CS 1358 Introduction to Programming in Python

Fall Semester 2019

Prof. Pai H. Chou

Self-Check 2

Due Date: Sunday, September 22, 2019, 11:59pm

Answer the following questions to check your understanding of your material. Expect the same kind of questions to show up on your tests.

## 1. Definitions and Short Answers

1. What is a **comment** in a program and what is its purpose?

-> start with the # character till end of line

-> explain to humans

1. What is an **operator**? Give some examples of **arithmetic operators** in Python.

-> 運算子

-> + (plus), - (minus), \* (times), / (divide), // (integer divide), % (remainder), \*\* (exponentiation)

1. What is a **comparison operator**? What are possible results of a comparison?

-> 比較運算子 e.g. < (less than), <= (less than or equal to), > (greater than), >= (greater than or equal to), == (equal to), != (not equal to)

-> True or False

1. What is a **logical operator**? What are possible results of a logical operation?

-> 邏輯運算子 e.g. and, or, not

-> True or False

1. What is 20 in **hexadecimal** representation? in **octal** representation?

-> 20h = 32d

-> 20o = 16d

-> 20d = 14h

-> 20d = 24o

1. Why does Python support two **division operators**? What is their difference?

-> / (divide) 呈現商

-> // (integer divide) 只呈現整數部分的商

1. What is the difference between **'12'** and **12** in Python?

-> **'12'** means string

-> **12** means number

1. What is the difference between **x = 3** and **x == 3** in Python?

-> **x = 3** means assign 3 to x

-> **x == 3** means x equals to 3

1. Assuming the variable y has been assigned the integer value of 4, which of the following are legal in Python and what do they do? which are illegal in Python?
   * y = 4 -> assign 4 to y
   * 4 = y -> illegal
   * y == 4 -> legal, y equals to 4, return True
   * 4 == y -> legal, 4 equals to y, return True
   * 'y' = y -> illegal, can't assign to literal
   * 'y' == '4' -> legal, two strings are not the same, return FALSE
   * '4' = y -> illegal, can't assign to literal
2. Assume variable x has integer value 3, and variable y has integer value of 4. What is the result of the following operator expressions, if they are legal in Python? Which of the following are not legal?
   * x \* y -> legal, return 12
   * 'x' \* y -> legal, return 'xxxx'
   * x \* 'y' -> legal, return 'yyy'
   * 'x' \* 'y' -> illegal, can't multiply sequence by non-int of type 'str'
   * x + y -> legal, return 7
   * 'x' + 'y' -> legal, return 'xy'
   * 'x' + y -> illegal, can only concatenate str (not "int") to str
   * x + 'y' -> illegal, unsupported operand type(s) for +: 'int' and 'str'
3. What is the data type of ['Sun','Mon','Tue','Wed','Thu','Fri','Sat']?

-> List

1. if L = ['Sun','Mon','Tue','Wed','Thu','Fri','Sat'], then what are the **values** of the following **expressions** if they are legal Python? Which are illegal?
   * L[3] -> legal, 'Wed'
   * L[1:5] -> legal, ['Mon','Tue','Wed','Thu']
   * L[5:1] -> legal, []
   * L(2:3) -> illegal
   * L[1,2,3] -> illegal
   * L{3} -> illegal
   * L[1-5] -> legal, ['Wed']
   * L['3'] -> illegal
2. Assume T = ('Sun','Mon','Tue','Wed','Thu','Fri','Sat'), which of the following are allowed in Python, and what are their output or effect? Which are not allowed, for what reasons?
   * print(T[3]) -> legal, Wed
   * print(T(3)) -> illegal
   * print(T{3}) -> illegal
   * T[3] = 'WED' -> illegal
   * T[3] == 'WED' -> legal, False
   * print(T[3:5]) -> legal, ('Wed','Thu')
   * print(T[3, 5]) -> illegal
   * print(T['3']) -> illegal
3. Assume S = {'Sun','Mon','Tue','Wed','Thu','Fri','Sat'}, which of the following are allowed in Python, and what are its output or effect? Which are not allowed, for what reasons?
   * print(S[3]) -> illegal
   * print(S(3)) -> illegal
   * print(S{3}) -> illegal
   * S[3] = 'WED' -> illegal
   * S[3] == 'WED' -> illegal
   * print(S[3:5]) -> illegal
   * print(S[3, 5]) -> illegal
   * print(S['3']) -> illegal
4. Assume D = {'Sun':0, 'Mon':1, 'Tue':2, 'Wed':3, 'Thu':4, 'Fri':5, 'Sat':6}, which of the following are legal in Python, and what are their values?
   * D[3] -> illegal
   * D['Thu'] -> legal, 4
   * D[0:3] -> illegal
   * D[2, 6] -> illegal
   * D{'Sun'} -> illegal
   * D(0) -> illegal
   * D{3} -> illegal
   * D('Sun') -> illegal
5. What is the value of { 2, 3, 4 } | { 3, 4, 5 } ?

