Appendix A. Answers to lesson exercises

This appendix contains the answers to the exercises found in the lessons. The answers to the Quick checks are very straightforward, but the answers to some of the Summary exercises may be achieved in several different ways. I have provided a possible solution for each, but your answers may vary slightly from the ones that I have provided.

LESSON 2

Answers to quick checks

Quick check 2.1

1:

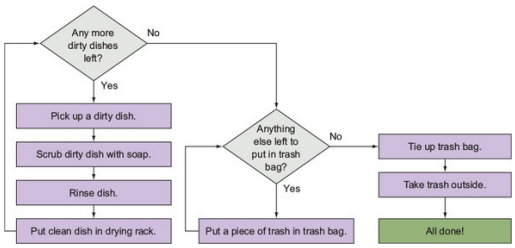
Problem—Make mac-and-cheese.

Vague statement—Dump box of mac-and-cheese in boiling water for 12 minutes.

Specific statement—Pour 6 cups of water in a pot, turn up stovetop temp and wait for water to boil, pour in noodles, let them cook for 12 minutes, drain noodles, add packet of cheese, and stir.

Quick check 2.2

1:



Quick check 2.3

1:

1. Keep value of interest on the left: $c^2 = a^2 + b^2$
2. Take the square root: $c = \sqrt{(a^2 + b^2)}$

Quick check 2.4

1:

```
# initialize times to fill pool (in fraction hours
# convert times to minutes
# convert times to rates
# add rates
# solve for minutes when using both hoses
```

LESSON 3

Answers to quick checks

Quick check 3.1

1

Peek

2

Print

Peek

4

Print

Quick check 3.2

1

Will see `-12`

2

Will see `19`

3

Won't see any output on the console

LESSON 4

Answers to quick checks

Quick check 4.1

1

Not allowed

2

Not allowed

3

Allowed

4

Allowed

Quick check 4.2

1:

Phone attributes—Rectangular, glossy, black, lights up, 4 in x 2 in, has buttons. *Operations*—Click buttons, makes noise, throw it, make a call, type an email.

Dog attributes—Furry, four paws, one mouth, two eyes, one nose, two ears. *Operations*—Bark, scratch, run, jump, whine, lick.

Mirror attributes—Reflective, fragile, sharp. *Operations*—Breaks, shows reflection.

Credit card attributes—3 in x 2 in, thin, flexible, has numbers and letters. *Operations*—Swipe, use to open doors, use to buy stuff.

Quick check 4.3

1

Yes

2

No

3

No

4

No

5

Yes

6

Yes

1

Yes

2

No

3

No (descriptive but not meaningful, unless you're writing a program about unicorns)

4

No (too long)

Quick check 4.5

1

```
apples = 5
```

2

```
oranges = 10
```

3

```
fruits = apples + oranges
```

4

```
apples = 20
```

5

```
fruits = apples + oranges
```

Answers to summary questions

Q4.1

```
x = b - a = 2 - 2 = 0
```

Q4.2

You still get an error. This is because the Python interpreter doesn't understand what to do with the last line. The interpreter is expecting a name to the left of the equal sign, but `a + x` isn't a name.

LESSON 5

Answers to quick checks

Quick check 5.1

1

```
six = 2 + 2 + 2
```

2

```
neg = six * (-6)
```

3

```
neg /= 10
```

Quick check 5.2

1

```
half = 0.25 * 2
```

2

```
other_half = 1.0 - half
```

Quick check 5.3

1

2

```
rain = False
```

3

```
day = cold and rain
```

Quick check 5.4

1

```
one = "one" or one = 'one'
```

2

```
another_one = "1.0" or another_one = '1.0'
```

3

```
last_one = "one 1" or last_one = 'one 1'
```

Quick check 5.5

1

float

2

int

3

bool

4

string

5

string

6

int

7

int

8

string

9

`NoneType`

Quick check 5.6

1

Statement and expression

2

Statement and expression

3

Statement

4

Statement

Quick check 5.7

```
str(True)
```

```
'True'
```

2

```
float(3)
```

```
3.0
```

3

```
str(3.8)
```

```
'3.8'
```

4

```
int(0.5)
```

```
0
```

5

```
int("4")
```

```
4
```

Quick check 5.8

1

```
float
```

```
1.25
```

2

```
float
```

```
9.0
```

3

```
int
```

```
8
```

4

```
int
```

```
201
```

5

```
float
```

```
16.0
```

6

```
float
```

```
1.0
```

7

```
float
```

```
1.5
```

8

```
int
```

```
2
```

9

```
int
0
```

LESSON 6

Answers to quick checks

Quick check 6.1

1

7 hours and 36 minutes

2

0 hours and 0 minutes

3

166 hours and 39 minutes

Quick check 6.2

1

13

2

0

3

12

Quick check 6.3

1

`stars = 50`

2

`stripes = 13`

3

`ratio = stars/stripes` `ratio` is a float

4

`ratio_truncated = int(ratio)` `ratio_truncated` is an int

Quick check 6.4

1:

```
minutes_to_convert = 789
hours_decimal = minutes_to_convert/60
hours_part = int(hours_decimal)
minutes_decimal = hours_decimal-hours_part
minutes_part = round(minutes_decimal*60)
print("Hours")
print(hours_part)
print("Minutes")
print(minutes_part)
```

Output:

```
Hours
13
Minutes
9
```

Answers to summary questions

Q6.1

```
fah = 75
```

Q6.2

```
miles = 5
km = miles/0.62137
meters = 1000*km
print("miles")
print(miles)
print("km")
print(km)
print("meters")
print(meters)
```

