IASP505

Foundations Coding in Python

Dr. John Yoon

Branching Statement

Simple Condition

- Condition takes an argument(s) to compare, and returns TRUE or FALSE

Getting data

if

otherwise

```
data = input("Enter your age in integer")

if (data >= 21):
  print("You are already an adult.")

else:
  print ("You are still a child.")
```

Operator

Any error?

- What is the type of an `input()` return data?
- How to covert a number in string to an integer?

```
data = input("Enter your age in integer")

if (data >= 21):
  print("You are already an adult.")

else:
  print ("You are still a child.")
```

```
data = int (data)
```

a built-in function int() takes a number in String and return the number in integer

Connecting Conditions

- Condition can connect one or more conditions to compare, and returns TRUE or FALSE

input data

if

otherwise

```
state = input("Enter your residential state\n")

if state == "New York" :
  print("You are in ", state)

else:
  print ("You are not in New York, but in ", state)
```

Comparison Operator

Connecting Conditions

- Condition expressed over both state and age

Connecting Conditions

- Condition can connect one or more conditions to compare, and returns TRUE or FALSE

Logical Connector

Two input data

if

otherwise

```
state = input("Enter your residential state\n")
age = input("Enter your age in integer\n")

if (state == "New York") and (age >= 21):
  print("You are already an adult in ", state)

else:
  print ("You are still a child.")
```

```
data = int (data)
```

Comparison Operator

Conditional Statement

```
if <conditions> :
          <then part actions>
else <other actions>
```

```
if <conditions1> :
          <then part actions>
elif <another condition> :
          <action for another>
elif <yet another> :
          <continue this pattern>
else <other actions>
```

Composite Conditions:

```
if <cond1> and <cond2>:
```

Series actions:

```
if (   ):
    <action 1>
    <action 2>

# here is outside the
# condition statement
```

OK as far as a proper indentation is made. No blocking symbol, e.g., { }, is needed

Try More

Write tempConv.py

Try to convert temperature in Celsius to Fahrenheit, or vice versa.

Check out the following webpage:

https://www.thoughtco.com/temperature-conversion-formulas-609324

Write two functions:

convF2C(), Convert F to C

convC2F(), Convert C to F

Suppose that your Python code, tempConv.py takes user input as a concatenation of integer and a temperature indicator, F or C. For example, it runs as follows:

Enter temperature followed by C or F:

57 F

57 F = 13.88888888888889 C

More

Consider the following sample run

Enter temperature followed by C or F:

57 F

57 F = 13.88888888888889 C

The result tempConf.py program displays can be improved:

57 F = 13.89° C

Please improve your program as above.

Hint:

- Use the Unicode \u00b0 for temperature degree symbol
- Use the function .format

Iteration

Repetition Statement

- Why repetitive operations are needed?
- Examples
  - The same grading rule needs to be applied to each of the class students
  - Stock trading rules should be applied at each every single trading
- Cyber Examples
  - Forensic search rules are applied to each single line of the given webpage, given emails, a file.
  - Inspection rules are applied to each every incoming network packets

Iterative Statement

- How can the same code segment be performed iteratively?
- It will be based on
  - For a number of times
    - Ex) Investigate first 15 suspects
  - For each in a list
    - Ex) Investigate each in a give list of suspects
  - For each element in a file
    - Ex) Search a keyword in each line of a file
  - While some condition is met
    - Ex) Search a keyword while network packets are incoming

Iterative Statement

Match one to one

- For a number of times
- For each in a list
- For each element in a file
- While some condition is met

```
for each in range(1,5):
    print(each)
```

Iterative Statement

Match one to one

- For a number of times
- For each in a list
- For each element in a file
- While some condition is met

```
for each in range(1,5):
    print(each)

with open(userFile) as f:
    for each in f:
        print (each)
```

Iterative Statement

Match one to one

- For a number of times
- For each in a list
- For each element in a file
- While some condition is met

```
for each in range(1,5):
    print(each)

with open(userFile) as f:
   for each in f:
       print (each)
```

Iterative Statement

Match one to one

- For a number of times
- For each in a list
- For each element in a file
- While some condition is met

```
for each in range(1,5):
    print(each)

with open(userFile) as f:
   for each in f:
       print (each)

while True:
   print(x)
   if x > 100:
      break
```

