Control Statements and Program Development

* Introduction

No questions.

* Algorithms

3.2 Q1: Which of the following statements is false?

a. You can solve any computing problem by executing a series of actions in a specific order.

b. An algorithm is a procedure for solving a problem in terms of the actions to execute, and the order in which these actions execute.

c. Program control specifies the order in which statements (actions) execute in a program.

d. All of the above statements are true.

Answer: d. All of the above statements are true.

* Pseudocode

3.3 Q1:          is an informal English-like language for “thinking out” algorithms.

a. Code

b. Python

c. Quasicode

d. Pseudocode

Answer: d. Pseudocode

* Control Statements

3.4 Q1: Various Python statements enable you to specify that the next statement to execute may be other than the next one in sequence. This is called          and is achieved with Python control statements.

a. transfer of control

b. alternate execution

c. spread execution

d. None of the above

Answer: a. transfer of control

3.4 Q2: Which of the following statements is false?

a. The notion of structured programming became almost synonymous with “goto elimination.”

b. Although Python has a goto statement, you should avoid it to minimize errors.

c. Bohm and Jacopini demonstrated that all programs could be written using three forms of control—namely, sequential execution, the selection statement and the iteration statement.

d. Python statements execute one after the other “in sequence,” unless directed otherwise.

Answer: b. Although Python has a goto statement, you should avoid it to minimize errors. Actually, Python does not have a **goto** statement.

3.4 Q3: The most important flowchart symbol is the          , which indicates that a decision is to be made, such as in an if statement.

a. rectangle

b. flowline

c. diamond

d. small circle

Answer: c. diamond

3.4 Q4: Which of the following statements is false?

a. Python provides three types of selection statements that execute code based on a condition—an expression that evaluates to either True or False:

b. The if...else statement performs an action if a condition is True or performs a different action if the condition is False.

c. Anywhere a single action can be placed, a group of actions can be placed.

d. The if…elif…else statement is called a double-selection statement because it selects one of two different actions (or groups of actions).

Answer: d. The if…elif…else statement is called a double-selection statement because it selects one of two different actions (or groups of actions). Actually, the if…elif…else statement is called a *multiple*-selection statement because it selects one of *many* different actions (or groups of actions).

3.4 Q5: Python provides two iteration statements—         and         :

a. do while and for

b. while and for each

c. do while and for each

d. while and for

Answer: **d. while and for**

3.4 Q6: Which of the following statements is false?

a. You form each Python program by combining as many control statements of each type as you need for the algorithm the program implements.

b. With Single-entry/single-exit (one way in/one way out) control statements, the exit point of one connects to the entry point of the next. This is similar to the way a child stacks building blocks—hence, the term control-statement stacking.

c. You can construct any Python program from only six different forms of control (sequential execution, and the if, if…else, if…elif…else, while and for statements). You combine these in only two ways (control-statement stacking and control-statement nesting). This is the essence of simplicity.

d. All of the above statements are true.

Answer: d. All of the above statements are true.

* **if** Statement

3.5 Q1: Which of the following statements is false?

a. Indenting a suite is required; otherwise, an IndentationError syntax error occurs.

b. If you have more than one statement in a suite, those statements do not need to have the same indentation.

c. Sometimes error messages may not be clear. The fact that Python calls attention to the line is usually enough for you to figure out what’s wrong.

d. Programs that are not uniformly indented are hard to read.

Answer: b. If you have more than one statement in a suite, those statements do not need to have the same indentation. Actually, an IndentationError occurs if you have more than one statement in a suite and those statements do not have the *same* indentation.

3.5 Q2: Which of the following statements is false?

a. The decision (diamond) symbol contains a condition that can be either True or False.

b. The diamond flowchart symbol has three flowlines emerging from it.

c. One flowline emerging from the diamond flowchart symbol indicates the direction to follow when the condition in the symbol is True. This points to the action (or group of actions) that should execute.

d. Another flowline emerging from the diamond flowchart symbol indicates the direction to follow when the condition is False. This skips the action (or group of actions).