LESSON 7

Answers to quick checks

Quick check 7.1

1

Yes

2

Yes

3

No

4

No

5

Yes

Quick check 7.2

1

Forward: 5 Backward: -8

2

Forward: 0 Backward: -13

3

Forward: 12 Backward: -1

Quick check 7.3

1

`'e'`

2

`' '`

(the space character)

3

```
'L'
'x'
```

Quick check 7.4

1

`'t'`

2

`'nhy tWp np'`

3

(empty string, because the start index is further in the string than the stop index, but the step is 1)

Quick check 7.5

1

```
'Python 4 ever&ever'
```

2

```
'PYTHON 4 EVER&ever'
```

3

```
'PYTHON 4 EVER&EVER'
```

4

```
'python 4 ever&ever'
```

Answers to summary questions

Q7.1

```
s = "Guten Morgen"
s[2:5].upper()
```

Q7.2

```
s = "RaceTrack"
s[1:4].captalize()
```

LESSON 8

Answers to quick checks

Quick check 8.1

1

```
14
```

2

```
9
```

3

```
-1
```

4

```
15
```

5

```
6
```

6

```
8
```

Quick check 8.2

1

```
True
```

2

```
True
```

3

```
False
```

Quick check 8.3

1

2

```
2
```

3

```
1
```

4

```
0
```

Quick check 8.4

1

```
'raining in the spring time.'
```

2

```
'Rain in the spr time.'
```

3

```
'Raining in the spring time.'
```

4

(No output) but `b` is now `'Raining in the spring tiempo.'`

Quick check 8.5

1

```
'lalaLand'
```

2

```
'USA vs Canada'
```

3

```
'NYcNYcNYcNYcNYc'
```

4

```
'red-circlered-circlered-circle'
```

Answers to summary questions

Q8.1

There are many other ways of achieving this!

```
s = "Eat Work Play Sleep repeat"
s = s.replace(" ", "ing ")
s = s[7:22]
s = s.lower()
print(s)
```

LESSON 9

Answers to summary questions

Q9.1

1. You're trying to access an index in the string that's beyond the size of the string.
2. You're trying to call the command with an object when the command doesn't need anything in the parentheses.
3. You're trying to call the command with only one object when the command needs two in the parentheses.
4. You're trying to call the command with an object of the wrong type. You must give it a string object, not an integer object.
5. You're trying to call the command with a variable name and not a string object. This would work if you initialized `h` to be a string before you use it.
6. You're trying to multiply two strings when you're only allowed to add two strings or multiply a string by an integer.

LESSON 10

Answers to quick checks

1

```
Yes
```

2

```
Yes
```

3

```
No
```

4

```
Yes
```

Quick check 10.2

1

```
4
```

2

```
2
```

3

```
1
```

4

```
0
```

Quick check 10.3

1

```
(1, 2, 3)
```

2

```
'3'
```

3

```
((1,2), '3')
```

4

```
True
```

Quick check 10.4

1

```
('no', 'no', 'no')
```

2

```
('no', 'no', 'no', 'no', 'no', 'no')
```

3

```
(0, 0, 0, 1)
```

4

```
(1, 1, 1, 1)
```

Quick check 10.5

1

```
(s, w) = (w, s)
```

2

```
(no, yes) = (yes, no)
```

There are many ways you can do this. Here's one way:

```
word = "echo"
t = ()
count = 3

echo = (word,)
echo *= count
cho = (word[1:],)
cho *= count
ho = (word[2:],)
ho *= count
o = (word[3:],)
o *= count

t = echo + cho + ho + o
print(t)
```

LESSON 11

Answers to quick checks

Quick check 11.1

1

```
12
```

2

(Nothing printed)

3

```
Nice is the new cool
```

Quick check 11.2

1

```
sweet = "cookies"
```

2

```
savory = "pickles"
```

3

```
num = 100
```

4

```
print(num, savory, "and", num, sweet)
```

5

```
print("I choose the " + sweet.upper() + "!")
```

Quick check 11.3

1

```
input("Tell me a secret: ")
```

2

```
input("What's your favorite color? ")
```

3

```
input("Enter one of: # or $ or % or & or *: ")
```

Quick check 11.4

1

```
song = input("Tell me your favorite song: ")
print(song)
print(song)
print(song)
```

```
celeb = input("Tell me the first & last name of a
space = celeb.find(" ")
print(celeb[0:space])
print(celeb[space+1:len(celeb)])
```

Quick check 11.5

1:

```
user_input = input("Enter a number to find the squ
num = float(user_input)
print(num*num)
```

Answers to summary questions

Q11.1

```
b = int(input("Enter a number: "))
e = int(input("Enter a number: "))
b_e = b**e
print("b to the power of e is", b_e)
```

Q11.2

```
name = input("What's your name? ")
age = int(input("How old are you? "))
older = age+25
print("Hi " + name + "! In 25 years you will be "
```

LESSON 13

Answers to quick checks

Quick check 13.1

1

No

2

Yes

3

No

4

No

5

No

Quick check 13.2

1

You live in a treehouse.

2

(Can't be converted.)

3

(Can't be converted.)

4

The word youniverse is in the dictionary.

5

The number 7 is even.