-> {2,3,4,5}

1. What is the value of { 2, 3, 4 } & { 3, 4, 5 }?

-> {3,4}

1. Suppose you have the following sequence of Python statements:  
   x = 3  
   y = 2  
   **if** x > y:  
    print("x is bigger than y")  
   **elif** x == y:  
    print("x and y are the same")  
   **else**:  
    print("x is smaller than y")  
   What is printed?

-> x is bigger than y

1. What is wrong with the following code, which is supposed to compute the total of a list of numbers?  
   L = [3, 2, 6, 5]  
   **for** i **in** L:  
    total = total + i  
   print(total)  
   -> total is not defined

How can it be fixed?

L = [3, 2, 6, 5]  
total = 0

**for** i **in** L:  
 total = total + i  
print(total)

1. What is the difference between  
   x = 0  
   **while** x < 100:  
    x = x + 1  
   -> x becomes 100

and  
x = 0  
**if** x < 100:  
 x = x + 1  
-> x is 1

1. What is an example of a **function** in Python?

-> input()

How do you **call** a function?

-> input()

What is a **parameter**?

-> print(“x= “, x)

“x=” is a parameter

1. What is an example of calling a function that **returns a value**?

-> def twice(x):

‘This returns x plus itself’

return x+x

1. Python supports two kinds of **loops**. What are they?

-> for loop

→ repeat the loop over a sequence of items

-> while loop

→ repeat as long as condition evaluates to true

1. What is a **suite**?

-> indented statement block inside a control construct

1. What does **import** math do?

-> import the module name ‘math’

How do you call the cos function (cosine) defined in the math module in Python?

-> **import** math

1. To read a file, it is common to see fh = open('filename'). What kind of data is fh called? -> file handle

Give an example of using fh for accessing (e.g., reading or writing) a file.

-> fh = open(‘filename’,’r’)

-> fh = open(‘filename’,’w’)

1. if s = 'hello', Python supports two styles of “calls” (or “invocation”):
   * len(s) is an example of a **function call**
   * s.upper() is another form of call. What kind of call is it?

-> method call

1. How are **class** and **instance** related to each other?

-> Instance: object made based on class definition

1. Why is it incorrect to split the statement  
   f = a + b \* 2 + c / 2 - 4 \* d  
   onto two separate lines as the following  
   f = a + b \* 2 + c / 2  
    - 4 \* d  
   ? How can it be fixed so Python will accept it?

f = a + b \* 2 + c / 2 \

- 4 \* d

1. If you want to **swap** the values of two variables x and y, why can't you just do  
   x = y  
   y = x  
   ?

-> x does get y , but old value of x is lost!

y gets the new value of x, which is y!  
Give two different ways you can swap their values correctly in Python.

-> (x,y) = (y,x)

-> tmp = x

x = y

y = tmp

1. What is a **keyword** in Python? Give some example keywords in Python.

-> words with reserved meaning in Python language

cannot be reused for other purposes

-> ‘False’, ‘None’, ‘True’, ‘and’, ‘as’, ‘assert’

1. Which of the following are legal and illegal **identifiers** in Python?
   * myname -> legal
   * my\_name -> legal
   * \_myname -> legal
   * MyName -> legal
   * myname\_ -> legal
   * my-name -> illegal
   * my11name -> legal
   * myname11 -> legal
   * 11myName -> illegal
   * my\_11Name -> legal
   * \_11myName -> legal
   * @myname -> illegal
   * my@name -> illegal
   * myname@ -> illegal
   * in -> illegal
   * out -> legal
   * \_in -> legal
   * \_out -> legal
   * IN -> legal
   * OUT -> legal
   * and -> illegal
   * or -> illegal
   * but -> legal
   * function -> legal
   * integer -> legal
   * number -> legal
   * class -> illegal
   * instance -> legal
   * global -> illegal
   * local -> legal
   * you+me -> illegal
   * I\_love\_$$ -> illegal
2. What is an example of a **snake-case** identifier?

-> thie\_is\_an\_example\_of\_a\_snake\_case\_identifier

a **camel-case** identifier?

-> thisIsAnExampleOfACamelCaseIdentifer