Answer: b. The diamond flowchart symbol has three flowlines emerging from it. Actually, the diamond flowchart symbol has two flowlines emerging from it.

3.5 Q3 Which of the following statements is false?

a. You can base decisions on any expression.

b. A condition which evaluates to a nonzero value is considered to be True.

c. A condition which evaluates to a value of zero is considered to be False:

d. Strings containing characters are True and empty strings ('', "" or """""") are False.

Answer: All of the above statements are true.

3.5 Q4 Which of the following statements is false?

a. Using the equality operator == instead of the assignment symbol = in an assignment statement can lead to subtle problems.

b. If instead of defining a variable grade with the assignment:

grade = 85

we accidentally write:

grade == 85

then grade would be undefined and we’d get a NameError.

c. If grade had been defined before the statement grade == 85, the statement would evaluate to True or False, depending on grade’s value, and not perform the intended assignment—fortunately, this is harmless.

d. All of the above statements are true.

Answer: c. If grade had been defined before the statement grade == 85, the statement would evaluate to True or False, depending on grade’s value, and not perform the intended assignment—fortunately, this is harmless. Actually, this is a logic error.

* **if**…**else** and **if**…**elif**…**else** Statements

3.6 Q1: Which of the following statements is false?

a. The if…else statement performs different suites, based on whether a condition is True or False.

b. If we initialize the variable grade to 55, the following code snippet would display Failed:

if grade >= 60:   
 print('Passed')  
else:   
 print('Failed')

c. The condition above is True, so the if suite displays Passed.

d. All of the above are *true*.

Answer: **c. The condition above is True, so the if suite displays Passed. Actually, the condition above is False, so the if suite displays Failed.**

3.6 Q2: Which of the following statements about navigating through snippets is false?

a. The up and down arrow keys navigate backwards and forwards through the current interactive session’s snippets.

b. Pressing Shift+Enter re-executes the snippet that’s displayed.

c. In IPython, every recalled snippet that you execute gets a new ID.

d. All of the above statements are true.

Answer. b. Pressing *Shift+Enter* re-executes the snippet that’s displayed. Actually, pressing *Enter* re-executes the snippet that’s displayed.

3.6 Q3: Which of the following statements is false?

a. Sometimes the suites in an if…else statement assign different values to a variable, based on a condition, as in:

grade = 87  
  
if grade >= 60:   
 result = 'Passed'  
else:   
 result = 'Failed'

b. You can if…else statements like the one above using a concise conditional expression:

result = ('Passed' if grade >= 60 else 'Failed')

c. The parentheses in a conditional expression are required, or you would get a syntax error.

d. In interactive mode, you also can evaluate the conditional expression directly, as in:

In [1]: 'Passed' if grade >= 60 else 'Failed'  
Out[1]: 'Passed'

Answer: c. The parentheses **a conditional expression** are required, or you would get a syntax error. Actually, the parentheses are not required, but they can make it clear that the statement in Part (b) assigns the conditional expression’s value to result.

3.6 Q4: Which of the following statements is false?

a. The following code shows two statements in the else suite of an if…else statement:

grade = 49  
  
if grade >= 60:   
 print('Passed')  
else:   
 print('Failed')   
 print('You must take this course again')

b. In the code in Part (a), grade is less than 60, so both statements in the else’s suite execute.

c. In the code of Part (a), even if you do not indent the second print, then it’s still in the else’s suite. So, the code runs correctly.

d. All of the above statements are true.

Answer: c. In the code of Part (a), even if you do not indent the second print, then it’s still in the else’s suite. So, the code runs correctly. Actually, in the code of Part (a), if you do not indent the second print, then it’s not in the else’s suite. So, that statement *always* executes, regardless of whether the condition is True or False.

3.6 Q5: Which of the following statements is false?

a. With a nonfatal logic error, an IPython interactive session terminates.

b. For a fatal logic error in a script, an exception occurs (such as a ZeroDivisionError from an attempt to divide by 0), so Python displays a traceback, then terminates the script.

c. A fatal error in interactive mode terminates only the current snippet. Then IPython waits for your next input.

d. All of the above statements are true.