6

Variables a and b are equal

1

```
num is less than 10
Finished
```

2

```
Finished
```

3

```
Finished
```

Quick check 13.4

1

```
word = input("Tell me a word: ")
print(word)
if " " in word:
    print("You did not follow directions!")
```

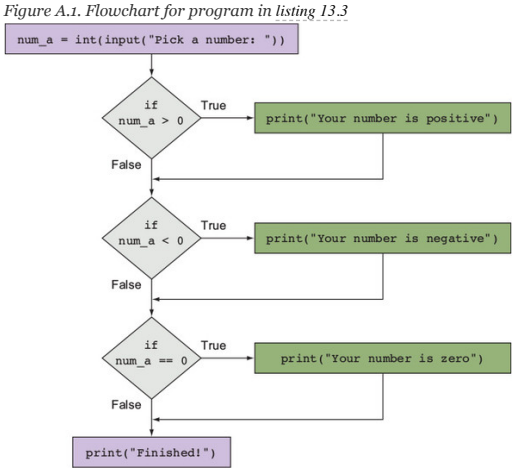
2

```
num1 = int(input("One number: "))
num2 = int(input("Another number: "))
print(num1+num2)
if num1+num2 < 0:
    print("Wow, negative sum!")
```

Quick check 13.5

1:

Figure A.1. Flowchart for program in listing 13.3



Quick check 13.6

1:

```
num_a       num_b       Answer (nested)          An
-------------------------------------------------
-9          5           num_a: is negative       nu
                        Finished                 Fi
-------------------------------------------------
 9          5           Finished                 Fi
-------------------------------------------------
-9          -5          num_a: is negative       nu
                        num_b is negative        nu
                        Finished                 Fi
-------------------------------------------------
 9          -5          Finished                 nu
                                                 Fi
```

Quick check 13.7

1:

One possible solution shown in listing 13.5.

Answers to summary questions

Q13.1

Q13.2

```
var = 0
if type(var) == int:
    print("I'm a numbers person.")
if type(var) == str:
    print("I'm a words person.")
```

Q13.3

```
words = input("Tell me anything: ")
if " " in words:
    print("This string has spaces.")
```

Q13.4

```
print("Guess my number! ")
secret = 7
num = int(input("What's your guess? "))
if num < secret:
    print("Too low.")
if num > secret:
    print("Too high.")
if num == secret:
    print("You got it!")
```

Q13.5

```
num = int(input("Tell me a number: "))
if num >= 0:
    print("Absolute value:", num)
if num < 0:
    print("Absolute value:", -num)
```

LESSON 14

Answers to quick checks

Quick check 14.1

1

Do you need milk and have a car? If yes, drive to the store to buy milk.

2

Is variable a zero and variable b zero and variable c zero? If yes, then all variables are zero.

3

Do you have a jacket or a sweater? Take one of these; it's cold outside.

Quick check 14.2

1

True

2

True

3

False

Quick check 14.3

1:

```
num_a       num_b
  0           0
  0          -5
-20           0
 -1          -1
-20        -988
```

Quick check 14.4

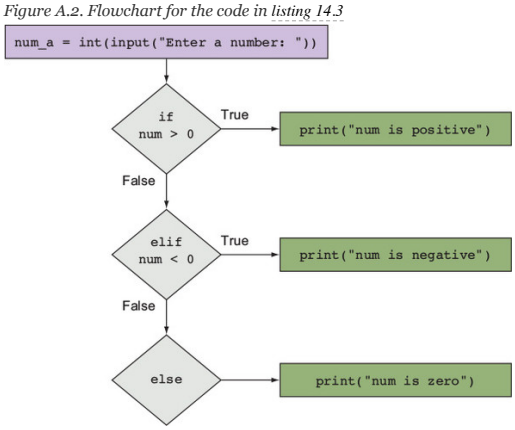
1:

```
num is -3       Output: num is negative
num is 0        Output: num is zero
num is 2        Output: num is positive
num is 1        Output: num is positive
```

Quick check 14.5

1:

Figure A.2. Flowchart for the code in listing 14.3



Quick check 14.6

1:

```
num         With if-elif-else             With if
-------------------------------------------------
20          num is greater than 3         num is g
            Finished.                     Finished
-------------------------------------------------
 9          num is less than 10           num is l
            Finished.                     num is g
                                          Finished
-------------------------------------------------
 5          num is less than 6            num is l
            Finished.                     num is l
                                          num is g
                                          Finished
-------------------------------------------------
 0          num is less than 6            num is l
            Finished.                     num is l
                                          Finished
```

Answers to summary questions

Q14.1

```
num1 = int(input("One number: "))
num2 = int(input("Another number: "))
if num1 < num2:
    print("first number is less than the second nu
elif num2 < num1:
    print("first number is greater than the second
else:
    print("numbers are equal")
```

Q14.2

```
words = input("Enter anything: ")
if "a" in words and "e" in words and "i" in words
    print("You have all the vowels!")
if words[0] == 'a' and words[-1] == 'z':
    print("And it's sort of alphabetical!")
```

LESSON 16

Answers to quick checks

Quick check 16.1

1

```
for i in range(8):
    print("crazy")
```

2

```
for i in range(100):
    print("centipede")
```

Quick check 16.2

1

0, 1

2

0, 1, 2, 3, 4

3

0, 1, 2, 3, 4, 5, 6, ..., 99

Answers to summary questions

Q16.1

```
num = int(input("Tell me a number: "))
for i in range(num):
    print("Hello")
```

It's not possible to write without a `for` loop because you don't know the number the user will give you.