Example

- Consider 10 iterations

Answer Display

1) Add 1 to 10

55

2) Add 1 to 10

- Display the formula

1+2+3+4+5+6+7+8+9+10 = 55

3) Add 1 to 10

- Display each step

1=1
1+2=3
1+2+3=6
1+2+3+4=10
1+2+3+4+5=15
1+2+3+4+5+6=21
1+2+3+4+5+6+7=28
1+2+3+4+5+6+7+8=36
1+2+3+4+5+6+7+8+9=45
1+2+3+4+5+6+7+8+9+10 = 55

Reading a File

- Use a built-in function open()
  - open(<file_name>, <mode>)

Mode	Description
r	Open for reading plain text
w	Open for writing plain text
a	Open an existing file for appending plain text
rb	Open for reading binary data
wb	Open for writing binary data

Ex) Reading a File

- File, tempFile.dat

```
57 F Dobbs Ferry
43 C London
```

- Read the file
  - Print it as a whole
  - Print line by line
- Returns

Ex) Reading a File

- File, tempFile.dat

```
57 F Dobbs Ferry
43 C London
```

- Read the file
  - Print it as a whole
  - Print line by line

```
g = open("tempFile.dat")
print(g)
for line in g:
    print(line)
```

- Returns

Ex) Reading a File

- File, tempFile.dat

```
57 F Dobbs Ferry
43 C London
```

- Read the file
  - Print it as a whole
  - Print line by line

```
g = open("tempFile.dat")
print(g)
for line in g:
    print(line)
```

- Returns

```
<_io.TextIOWrapper name='tempFile.dat' mode='r' encoding='cp1252'>
57 F Dobbs Ferry

43 C London
```

This is an object wrapper data in response to print(g), which is not readable. So we need to read the file line-by-line

Defining Functions

Functions in Programming

- Known flat coding, now create it in a function
- Why functions?

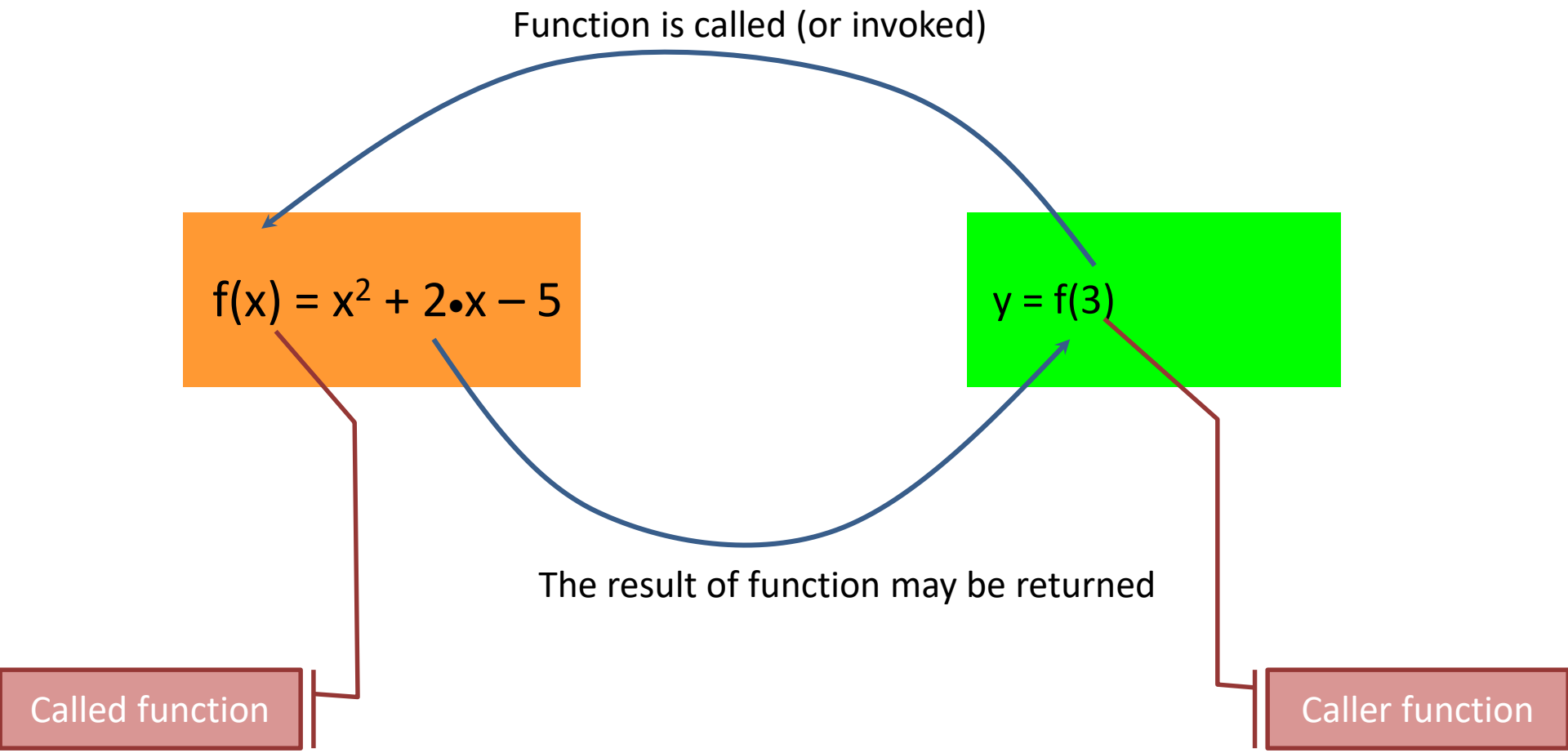
$y = 3^2 + 2 \bullet 3 - 5$
$y = 2^2 + 2 \bullet 2 - 5$
$y = (-2)^2 + 2 \bullet (-2) - 5$

