CS 171 - Lab 6

Professor Mark W. Boady and Professor Adelaida Medlock

Content by Professor Lisa Olivieri of Chestnut Hill College

Detailed instructions to the lab assignment are found in the following pages.

- Complete all the exercises and type your answers in the space provided.

What to submit:

- Lab sheet in PDF.

Submission must be done via Gradescope

- Please make sure you have marked the questions on the correct pages of the PDF and added any teammates to the submission
- We only accept submissions via Gradescope.

Student Name: Tony Kabilan Okeke

User ID (abc123): tko35

Possible Points: 88

Your score out of 88:

Lab Grade on 100% scale:

Graded By (TA Signature):

Question 1: 1 point

FYI: Predefined/built-in functions: segments of code already included in Python. `print()`, `round()`, `abs()`,`pow()`,`int()` are examples.
Arguments: The information that a function needs to work. Arguments are sent to the function between the parentheses `()`.
To use a function, call the function. `input("Enter your name")` is a call to the `input` function sending the string `"Enter your name"` as an argument.

Closely examine the Python program below.

```
#Get Input
name = input("Enter your name: ")
#Print Results
print("Name:", name)
```

(a) (1 point) Circle / highlight calls to predefined functions in the program above.

Question 2: 8 points

Enter and execute the following Python program.

```
1  #Sample functions
2  print(abs(-4.67) )
3  print(pow(5, 3) )
4  print(pow(49, 0.5) )
5  print( int (34.8) )
6  print(round(6.9) )
7
8  import random
9  print(random.randint(1 ,100) )
```

(a) (1 point) What is printed by Line 2? 4.67

(b) (1point) What is printed by Line 3? 125.0

(c) (1 point) What is printed by Line 4? 7.0

(d) (1 point) What is printed by Line 5? 34

(e) (1 point) What is printed by Line 6? 7

(f) (1 point) What is printed by Line 9? 95 (A random integer between 1 and 100)

(g) (2 points) What is the difference between the **`round`**`()` function and the **`int`**`()` function?

The round() function rounds the argument up or down (depending on its value) to the nearest integer.
Th int() function returns only the integer part of its argument (e.g., 34 from 34.8)

Question 3: 5 points

What value is returned by each of the following function calls?

(a) (1 point) `abs(4.5)` 4.5

(b) (1 point) `int("678")` 618

(c) (1 point) `round(-5.6)` -6

(d) (1 point) `random.randint(4, 10)` 6 (A random integer between 4 and 10)

(e) (1 point) Is `import random` required to run `random.randint(4, 10)`?

- ◯ Yes
- ◯ No

Question 4: 1 point

Circle / highlight the **argument** in the call to the built-in function:

```
number = 45.78
answer = round(number)
```

Question 5: 1 point

`answer = pow(4, 3)`. What is/are the argument(s) in this code?

- ◯ 4
- ◯ 3
- ◯ 4 and 3
- ◯ 12

Question 6: 2 points

If a function contains more than one argument, do you think the order of the arguments makes a difference? Explain your answer with an example.

Yes, the order of arguments matters. For example, the function math.pow(a, b) returns the result of a^b. The output produced by math.pow(5,4) and math.pow(4,5) would therefore be different.

Question 7: 6 points

Execute the following lines of code:

```
import math
x = 4.7
y = 5.3
z = -4.8
a = -3.2
print(math.ceil(x))
print(math.ceil(y))
print(math.ceil(z))
print(math.ceil(a))
print(math.floor(x))
print(math.floor(y))
print(math.floor(z))
print(math.floor(a))
```

(a) (2 points) Explain the purpose of the `ceil()` function

The ceil() function rounds its argument up to the nearest integer.

(b) (2 points) Explain the purpose of the `floor()` function.

The floor() function rounds its argument down to the nearest integer.

(c) (2 points) Why are the calls to the `floor()` and `ceil()` functions preceded by "`math`."?

The ceil() and floor() functions are part of the math module .

FYI:
A **function** is a segment of code that performs a single task.
A **function definition** is the segment of code that tells the program what to do when the function is executed. The first line of a function definition is known as the **function header**

Question 8: 6 points

Review the following Code.

```
1.  #Description: This program uses a function to print a message
2.
3.  #Function Definition
4.  def printMessage ():
5.      print("Welcome to Python.")
6.      print("Learn the power of functions!")
7.
8.  #Function Definition
9.  def main() :
10.     print("Hello Programmer!")
11.     #Function Call
12.     printMessage ()
13.
14.     #Function Call
15.     main()
```

(a) (1 point) What Python keyword is used to indicate that a code segment is a function definition?
def

(b) (1 point) What are the two function headers in the Python code?
def printMessage ():
def main():

(c) (1 point) The name of the function is in the function header. What are the names of the two functions?
printMessage()
main()

(d) (1 point) Enter and execute the Python program. What is the output?
The program prints the following text in the command window:
Hello Programmer!
Welcome to Python.
Learn the power of functions!