LESSON 17

Answers to quick checks

Quick check 17.1

1

0, 1, 2, 3, 4, 5, 6, 7, 8

2

3, 4, 5, 6, 7

3

-2, 0, 2

4

5, 2, -1, -4

5

(Nothing)

Quick check 17.2

1:

```
vowels = "aeiou"
words = input("Tell me something: ")
for letter in words:
    if letter in vowels:
        print("vowel")
```

Answers to summary questions

Q17.1

```
counter = 0
for num in range(2, 100, 2):
    if num%6 == 0:
        counter += 1
print(counter, "numbers are even and divisible by
```

Q17.2

```
count = int(input("How many books on Python do you
for n in range(count,0,-1):
    if n == 1:
        print(n, "book on Python on the shelf", n,
        print("Take one down, pass it around, no m
```

```
            print(n, "books on Python on the shelf", r
            print("Take one down, pass it around,", n-
```

Q17.3

```
names = input("Tell me some names, separated by sp
name= ""
for ch in names:
    if ch == " ":
        print("Hi", name)
        name = ""
    else:
        name += ch
# deal with the last name given (does not have a s
lastspace = names.rfind(" ")
print("Hi", names[lastspace+1:])
```

LESSON 18

Answers to quick checks

Quick check 18.1

1:

```
password = "robot fort flower graph"
space_count = 0
for ch in password:
    if ch == " ":
        space_count += 1
print(space_count)
```

As a side note, the preceding code can also be written using a command on strings, `count`, with `password.count(" ")`.

Quick check 18.2

1:

```
secret = "snake"
word = input("What's my secret word? ")
guesses = 1
while word != secret:
    word = input("What's my secret word? ")
    if guesses == 20 and word != secret:
        print("You did not get it.")
        break
    guesses += 1
```

Answers to summary questions

Q18.1

```
# corrected code
num = 8
guess = int(input("Guess my number: "))
while guess != num:
    guess = int(input("Guess again: "))
print("Right!")
```

Q18.2

```
play = input("Play? y or yes: ")
while play == 'y' or play == "yes":
    num = 8
    guess = int(input("Guess a number! "))
    while guess != num:
        guess = int(input("Guess again: "))
    print("Right!")
    play = input("Play? y or yes: ")
print("See you later!")
```

LESSON 20

Answers to quick checks

Quick check 20.1

1

Independent

2

3

Independent

Quick check 20.2

1

In: Pen, paper, name, address, envelope, stamp, wedding date, fiancée
Out: Wedding invitation ready to be mailed

2

In: Phone number, phone

Out: No output

3

In: Coin

Out: Heads or tails

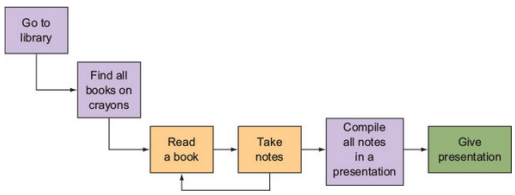
4

In: Money

Out: A dress

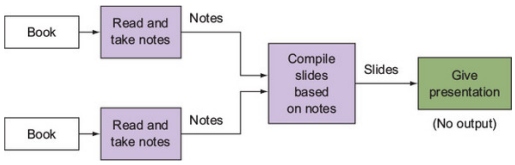
Quick check 20.3

1:



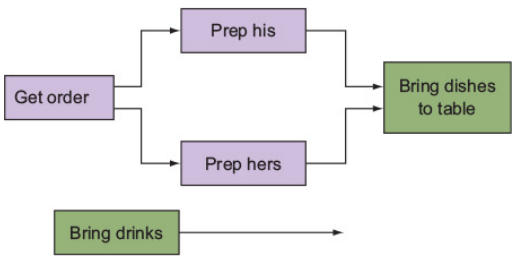
Quick check 20.4

1:



Answers to summary questions

Q20.1



LESSON 21

Answers to quick checks

Quick check 21.1

1

```
def set_color(name, color):
```

2

```
def get_inverse(num):
```

3

```
def print_my_name():
```

Quick check 21.2

1

3

2

0

3

4

Quick check 21.3

1

Yes (when 2 and 3 are variables types that can be added together)

2

Yes

3

No (indentation error)

Quick check 21.4

These are only a few possibilities; many others exist:

1

`get_age` or `get_tree_age`

2

`translate` or `dog_says`

3

`cloud_to_animal` or `take_picture`

4

`age` or `get_age` or `years_later`

Quick check 21.5

1

Length of variable sign (return type is an integer)

2

True (return type is a Boolean)

3

`"and toes"` (return type is a string)

Quick check 21.6

1

```
return (money_won, guessed)
```

2

- `(100, True)`
- `(1.0, False)`
- Doesn't print anything
- Doesn't print anything

```
False
8.0
```

Quick check 21.7

1

2

```
Hector is eating
```

3

```
Hector is eating 8 bananas
```

4

```
Hector is bananas is eating 8 bananas
```

5

```
None
```

Answers to summary questions

Q21.1

1.
```
def calculate_total(price, percent):
    tip = price*percent/100
    total = price + tip
    return total
```
2. `calculate_total(20, 15)`
3.
```
my_price = 78.55
my_tip = 20
total = calculate_total(my_price, my_tip)
print("Total is:", total)
```

LESSON 22

Answers to quick checks

Quick check 22.1

1

-11

-11.0

2

-3

-3.0

3

24

1.5

4

32

2.0

Quick check 22.2

1

42

2

6

3

12

4

21

Quick check 22.3

```
-------------------------------------------------
def sandwich(kind_of_sandwich):
   print("--------")
   print(kind_of_sandwich ())
   print("--------")
def blt():
    my_blt = " bacon\nlettuce\n tomato"
    return my_blt
def breakfast():
    my_ec = " eggegg\n cheese"
    return my_ec

print(sandwich(blt))
```

```
    return my_ec

print(sandwich(blt))
```

```
SCOPE OF blt()
Returns:  bacon
         lettuce
          tomato

-------------------------------------------------
def sandwich(kind_of_sandwich):
   print("--------")
   print(kind_of_sandwich ())
   print("--------")
def blt():
    my_blt = " bacon\nlettuce\n tomato"
    return my_blt
def breakfast():
    my_ec = " eggegg\n cheese"
    return my_ec

print(sandwich(blt))
```

