FACULTY OF COMPUTING & INFORMATION TECHNOLOGY
KING ABDULAZIZ UNIVERSITY

كلية الحاسبات وتقنية المعلومات
جامعة الملك عبدالعزيز

CPIT 110

Instructor Manual

For 50 Minutes Lectures

Week 14

1/12/2019 – 5/12/2019

Chapter 6

Functions

This Week Events	– Lab #9 (Chapter 6)
Next Week Events	– **Final Exam - Part 2 Writing Code** – **Final Exam - Part 1 MCQ (Week #16)**

CPIT 110

Instructor Manual – Lecture #1 in Week 14

Chapter	6. Functions
Number of Lectures	3 (50 minutes / Lecture)
Lecture	4 of 6
Slides	100 - 114
Date	Sunday 1/12/2019

Week	Lecture
14	4 of 6
	Slides 100 - 114

Topics to Be Covered

- **6.5. Positional and Keyword Arguments**
- **6.6. Passing Arguments by Reference Values**

Learning Objectives

Learning Outcomes	Topics
– To invoke a function using positional arguments or keyword arguments.	6.5. Positional and Keyword Arguments
– To pass arguments by passing their reference value.	6.6. Passing Arguments by Reference Values

Exercises

- **6.5. Positional and Keyword Arguments**
  1. Compare positional arguments and keyword arguments.
  2. Suppose a function header is as follows:
     ```
     def f(p1, p2, p3, p4):
     ```
     Which of the following calls are correct?
     - `f(1, p2 = 3, p3 = 4, p4 = 4)`
     - `f(1, p2 = 3, 4, p4 = 4)`
     - `f(p1 = 1, p2 = 3, 4, p4 = 4)`
     - `f(p1 = 1, p2 = 3, p3 = 4, p4 = 4)`
     - `f(p4 = 1, p2 = 3, p3 = 4, p1 = 4)`

❖ 6.6. Passing Arguments by Reference Values

1. What is pass-by-value?
2. Can the argument have the same name as its parameter?
3. Show the result of the following programs:

```
def main():
    max = 0
    getMax(1, 2, max)
    print(max)

def getMax(value1, value2, max):
    if value1 > value2:
        max = value1
    else:
        max = value2

main()
```

```
def main():
    i = 1
    while i <= 6:
        print(function1(i, 2))
        i += 1

def function1(i, num):
    line = ""
    for j in range(1, i):
        line += str(num) + " "
        num *= 2
    return line

main()
```

```
def main():
    # Initialize times
    times = 3
    print("Before the call, variable",
    "times is", times)
    # Invoke nPrintln and display times
    nPrint("Welcome to CS!", times)
    print("After the call, variable",
    "times is", times)

# Print the message n times
def nPrint(message, n):
    while n > 0:
        print("n = ", n)
        print(message)
        n -= 1

main()
```

```
def main():
    i = 0
    while i <= 4:
        function1(i)
        i += 1
        print("i is", i)

def function1(i):
    line = " "
    while i >= 1:
        if i % 3 != 0:
            line += str(i) + " "
            i -= 1
        print(line)

main()
```

4. For (a) in the preceding question, show the contents of the stack just before the function max is invoked, just as max is entered, just before max is returned, and right after max is returned.

CPIT 110

Instructor Manual – Lecture #2 in Week 14

Chapter	6. Functions
Number of Lectures	3 (50 minutes / Lecture)
Lecture	5 of 6
Slides	115 - 136
Date	Tuesday 3/12/2019

Week	Lecture
14	5 of 6
	Slides 115 - 136

Topics to Be Covered

- **6.7. Modularizing Code**

Topics Not to Be Covered

- 6.8. Case Study: Converting Decimals to Hexadecimal

Learning Objectives

Learning Outcomes	Topics
– To develop reusable code that is modular and is easy to read, debug, and maintain.	6.7. Modularizing Code

Exercises

No exercises.

CPIT 110

Instructor Manual – Lecture #3 in Week 14

Chapter	6. Functions
Number of Lectures	3 (50 minutes / Lecture)
Lecture	6 of 6
Slides	137 - 165
Date	Thursday 5/12/2019

Week	Lecture
14	6 of 6
	Slides 137 - 165

Topics to Be Covered

- **6.9. The Scope of Variables**
- **6.10. Default Arguments**
- **6.11. Returning Multiple Values**

Topics Not to Be Covered

- 6.12. Case Study: Generating Random ASCII Characters
- 6.13. Function Abstraction and Stepwise Refinement
- 6.14. Case Study: Reusable Graphics Functions

Learning Objectives

Learning Outcomes	Topics
– To determine the scope of variables.	6.9. The Scope of Variables
– To define functions with default arguments.	6.10. Default Arguments
– To define a function that returns multiple values.	6.11. Returning Multiple Values

Exercises

❖ 6.9. The Scope of Variables

1. What is the printout of the following code?

```
def function(x):
    print(x)
    x = 4.5
    y = 3.4
    print(y)

x = 2
y = 4
function(x)
print(x)
print(y)
```

```
def f(x, y = 1, z = 2):
    return x + y + z

print(f(1, 1, 1))
print(f(y = 1, x = 2, z = 3))
print(f(1, z = 3))
```

2. What is wrong in the following code?

```
def function():
    x = 4.5
    y = 3.4
    print(x)
    print(y)

function()
print(x)
print(y)
```

3. Can the following code run? If so, what is the printout?

```
x = 10
if x < 0:
    y = -1
else:
    y = 1

print("y is", y)
```

❖ 6.10. Default Arguments

1. Show the printout of the following code:

```
def f(w = 1, h = 2):
    print(w, h)

f()
f(w = 5)
f(h = 24)
f(4, 5)
```

2. Identify and correct the errors in the following program:

```
def main():
    nPrintln(5)

def nPrintln(message = "Welcome to Python!", n):
    for i in range(n):
        print(message)

main() # Call the main function
```

3. What happens if you define two functions in a module that have the same name?

❖ 6.11. Returning Multiple Values

1. Can a function return multiple values? Show the printout of the following code:

```
def f(x, y):
    return x + y, x - y, x * y, x / y

t1, t2, t3, t4 = f(9, 5)
print(t1, t2, t3, t4)
```