(e) (2 points) What line of code would you add to the program to print the last two lines twice? Where would you add the code?
I would add a second call to the printMessage() function within the definition of main() (i.e., on line 13 within the indented block).

Question 9: 14 points

Examine the following code.

```
#Description: This programuses functions to calculate
# the area of a circle, given the radius

import math

def calculateArea (radius):
    area = math.pi * radius ** 2
    print("A circle of Radius %d has area %.2f"  %(radius, area))

def calculateDiameter(radius):
    diameter = radius / 2
    print ("A circle of radius %d has an area %.2f" % (radius, diameter))


def main() :
    radius = int(input("Enter the radius: "))
    calculateArea(radius)
    calculateArea(6)
    calculateDiameter(radius)

#### Call to main ####
main()
```

(a) (2 points) Label the **function definitions (in blue)** and the **function calls (in green)** in the above code.

(b) (2 points) The function call and the function definition for *calculateArea* each include a variable within the parentheses. The variable in the function call is known as an **argument**. The variable in the function definition is called a **parameter**. What is the parameter in the function definition? What is its purpose?

The parameter is a special placeholder that receives the value of an argument when a function is called. It acts as a variable containing the value of the argument within the function's namespace.

(c) (2 points) In this example the parameter in the *function definition* and the argument in the *function call* have the same name. Is this required?

No, the argument in the function call does not have to possess the same name as the parameter.

(d) (2 points) Enter and execute the program. Verify your answer to question (c) by changing the variable name in the main function from **radius** to **number**. Do not change the parameter variable name in the function definition. Does the program still work?

Yes, the program still runs and produces the same output.

(e) (2 points) Write a line of code that calls the *calculateArea* function and sends the value **6** as the argument. Add the line of code to the main program and execute it to be sure it works properly.

(f) (2 points) Add another function to the program that calculates and prints the **diameter** of a circle, given the **radius** as the parameter. Place the function definition above call to the main function of the program. Write the function below.

```
def calculateDiameter(radius):
    diameter = radius / 2
    Print ("A circle of radius %d has an area %.2f" % (radius, diameter))
```

(g) (2 points) Add another line of code to the main function of the program to call the function that was created in part (f). Send the radius entered by the user as the argument to the function.

Question 10: 5 points

Write a function that draws a frog. Call the function to be sure it works. Put your Python code for this answer.

```
  @..@
 (----)
( >--< )
^^ __ ^^
```

```
In [4]: def showFrog():
   ...:     print("  @..@  \n (----)\n( >--< )\n^^ __ ^^")

In [5]: showFrog()
  @..@
 (----)
( >--< )
^^ __ ^^
```

Question 11: 5 points

Add a **for** loop to call the function multiple times and print the following. Put your Python code for this answer.

Frog 1

```
  @..@
 (----)
( >--< )
^^ __ ^^
```

Frog 2

```
  @..@
 (----)
( >--< )
^^ __ ^^
```

Frog 3

```
  @..@
 (----)
( >--< )
^^ __ ^^
```

Frog 4

```
  @..@
 (----)
( >--< )
^^ __ ^^
```

```
In [6]: for i in range(4):
   ...:     print(f"Frog {i+1}")
   ...:     showFrog()
Frog 1
  @..@
 (----)
( >--< )
^^ __ ^^
Frog 2
  @..@
 (----)
( >--< )
^^ __ ^^
Frog 3
  @..@
 (----)
( >--< )
^^ __ ^^
Frog 4
  @..@
 (----)
( >--< )
^^ __ ^^
```

FYI: So far, the functions you have created print the results within the function. They do not send back any information to the original calling code. Functions that do not send back information are known as **void functions**. Functions often send back or return a result and are known as **value returning functions**.

Question 12: 10 points

Review the following code.

```
# This program prompts the user for two numbers,
# calls a function to determine the smaller number
# and prints the smaller number that is returned from the function
def getSmaller (num1, num2):
     if num1 < num2:
          smaller = num1
     else :
          smaller = num2
     return smaller

def main() :
     userInput1 = int (input("Enter a number: ") )
     userInput2 = int (input("Enter a second number: ") )
     smallerNumber = getSmaller ( userInput1, userInput2 )
     print("The smaller of the two numbers is", smallerNumber)

##Call Main
 main()
```

(a) (2 points) What is the new **keyword** used in the function definition?
return

(b) (2 points) What do you think the keyword tells the program to do?
The return keyword specifies what information the function should send back to the original calling code.

(c) (2 points) Circle/highlight the line of code from the program with the **function call** to **`getSmaller`**.

(d) (2 points) In a **void function**, the **function call** is on a line by itself. Why is this **function call** placed on the right-hand-side of an **assignment statement**?
The getSmaller() functions returns an output which needs to be stored in a variable. Therefore it is placed on the right side of the assignment statement.