Quick check 22.4

1:

```
def grumpy():
    print("I am a grumpy cat:")
    def no_n_times(n):
        print("No", n,"times...")
        def no_m_more_times(m):
            print("...and no", m,"more times")
            for i in range(n+m):
                print("no")
        return no_m_more_times
    return no_n_times

grumpy()(4)(2)
```

```
-------------------------------------------------
def grumpy():
    print("I am a grumpy cat:")
    def no_n_times(n):
        print("No", n,"times...")
        def no_m_more_times(m):
            print("...and no", m,"more times")
            for i in range(n+m):
                print("no")
        return no_m_more_times
    return no_n_times

grumpy()(4)(2)
```

```
-------------------------------------------------
def grumpy():
    print("I am a grumpy cat:")
    def no_n_times(n):
        print("No", n,"times...")
        def no_m_more_times(m):
            print("...and no", m,"more times")
            for i in range(n+m):
                print("no")
        return no_m_more_times
    return no_n_times

grumpy()(4)(2)
```

```
def grumpy():
    print("I am a grumpy cat:")
    def no_n_times(n):
        print("No", n,"times...")
        def no_m_more_times(m):
            print("...and no", m,"more times")
            for i in range(n+m):
                print("no")
        return no_m_more_times
    return no_n_times

grumpy()(4)(2)
```

this lir

```
def grumpy():
    print("I am a grumpy cat:")
    def no_n_times(n):
        print("No", n,"times...")
        def no_m_more_times(m):
            print("...and no", m,"more times")
            for i in range(n+m):
                print("no")
        return no_m_more_times
    return no_n_times

grumpy()(4)(2)
```

GLOBAL
grumpy:

SCOPE OF
n:
no_m_mor

```
def grumpy():
    print("I am a grumpy cat:")
    def no_n_times(n):
        print("No", n,"times...")
        def no_m_more_times(m):
            print("...and no", m,"more times")
            for i in range(n+m):
                print("no")
        return no_m_more_times
    return no_n_times

grumpy()(4)(2)
```

GLOBAL
grumpy:

SCOPE OF
n: 4
no_m_mor
Returns:

```
def grumpy():
    print("I am a grumpy cat:")
    def no_n_times(n):
        print("No", n,"times...")
        def no_m_more_times(m):
            print("...and no", m,"more times")
            for i in range(n+m):
                print("no")
        return no_m_more_times
    return no_n_times

grumpy()(4)(2)
```

this line

GLOBAL
grumpy:

```
def grumpy():
    print("I am a grumpy cat:")
    def no_n_times(n):
        print("No", n,"times...")
        def no_m_more_times(m):
            print("...and no", m,"more times")
            for i in range(n+m):
```

```
grumpy()(4)(2)
                                                GLOBAL S
                                                grumpy:

                                                SCOPE OF
                                                m:     2

-------------------------------------------------
def grumpy():
    print("I am a grumpy cat:")
    def no_n_times(n):
        print("No", n,"times...")
        def no_m_more_times(m):
            print("...and no", m,"more times")
            for i in range(n+m):
                print("no")                      <
        return no_m_more_times
    return no_n_times

grumpy()(4)(2)
GLOBAL SCOPE
                                                grumpy:

                                                SCOPE OF
                                                m:     2
                                                Returns:

-------------------------------------------------
def grumpy():
    print("I am a grumpy cat:")
    def no_n_times(n):
        print("No", n,"times...")
        def no_m_more_times(m):
            print("...and no", m,"more times")
            for i in range(n+m):
                print("no")
        return no_m_more_times
    return no_n_times

grumpy()(4)(2)
and done with this line
GLOBAL SCOPE
                                                grumpy:
```

Answers to summary questions

Q22.1

```
def area(shape, n):
    # write a line to return the area
    # of a generic shape with a parameter of n
    return shape(n)
```

1. `area(circle, 10)`
2. `area(square, 5)`
3. `area(circle, 4/2)`

Q22.2

```
def person(age):
    print("I am a person")
    def student(major):
        print("I like learning")
        def vacation(place):
            print("But I need to take breaks")
            print(age,"|",major,"|",place)
        return vacation
    return student
```

1. `person(29)("CS")("Japan")`
2. `person(23)("Law")("Florida")`

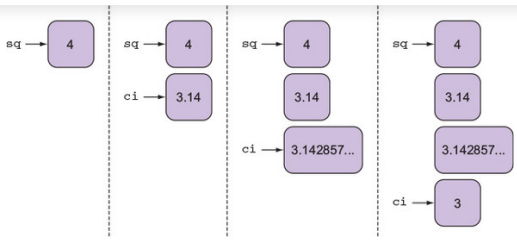
LESSON 24

Answers to quick checks

Quick check 24.1

1:

Figure A.3. Visualization of the sequence of statements



Quick check 24.2

1

Either immutable object (tuple, because the names of cities won't change) or mutable (list, because you may add/remove cities as needed)

2

Immutable object, an int (a mutable object would be overkill because age is only one item, so the overhead of changing it isn't worth making it mutable)

3

Mutable object, a dictionary to store an item and its cost

4

Immutable object, a string

Answers to summary questions

Q24.1

`one` is an immutable object.

`age` is a mutable object.

