Chapter 1

1. The operators are +, -, \*, and /. The values are 'hello', -88.8, and 5.

2. The string is 'spam'; the variable is spam. Strings always start and end

with quotes.

3. The three data types introduced in this chapter are integers, floatingpoint numbers, and strings.

4. An expression is a combination of values and operators. All expressions

evaluate (that is, reduce) to a single value.

5. An expression evaluates to a single value. A statement does not.

6. The bacon variable is set to 20. The bacon + 1 expression does not reassign

the value in bacon (that would need an assignment statement: bacon =

bacon + 1).

7. Both expressions evaluate to the string 'spamspamspam'.

8. Variable names cannot begin with a number.

9. The int(), float(), and str() functions will evaluate the integer, floating-point number, and string versions of the value passed to them.

10. The expression causes an error because 99 is an integer, and only

strings can be concatenated to other strings with the + operator. The

the correct way is' I have eaten ' + str(99) + ' burritos.'.

Chapter 2

1. True and False, using capital T and F, with the rest of the word in

lowercase

2. and, or, and not

3. True and True is True.

True and False is False.

False and True is False.

False and False is False.

True or True is True.

True or False is True.

False or True is True.

False or False is False.

not True is False.

not False is True.

4. False

False

True

False

False

True

Answers to the Practice Questions 449

5. ==, !=, <, >, <=, and >=.

6. == is the equal to operator that compares two values and evaluates to

a Boolean, while = is the assignment operator that stores a value in a

variable.

7. A condition is an expression used in a flow control statement that evaluates to a Boolean value.

8. The three blocks are everything inside the if statement and the lines

print('bacon') and print('ham').

print('eggs')

if spam > 5:

print('bacon')

else:

print('ham')

print('spam')

9. The code:

if spam == 1:

print('Hello')

elif spam == 2:

print('Howdy')

else:

print('Greetings!')

10. Press ctrl-C to stop a program stuck in an infinite loop.

11. The break statement will move the execution outside and just after a loop.

The continue statement will move the execution to the start of the loop.

12. They all do the same thing. The range(10) call ranges from 0 up to (but

not including) 10, range(0, 10) explicitly tells the loop to start at 0, and

range(0, 10, 1) explicitly tells the loop to increase the variable by 1 on

each iteration.

13. The code:

for i in range(1, 11):

print(i)

and:

i = 1

while i <= 10:

print(i)

i = i + 1

14. This function can be called with spam.bacon().

Chapter 3

1. Functions reduce the need for duplicate code. This makes programs

shorter, easier to read, and easier to update.

2. The code in a function executes when the function is called, not when

the function is defined.

3. The def statement defines (that is, creates) a function.

4. A function consists of the def statement and the code in its def clause.

A function call is what moves the program execution into the function,

and the function call evaluates to the function’s return value.

5. There is one global scope, and a local scope is created whenever a function is called.

6. When a function returns, the local scope is destroyed, and all the variables in it are forgotten.

7. A return value is the value that a function call evaluates to. Like any

value, a return value can be used as part of an expression.

8. If there is no return statement for a function, its return value is None.

9. A global statement will force a variable in a function to refer to the

global variable.

10. The data type of None is NoneType.

11. That import statement imports a module named areallyourpetsnamederic.

(This isn’t a real Python module, by the way.)

12. This function can be called with spam.bacon().

13. Place the line of code that might cause an error in a try clause.

14. The code that could potentially cause an error goes in the try clause.

The code that executes if an error happens goes in the except clause.

Chapter 4

1. The empty list value, which is a list value that contains no items. This is

similar to how '' is the empty string value.

2. spam[2] = 'hello' (Notice that the third value in a list is at index 2

because the first index is 0.)

3. 'd' (Note that '3' \* 2 is the string '33', which is passed to int() before

being divided by 11. This eventually evaluates to 3. Expressions can be

used wherever values are used.)

4. 'd' (Negative indexes count from the end.)

5. ['a', 'b']

6. 1

7. [3.14, 'cat', 11, 'cat', True, 99]

8. [3.14, 11, 'cat', True]

Answers to the Practice Questions 451

9. The operator for list concatenation is +, while the operator for replication is \*. (This is the same as for strings.)

10. While append() will add values only to the end of a list, insert() can add

them anywhere in the list.

11. The del statement and the remove() list method are two ways to remove

values from a list.

12. Both lists and strings can be passed to len(), have indexes and slices, be

used in for loops, be concatenated or replicated, and be used with the

in and not in operators.

13. Lists are mutable; they can have values added, removed, or changed.

Tuples are immutable; they cannot be changed at all. Also, tuples are

written using parentheses, ( and ), while lists use the square brackets,

[ and ].

14. (42,) (The trailing comma is mandatory.)

15. The tuple() and list() functions, respectively

16. They contain references to list values.

17. The copy.copy() function will do a shallow copy of a list, while the

copy.deepcopy() function will do a deep copy of a list. That is, only copy

.deepcopy() will duplicate any lists inside the list.

Chapter 5

1. Two curly brackets: {}

2. {'foo': 42}

3. The items stored in a dictionary are unordered, while the items in a list

are ordered.

4. You get a KeyError error.

5. There is no difference. The in operator checks whether a value exists as

a key in the dictionary.

6. 'cat' in spam checks whether there is a 'cat' key in the dictionary, while

'cat' in spam.values() checks whether there is a value 'cat' for one of

the keys in spam.

7. spam.setdefault('color', 'black')

8. pprint.pprint()

Chapter 6

1. Escape characters represent characters in string values that would

otherwise be difficult or impossible to type into code.