Instead of repeating the same procedure, consider the following:

$$f(x) = x^2 + 2 \bullet x - 5$$

$y = f(3)$
$y = f(2)$
$y = f(-2)$

Function Calls



Function Calls in Python

- Convert a fahrenheit temperature to celsius

```
f(x) = (x-32)*5/9
```

- Coding

```
uInput = input("Enter temperature in F:\n")
# input() is a builtin function which takes
user input and returns it in String.

f = int(uInput)
# due to the math computation below, the
user input String should be in integer.
# int() is a builtin function which converts
an argument into an integer.

c = (f -32)*5.0/9.0

print(f, "in F is ", c , "in C.")
```

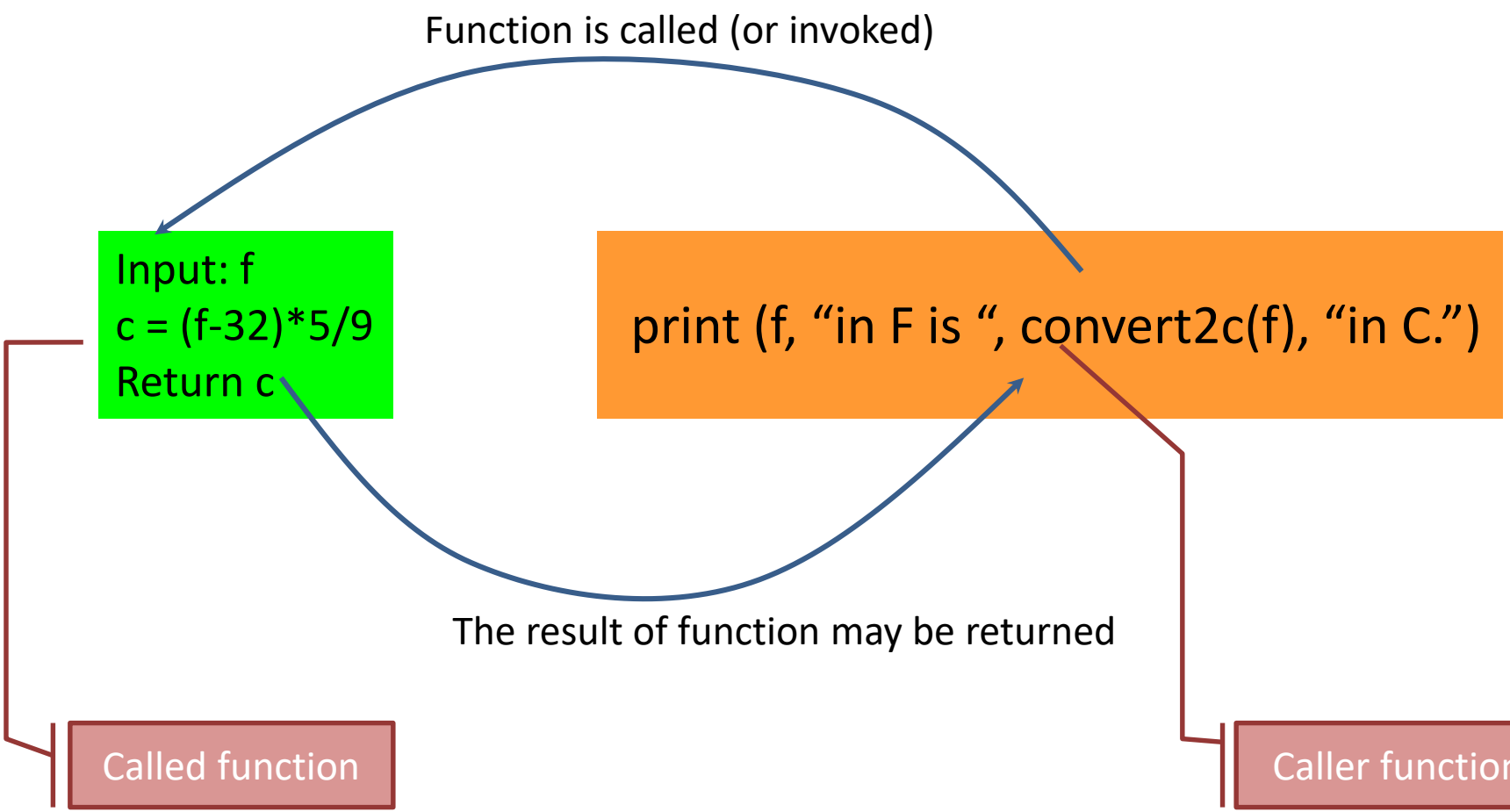
Define a function which takes _____ and returns _____.

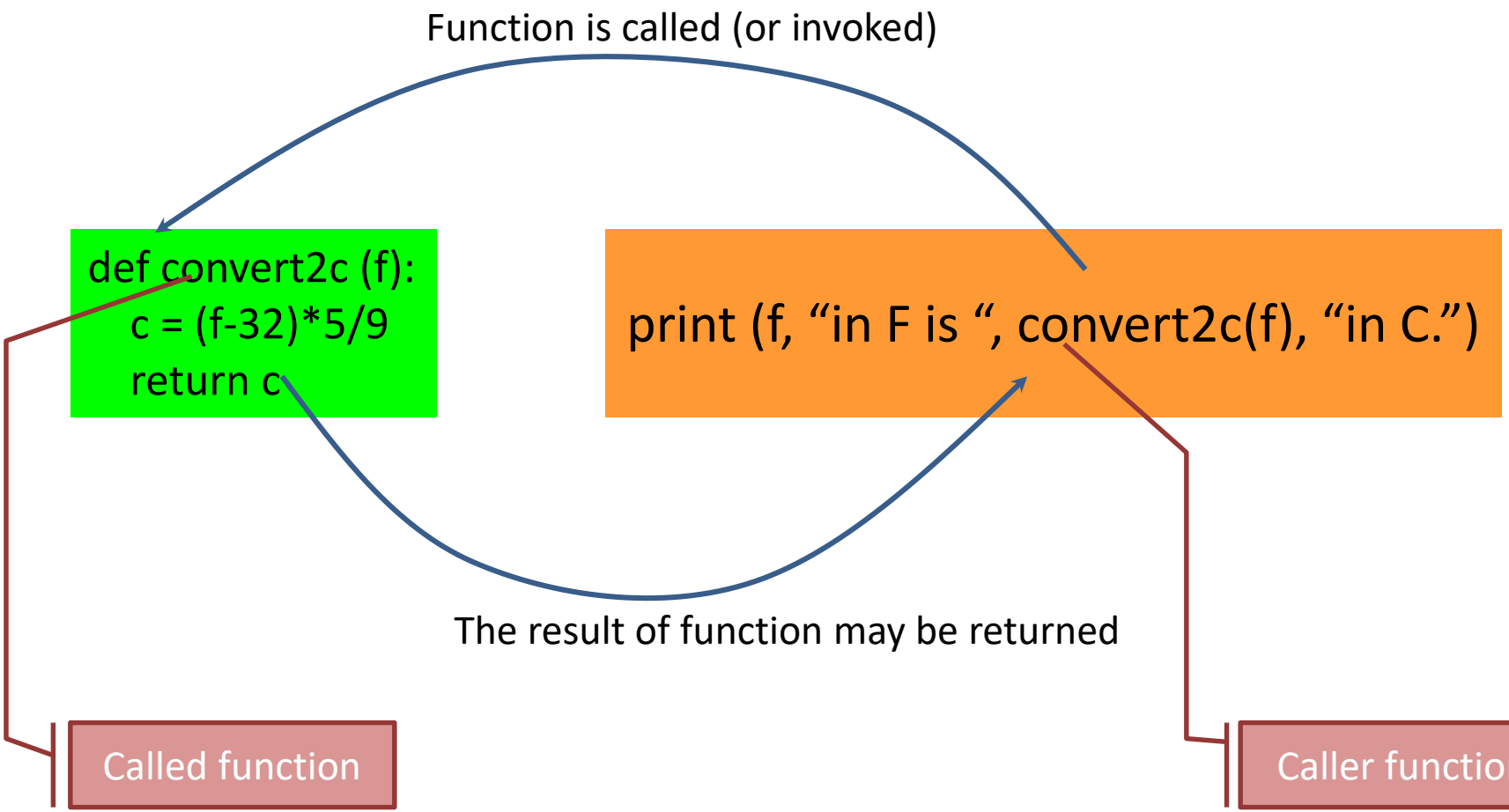
```
# int() is a builtin function which converts an
argument into an integer.

c = (f -32)*5.0/9.0

print(f, "in F is ", c , "in C.")
```