LESSON 25

Answers to quick checks

Quick check 25.1

1

Tuple

2

Tuple

3

Tuple

4

List

5

List

Quick check 25.2

1

tape

2

mouse

3

Error, index out of bounds

4

stapler

1:

```
1
4
0
8
2
error
```

Quick check 25.4

1

```
[1, '1']
```

2

`[0, ['zero']]` (Notice the second element is another list.)

3

```
[]
```

4

```
[1,2,3,4,5]
[0,1,2,3,4,5]
```

Quick check 25.5

1:

```
[3,1,4,1,5,9]
[3,1,4,1,5]
[3,1,4,1]
```

Quick check 25.6

1

```
[1, 2, 3, 4, 7, 11, 13, 17]
```

2

```
[1, 2, 3, 4, 6, 11, 13, 17]
```

3

```
[1, 2, 3, 4, 6, 11, 13, 1]
```

4

```
[3, 2, 3, 4, 6, 11, 13, 1]
```

Answers to summary questions

Q25.1

1.
```
menu = []
menu.append("pizza")
menu.append("beer")
menu.append("fries")
menu.append("wings")
menu.append("salad")
```

2.
```
menu[0] = menu[-1]
menu[-1] = ""
menu.pop(1)
menu[-1] = "pizza"
```

3.
```
menu.pop()
menu.pop()
menu.pop()
menu.append("quinoa")
menu.append("steak")
```

```
def unique(L):
    L_unique = []
    for n in L:
        if n not in L_unique:
            L_unique.append(n)
    return L_unique
```

Q25.3

```
def unique(L):
    L_unique = []
    for n in L:
        if n not in L_unique:
            L_unique.append(n)
    return L_unique

def common(L1, L2):
    unique_L1 = unique(L1)
    unique_L2 = unique(L2)
    length_L1 = len(unique_L1)
    length_L2 = len(unique_L2)
    if length_L1 != length_L2:
        return False
    else:
        for i in range(length_L1):
            if L1[i] not in L2:
                return False
       return True
```

LESSON 26

Answers to quick checks

Quick check 26.1

1:

```
['g', 'n', 'i', 'm', 'm', 'a', 'r', 'g', 'o', 'r',
['a', 'g', 'g', 'i', 'm', 'm', 'n', 'o', 'p', 'r',
['r', 'r', 'p', 'o', 'n', 'm', 'm', 'i', 'g', 'g',
['a', 'g', 'g', 'i', 'm', 'm', 'n', 'o', 'p', 'r',
['a', 'g', 'g', 'i', 'm', 'm', 'n', 'o', 'p', 'r',
```

Quick check 26.2

1

```
board = [[empty, empty, empty], [x, x, x], [o, o,
o]]
```

2

```
board = [[x, o, x, o], [o, o, x, x], [o, empty, x,
x]]
```

Quick check 26.3

1

```
" abcdefghijklmnopqrstuvwxyz".split(" ")
```

2

```
"spaces and more spaces".split(" ")
```

3

```
"the secret of life is 42".split("s")
```

Quick check 26.4

1

Stack

2

Stack

3

Queue

Neither (because the first luggage out may never get picked up)

Answers to summary questions

Q26.1

```
cities = "san francisco,boston,chicago,indianapoli
city_list = cities.split(",")
city_list.sort()
print(city_list)
```

Q26.2

```
def is_permutation(L1, L2):
    L1.sort()
    L2.sort()
    return L1 == L2
```

LESSON 27

Answers to quick checks

Quick check 27.1

1

```
employee_database = {}
Key: string for the name
Value: tuple of (phone number as a string, home ad
```

2

```
snow_accumulation = {}
Key: string for the city
Value: tuple (int year 1990, float for snow in 19
2000)
```

3

```
valuables = {"tv": 2000, "sofa": 1500}
Key: string for the item name
Value: int for the value
```

Quick check 27.2

1

Three entries. Maps integers to integers.

2

Three entries. Maps strings to integers.

3

Three entries. Maps integers to lists.

Quick check 27.3

1:

```
{}
{'LA': 3884}
{'NYC': 8406, 'LA': 3884}
{'NYC': 8406, 'LA': 3884, 'SF': 837}
{'NYC': 8406, 'LA': 4031, 'SF': 837}
```

Quick check 27.4

1:

```
3.14
1.41
(there will be an error)
```

Quick check 27.5

1:

```
25
51
35
```

Answers to summary questions

Q27.1

```
songs = {"Wannabe": 3, "Roar": 4, "Let It Be": 5,

for s in songs.keys():
    if songs[s] == 5:
        print(s)
```

Q27.2

```
def replace(d, v, e):
    for k in d:
        if d[k] == v:
            d[k] = e
```

Q27.3

```
def invert(d):
    d_inv = {}
    for k in d:
        v = d[k]
        if v not in d_inv:
            d_inv[v] = [k]
        else:
            d_inv[v].append(k)
    return d_inv
```

LESSON 28

Answers to quick checks

Quick check 28.1

1

Same ID

2

Same ID

3

Same ID (Technically, this should be a different ID because immutable objects don't have aliases. But Python is optimizing behind the scenes by referencing the object that already exists with the same value instead of creating another one. These optimizations aren't guaranteed to happen all the time.)

Quick check 28.2

1

Same ID

2

Same ID

3

Different ID (You're creating another object that happens to have the same elements, not an alias.)

Quick check 28.3

1

Yes

2

Yes

3

4

Yes

5

No

Quick check 28.4

1

```
order = sorted(chaos)
```

2

```
colors.sort()
```

3

```
cards = deck
```

4 `deck.sort()`

Answers to summary questions

Q28.1

```
def invert_dict(d):
    new_d = {}
    for k in d.keys():
        new_d[d[k]] = k
    return new_d
```

Q28.2

```
def invert_dict_inplace(d):
    new_d = d.copy()
    d = {}
    for k in new_d.keys():
        d[d_new[k]] = k
```

LESSON 30

Answers to quick checks

Quick check 30.1

1

Yes, with an integer

2

Yes, with a tuple (or a list)

3

No (would need to decide which properties and behaviors to define a person—for example, a name, an age, height, weight, can they walk, talk?)

