12 inch radius) $20

Develop a custom function that will calculate the price per square inch of a pizza given its radius and price. Which pizza has the cheapest unit price?

Software Development Process

1. **Analyze** the problem. Think about how the problem could be executed using a computer.
2. **Specifications.** Think about **what** the program will do
   a. What are the **inputs and outputs**? What format will they have?
   b. How do they relate to each other?
3. **Design.** What **process** will be used the solve the problem? What algorithm will be used?
4. **Implement**. Translate the program into a computer language (i.e. Python)
5. **Test.** Does the program work? For all cases?
6. **Maintain.** Continue developing the program for the needs of users.

Practice Problem 1 (at home)

Develop a function that can print the first n elements of the Fibonacci sequence.

Example:

print_fibonacci (10)

Computer will output to the screen:

```
 0 1 1 2 3 5 8 13 21 34
```

Practice Problem 2 (at home)

Create a program that computes and prints a table of Celsius temperatures and the Fahrenheit equivalents every 10 degrees from 0° C to 100°C.