(e) (2 points) What are the arguments used for the function call?
userInput1
userInput2

Question 13: 8 points

Examine the following code.

```
import math

def getQuadratic (a, b):
     square = a ** 2 + b **2
     squareRoot = math.sqrt ( square )
     return squareRoot

def main() :
     print("The square root of the sum of the squares of 3 and 4 is:", getQuadratic(3, 4))

##Call Main
main()
```

(a) (2 points) What does the program do?

The program computes the square root of the sum of squares of 3 and 4, and prints that out in the console as shown below:

The square root of the sum of the squares of 3 and 4 is: 5.0

(b) (2 points) Circle/highlight the function call.

(c) (2 points) Is the function a void function or a value returning function?

It is a value returning function

(d) (2 points) Why is the `import` statement needed in this program?

The sqrt function is part of the math package and can only be used once the math package has been imported.

Question 14: 6 points

Develop and test the following code in Python.

Carefully examine and then complete the following Python program.

- The program prompts the user to enter a number between 1 and 5
- It also generates a random number between 1 and 5.
- The program prints the number the user enters and prints the random number.
- The program then compares the two numbers.
  - If the numbers are the same, it prints the message "You picked the same number as the computer!".
  - If the number the user entered is higher than the random number, the program should print "Your number is higher than the computer's number."
  - Otherwise, it should print: "Your number is smaller than the computer's number".

Complete the code for the `getMessage()` function so that it returns the appropriate message, depending on the values stored in the two parameters. See Sample Output for the correct messages to be returned.

```
import random

def getMessage (userNum, randNum):
    #Rewrite this function
    return ""

def main() :
    userNum = int (input("Enter a number between 1 and 5: ") )
    while userNum > 5 or userNum < 1:
        userNum = int (input("Invalid number. Enter a number between 1 and 5: ") )
    randNum = random.randint (1, 5)
    print("Computer number:", randNum)
    print("User number:", userNum)
    print( getMessage (userNum, randNum) )

##Call Main
main()
```

Rewrite the `getMessage` function so that it prints the below responses.

- You picked the same number as the computer!
- Your number is smaller than the computer's number.
- Your number is higher than the computer's number.

Write your new function definition below.

```
In [1]: def getMessage(userNum, randNum):
   ...:     if userNum == randNum:
   ...:         message = "You picked the same number as the computer!"
   ...:     elif userNum < randNum:
   ...:         message = "Your number is smaller than the computer's number."
   ...:     else:
   ...:         message = "Your number is higher than the computer's number.'"
   ...:
   ...:     return message
```

Question 15: 10 points

Complete the following program.
The program should:

- Display five addition facts, one at a time, and allow the user to answer them.
- Provide the correct answer if user enters incorrect answer.
- Print a congratulatory answer, if the answer is correct.
- Keep track of the number of problems the user answers correctly.
- Prints a special message, if the user gets all five problems correct.

```
1   # Include the required import statement
2   # Fill-in 1
3
4   def printRocket ():
5       # Include a FOR loop that prints the numbers from 10 to 0
6       # Fill-in 2
7       # Fill-in 3
8       print("Blast-off!!")
9       print(" *")
10      print("***")
11      print("*****")
12      print("*****")
13      print("*****")
14      print("*****")
15
16  def main ():
17       numCorrect = 0
18       for x in range (5):
19            # Assign a random number between 1 and 10
20            # to num1 and num2
21            num1 = #Fill-in 4
22            num2 = #Fill-in 5
23            answer = int (input( str (num1) + "+" + str (num2)+ "=" ) )
24            # Write the test condition to determine if the user's
25            # answer is equal to the sum of the two numbers
26
27            if # Fill-in 6
28               print ("Congratulations! Your answer is correct!\n")
29               # Write the line of code to increment the variable
30               # that keeps track of the number of correct answers
31               # Fill-in 7
32            else :
33               print ("Sorry, your answer is incorrect.", end = " ")
34               # Print the correct answer
35               # Fill-in 8
36               print ()
37
38       # Write the test condition to determine if the user answers
39       # all the questions correctly
40       if # Fill-in 9
41            print("\nGreat Job! You answered the problems correctly!")
42            # Write the code call the function that prints the rocket
43            # Fill-in 10
```

```
44      else:
45          print("You answered:", numCorrect, "questions correctly.")
46          print("Type again for a perfect score!")
47  ### Call Main
48  main()
```

(a) (1 point) What should go on Line 2 import random

(b) (1 point) What should go on Line 6 for in in range(11, -1, -1):

(c) (1 point) What should go on Line 7 print(i)

(d) (1 point) What should go on Line 21 random.randint(1,10)

(e) (1 point) What should go on Line 22 random.randint(1,10)

(f) (1 point) What should go on Line 27 answer == num1+num2

(g) (1 point) What should go on Line 31 numCorrect += 1

(h) (1 point) What should go on Line 35 print(f'{num1} + {num2} = {num1+num2}')

(i) (1 point) What should go on Line 40 numCorrect == 5

(j) (1 point) What should go on Line 43 printRocket()