4

No (would need to decide which properties and behaviors to define a chair—for example, number of legs, height, depth, what can you do with a chair?)

Quick check 30.2

1

A width and a height

2

A width, a height, a depth, number of ports, number of pixels, and so forth

3

Number of legs, seat back or not, cushioned or not

4

Name, age, height, weight, hair color, eye color, and so forth

Quick check 30.3

1

Find the area or the perimeter

2

Turn it on/off, get its diagonal, connect a cable to a port

3

Have a person sit on a chair, cut off a leg, add a cushion

4

Change name, increment the age, change hair color

Quick check 30.4

1

String

2

List

3

Dictionary

4

String

LESSON 31

Answers to quick checks

Quick check 31.1

1

```
class Person(object):
```

2

```
class Car(object):
```

3

```
class Computer(object):
```

Quick check 31.2

1:

```
class Person(object):
    def __init__(self):
        self.name = ""
        self.age = 0


class Car(object):
    def __init__(self):
        self.length = 0
        self.width = 0
        self.height = 0


class Computer(object):
    def __init__(self):
        self.on = False
        self.touchscreen = False
```

Quick check 31.3

1:

```
class Door(object):
    def __init__(self):
        self.width = 1
        self.height = 1
        self.open = False
    def get_status(self):
        return self.open
    def get_area(self):
        return self.width*self.height
```

Quick check 31.4

1:

```
square_door = Door()
square_door.change_state()
square_door.scale(3)
```

Quick check 31.5

1:

```
a = Rectangle(1,1)
b = Rectangle(1,1)
Rectangle.set_length(a, 4)
Rectangle.set_width(b, 4)
```

Answers to summary questions

Q31.1

```
def get_area(self):
    """ returns area of a circle """
    return 3.14*self.radius**2

# testing method
a = Circle()
print(a.get_area()) # shoould be 0
a.change_radius(3)
print(a.get_area()) # should be 28.26
```

Q31.2

```
def get_area(self):
    """ returns area of a rectangle """
    return self.length*self.width

def get_perimeter(self):
    """ returns perimeter of a rectangle """
    return self.length*2 + self.width*2
```

LESSON 32

Answers to quick checks

Quick check 32.1

1:

```
def add_list(self, L):
    for e in L:
        self.stack.append(e)
```

Quick check 32.2

1:

```
circles  = Stack()
for i in range(3):
    one_circle = Circle()
    one_circle.change_radius(3)
    circles.add_one(one_circle)
rectangles  = Stack()
one_rectangle = Rectangle(1, 1)
rectangles.add_many(one_rectangle, 5)
```

Answers to summary questions

Q32.1

```
class Queue(object):
    def __init__(self):
```

```
            return self.queue.copy()
        def add_one(self, item):
            self.queue.append(item)
        def add_many(self, item, n):
            for i in range(n):
                self.queue.append(item)
        def remove_one(self):
            self.queue.pop(0)
        def remove_many(self, n):
            for i in range(n):
                self.queue.pop(0)
        def size(self):
            return len(self.queue)
        def prettyprint(self):
            for thing in self.queue[::-1]:
                print('|_',thing, '_|')

    # testing the class by making objects and doing op
    a = Queue()
    a.add_one(3)
    a.add_one(1)
    a.prettyprint()
    a.add_many(6,2)
    a.prettyprint()
    a.remove_one()
    a.prettyprint()
    b = Queue()
    b.prettyprint()
```

LESSON 33

Answers to quick checks

Quick check 33.1

1:

```
def __sub__(self, other_fraction):
    new_top = self.top*other_fraction.bottom -  \
              self.bottom*other_fraction.top
    new_bottom = self.bottom*other_fraction.bottom
    return Fraction(new_top, new_bottom)
```

Quick check 33.2

1:

```
def __str__(self):
        toreturn = str(self.top) + "\n--\n" + str(
        return toreturn
```

Quick check 33.3

1

```
quarter.__mul__(half)
Fraction.__mul__(quarter, half)
```

2

```
quarter.__str__()
Fraction.__str__(quarter)
```

3

```
(half.__mul__(half)).__str__()
Fraction.__str__(Fraction.__mul__(half, half))
```

Answers to summary questions

Q33.1

```
class Circle(object):
    def __init__(self):
        self.radius = 0
    def change_radius(self, radius):
        self.radius = radius
    def get_radius(self):
        return self.radius
    def __str__(self):
        return "circle: "+str(self.radius)
```

```
    def __init__( self):
        self.stack = []
    def get_stack_elements(self):
        return self.stack.copy()
    def add_one(self , item):
        self.stack.append(item)
    def add_many(self , item, n):
        for i in range(n):
            self.stack.append(item)
    def remove_one(self):
        self.stack.pop()
    def remove_many(self , n):
        for i in range(n):
            self.stack.pop()
    def size(self):
        return len(self.stack)
    def prettyprint(self):
        for thing in self.stack[::-1]:
            print('|_',thing, '_|')
    def __str__(self):
        ret = ""
        for thing in self.stack[::-1]:
            ret += ('|_ '+str(thing)+ ' _|\n')
        return ret
```