2. \n is a newline; \t is a tab.

3. The \\ escape character will represent a backslash character.

452 Appendix C

4. The single quote in Howl's is fine because you’ve used double quotes to

mark the beginning and end of the string.

5. Multiline strings allow you to use newlines in strings without the \n

escape character.

6. The expressions evaluate to the following:

• 'e'

• 'Hello'

• 'Hello'

• 'lo world!

7. The expressions evaluate to the following:

• 'HELLO'

• True

• 'hello'

8. The expressions evaluate to the following:

• ['Remember,', 'remember,', 'the', 'fifth', 'of', 'November.']

• 'There-can-be-only-one.'

9. The rjust(), ljust(), and center() string methods, respectively

10. The lstrip() and rstrip() methods remove whitespace from the left and

right ends of a string, respectively.

Chapter 7

1. The re.compile() function returns Regex objects.

2. Raw strings are used so that backslashes do not have to be escaped.

3. The search() method returns Match objects.

4. The group() method returns strings of the matched text.

5. Group 0 is the entire match, group 1 covers the first set of parentheses,

and group 2 covers the second set of parentheses.

6. Periods and parentheses can be escaped with a backslash: \., \(, and \).

7. If the regex has no groups, a list of strings is returned. If the regex has

groups, a list of tuples of strings is returned.

8. The | character signifies matching “either, or” between two groups.

9. The ? character can either mean “ match zero or one of the preceding

group” or be used to signify nongreedy matching.

10. The + matches one or more. The \* matches zero or more.

11. The {3} matches exactly three instances of the preceding group. The

{3,5} matches between three and five instances.

12. The \d, \w, and \s shorthand character classes match a single digit,

word, or space character, respectively.

13. The \D, \W, and \S shorthand character classes match a single character

that is not a digit, word, or space character, respectively.

Answers to the Practice Questions 453

14. Passing re.I or re.IGNORECASE as the second argument to re.compile() will

make the matching case insensitive.

15. The . character normally matches any character except the newline

character. If re.DOTALL is passed as the second argument to re.compile(),

then the dot will also match newline characters.

16. The .\* performs a greedy match, and the .\*? performs a nongreedy

match.

17. Either [0-9a-z] or [a-z0-9]

18. 'X drummers, X pipers, five rings, X hens'

19. The re.VERBOSE argument allows you to add whitespace and comments to

the string passed to re.compile().

20. re.compile(r'^\d{1,3}(,{3})\*$') will create this regex, but other regex

strings can produce a similar regular expression.

21. re.compile(r'[A-Z][a-z]\*\sNakamoto')

22. re.compile(r'(Alice|Bob|Carol)\s(eats|pets|throws)\

s(apples|cats|baseballs)\.', re.IGNORECASE)

Chapter 8

1. Relative paths are relative to the current working directory.

2. Absolute paths start with the root folder, such as / or C:\.

3. The os.getcwd() function returns the current working directory. The

os.chdir() function changes the current working directory.

4. The . folder is the current folder, and .. is the parent folder.

5. C:\bacon\eggs is the dir name, while spam.txt is the base name.

6. The string 'r' for read mode, 'w' for write mode, and 'a' for append mode

7. An existing file opened in write mode is erased and completely

overwritten.

8. The read() method returns the file’s entire contents as a single string

value. The readlines() method returns a list of strings, where each

string is a line from the file’s contents.

9. A shelf value resembles a dictionary value; it has keys and values, along

with keys() and values() methods that work similarly to the dictionary

methods of the same names.

Chapter 9

1. The shutil.copy() function will copy a single file, while shutil.copytree()

will copy an entire folder, along with all its contents.

2. The shutil.move() function is used for renaming files, as well as

moving them.

454 Appendix C

3. The send2trash functions will move a file or folder to the recycle bin,

while shutil functions will permanently delete files and folders.

4. The zipfile.ZipFile() function is equivalent to the open() function; the

first argument is the filename, and the second argument is the mode to

open the ZIP file in (read, write, or append).

Chapter 10

1. assert(spam >= 10, 'The spam variable is less than 10.')

2. assert(eggs.lower() != bacon.lower(), 'The eggs and bacon variables are

the same!') or assert(eggs.upper() != bacon.upper(), 'The eggs and bacon

variables are the same!')

3. assert(False, 'This assertion always triggers.')

4. To be able to call logging.debug(), you must have these two lines at the

start of your program:

import logging

logging.basicConfig(level=logging.DEBUG, format=' %(asctime)s -

%(levelname)s - %(message)s')

5. To be able to send logging messages to a file named programLog.txt

with logging.debug(), you must have these two lines at the start of your

program:

import logging

>>> logging.basicConfig(filename='programLog.txt', level=logging.DEBUG,

format=' %(asctime)s - %(levelname)s - %(message)s')

6. DEBUG, INFO, WARNING, ERROR, and CRITICAL

7. logging.disable(logging.CRITICAL)

8. You can disable logging messages without removing the logging function calls. You can selectively disable lower-level logging messages. You

can create logging messages. Logging messages provides a timestamp.

9. The Step button will move the debugger into a function call. The Over

button will quickly execute the function call without stepping into it.

The Out button will quickly execute the rest of the code until it steps

out of the function it currently is in.

10. After you click Go, the debugger will stop when it has reached the end

of the program or a line with a breakpoint.

11. A breakpoint is a setting on a line of code that causes the debugger to

pause when the program execution reaches the line.

12. To set a breakpoint in IDLE, right-click the line and select Set

Breakpoint from the context menu.