Define a function which takes _____ and returns _____.




Function is called (or invoked)
def convert2c (f):
c = (f-32)*5/9
return c
print (f, “in F is “, convert2c(f), “in C.”)
The result of function may be returned
Called function
Caller function

How does it work?

- Known your coding, try to define another yet similar function, convert2f(c) and call the function in your coding.

Exercise (function)

```
from tkinter import *

root = Tk() # constructor: an object of Tk is constructed and labeled by "root"
root.title("Mercy GUI #1")
can = Canvas(root, width=700, height=500, bg="#aa88ff")
# constructor of Canvas: construct an object of Canvas
can.create_rectangle(10,10, 100,100, fill="#ff7711")
# x,y coordinators of the upper left cornder and lower right
can.create_rectangle(100,100, 500,250)
can.create_oval(70,70, 130,130)
can.create_oval(320,220, 380,280)
can.pack()
```

- Rewrite the above to define a function and call the function
  - Therefore, by calling, the above component is drawn on the canvas.
  - By sending different sets of parameters, different size of the components are drawn differently on different sized canvases.

HW5

- Consider the list of Computer Science areas, check out from http://csrankings.org/#/index?all.
  - Define a dictionary of CS areas: key (i.e., AI, System, Theory, Interdisciplinary) is a top area, and the value to a key is a list of subareas.
    - For example, AI has the subareas, artificial intel, computer vision, machine learning, etc
  - Each subarea has a list of the best publication
    - For example, artificial intelligence has AAAI and IJCAI
  - Each publication has a web URL
    - For example, AAAI is linking https://dblp.org/db/conf/aaai/
- Write a Python program running a loop to
  - Display all together hierarchically
  - Ask users to enter any terms of top areas or sub areas of computer sciences
- Accordingly, the program can do
  - Open the web site
  - If the user input is from the top area list, visit all of its subareas and open web sites on webbrowser
  - If the user input is from the subarea, open its web sites
- Hints:
  - You may want three functions, displayCSareas(), openAllWeb(area), openSubWeb(area). Each function may have one or two for loops.
  - First call displayCSareas()
  - Then, get user input
  - Check whether it is in the top area list or the sub areas.
  - Depending on it, either openAllWeb() or openSubWeb() is called.
  - Make sure that if a user input is not in any lists, then print the S“unavailable area!” message.