LESSON 35

Answers to quick checks

Quick check 35.1

1:

```
import fruits
import activities
```

Quick check 35.2

1:

```
import math

distance = float(input("How far away is your frier
speed = float(input("How fast can you throw? (m/s)

tolerance = 2

# 0 degrees means throw horizontal and 90 degrees
for i in range(0,91):
    angle_r = math.radians(i)
    reach = 2*speed**2*math.sin(angle_r)*math.cos(
    if reach > distance - tolerance and reach < di
        print("angle: ", i, "Nice throw!")
    elif reach < distance - tolerance:
        print("angle: ", i, "You didn't throw far
    else:
        print("angle: ", i, "You threw too far.")
```

Quick check 35.3

1:

```
import random

heads = 0
tails = 0
for i in range(100):
    r = random.random()
    if r < 0.5:
        heads += 1
    else:
        tails += 1
print("Heads:", heads)
print("Tails:", tails)
```

Quick check 35.4

1:

```
import time
import random

count = 0
start = time.clock()
for i in range(10000000):
    count += 1
```

```
end = time.clock()
print(end-start)     # prints about 4.5 seconds
```

Answers to summary questions

Q35.1

```
import time
import random

def roll_dice():
    r = str(random.randint(1,6))
    # put bars around the number so it looks like
    dice = " _ \n|" + r + "|"
    print(dice)
    return r

start = time.clock()

p = "roll"
while p == "roll":
    print("You rolled a dice...")
    userroll = roll_dice()
    print("Computer rolling...")
    comproll = roll_dice()
    time.sleep(2)
    if userroll >= comproll:
        print("You win!")
    else:
        print("You lose.")
    p = input("Type roll to roll again, any other

end = time.clock()
print("You played for", end-start, "seconds.")
```

LESSON 36

Answers to quick checks

Quick check 36.1

1:

```
class TestMyCode(unittest.TestCase):
    def test_addition_5_5(self):
        self.assertEqual(5+5, 10)
    def test_remainder_6_2(self):
        self.assertEqual(6%2, 0)
```

Quick check 36.2

1:

```
def is_prime(n):
    prime = True
    for i in range(2,n):
        if n%i == 0:
            prime = False
    return prime

def absolute_value(n):
    if n < 0:
        return -n
    elif n >= 0:
        return n
```

Quick check 36.3

1

```
assertFalse(x, msg=None)
```

2

```
assertIn(a, b, msg=None)
```

3

```
assertDictEqual(a, b, msg=None)
```

Quick check 36.4

1

```
Breakpoint at line isprime = funcs.is_prime(5)
```

2

Click blue arrow with two vertical lines, click button with two arrows

3

Step into function and notice that loop starts at 1, not 2

Answers to summary questions

Q36.1

```
import unittest

def remove_buggy(L, e):
    """
    L, list
    e, any object
    Removes all e from L.
    """
    for i in L:
        if e == i:
            L.remove(i)

def remove_fixed(L, e):
    """
    L, list
    e, any object
    Removes all e from L.
    """
    for i in L.copy():
        if e == i:
            L.remove(i)

class Tests(unittest.TestCase):
    def test_123_1(self):
        L = [1,2,3]
        e = 1
        remove_buggy(L,e)
        self.assertEqual(L, [2,3])
    def test_1123_1(self):
        L = [1,1,2,3]
        e = 1
        remove_buggy(L,e)
        self.assertEqual(L, [2,3])

unittest.main()
```

LESSON 37

Answers to quick checks

Quick check 37.1

1

A button: click it

2

A scrollbar: hold mouse button and drag

3

A menu: hover over an item and click it

4

A canvas: draw lines, circles, rectangles, erase

Quick check 37.2

1

```
import tkinter
window = tkinter.Tk()
window.geometry("500x200")
window.title("go go go")
window.configure(background="green")
window.mainloop()
```

2

```
window.geometry("100x900")
window.title("Tall One")
window.configure(background="red")
window.mainloop()
```

3

```
import tkinter
window1 = tkinter.Tk()
window1.geometry("100x100")
window1.configure(background="white")
window2 = tkinter.Tk()
window2.geometry("100x100")
window2.configure(background="black")
window1.mainloop()
window2.mainloop()
```

Quick check 37.3

1:

```
btn = tkinter.Button(window, text="Click here", bg
radio_btn1 = tkinter.Radiobutton()
radio_btn2 = tkinter.Radiobutton()
check_btn = tkinter.Checkbutton()
```

Quick check 37.4

1:

```
import tkinter
import random

def changecolor():
    r = random.choice(["red", "green", "blue"])
    window.configure(background=r)

window = tkinter.Tk()
window.geometry("800x600")
window.title("My first GUI")

btn = tkinter.Button(window, text="Random color!",
btn.pack()

window.mainloop()
```

Answer to summary questions

Q37.1

```
import tkinter

window = tkinter.Tk()
window.geometry("200x800")
window.title("PhoneBook")

phonebook = {}

def add():
    name = txt_name.get()
    phone = txt_phone.get()
    email = txt_email.get()
    phonebook[name] = [phone, email]
    lbl.configure(text = "Contact added!")

def show():
    s = ""
    for name, details in phonebook.items():
        s += name+"\n"+details[0]+"\n"+details[1]+
    lbl.configure(text=s)

txt_name = tkinter.Entry()
txt_phone = tkinter.Entry()
txt_email = tkinter.Entry()

btn_add = tkinter.Button(text="Add contact", comma
btn_show = tkinter.Button(text="Show all", commanc

lbl = tkinter.Label()

txt_name.pack()
txt_phone.pack()
txt_email.pack()
btn_add.pack()
```

PREV
Lesson 38. Capstone project: game of tag

NEXT
Appendix B. Python cheat sheet