Answer: a. With a nonfatal logic error, an IPython interactive session terminates. Actually, with a nonfatal logic error, the code executes, but produces incorrect results.

* **while** Statement

3.7 Q1: Which of the following statements is false?

a. The while statement allows you to repeat one or more actions while a condition remains True. Such a statement often is called a loop.

b. The following code finds the first power of 3 larger than 50:

product = 3  
  
while product < 50:  
 product = product \* 3

c. Something in a while statement’s suite must ensure that the condition eventually becomes False. Otherwise, a logic error called an infinite loop occurs.

d. In applications executed from a Terminal, Command Prompt or shell, type Ctrl + c or control + c (depending on your keyboard) to terminate an infinite loop.

Answer: b. The following code finds the first power of 3 larger than 50:

**product = 3  
  
while product < 50:  
 product = product \* 3**

Actually, the condition in the **while** statement must be product <= 50 or the code finds the first power of 3 which is 50 or larger.

* **for** Statement

3.8 Q1: Which of the following statements is false?

a. Built-in function print displays its argument(s), then moves the cursor to the next line. You can change this behavior with the argument end, as in

print(character, end=' ')

b. Python calls end a keyword argument, and end is a Python keyword.

c. The end keyword argument is optional. If you do not include it, print uses a newline ('\n') by default.

d. Keyword arguments are sometimes called named arguments.

Answer: b. Python calls end a keyword argument, and end is a Python keyword. Actually, end is *not* a Python keyword.

3.8 Q2: What does the following line of code display?

print(10, 20, 30, sep=', ')

a. 102030

b. 10,20,30

c. 10 20 30

d. 10, 20, 30

Answer: d. 10, 20, 30

* **Iterables, Lists and Iterators**

3.8 Q3 Which of the following statements is false?

a. The sequence to the right of the for statement’s keyword in must be an iterable.

b. An iterable is an object from which the for statement can take one item at a time until no more items remain.

c. One of Python’s most common iterable sequences is the list, which is a comma-separated collection of items enclosed in square brackets ([ and ]).

d. The following code totals five integers in a list:

total = 0  
  
for number in [2, -3, 0, 17, 9]:  
 total + number

Answer: **The following code totals five integers in a list:**

**total = 0  
  
for number in [2, -3, 0, 17, 9]:  
 total + number**

Actually, change the statement in the **for**’s suite from **total + number** to **total = total + number** for the code to total the five elements in the list.

3.8 Q4: Which of the following statements is false?

a. Each sequence has an iterator.

b. The for statement uses the iterator “behind the scenes” to get each consecutive item until there are no more to process.

c. The iterator is like a bookmark—it always knows where it is in the sequence, so it can return the next item when it’s called upon to do so.

d. Lists are unordered and a list’s items are mutable.

Answer: d. Lists are unordered and a list’s items are mutable. Actually, lists are *ordered* (and mutable).

* **Built-In** range **Function**

3.8 Q5: What does the following for statement print?

for counter in range(10):   
 print(counter, end=' ')

a. It doesn’t run because it’s syntactically incorrect.

b. 0 1 2 3 4 5 6 7 8 9 10

c. 1 2 3 4 5 6 7 8 9

d. 0 1 2 3 4 5 6 7 8 9

Answer: d. 0 1 2 3 4 5 6 7 8 9

* Augmented Assignments

3.9 Q1: Which of the following statements is false?

a. Augmented assignments abbreviate assignment expressions in which the same variable name appears on the left and right of the assignment’s =, as total does in:

for number in [1, 2, 3, 4, 5]:  
 total = total + number

b. The following code re-implements the preceding for statement using an addition augmented assignment (+=) statement:

for number in [1, 2, 3, 4, 5]:  
 total += number

c. The statement f = f \*\* 3 may be replaced by the more concise augmented assignment statement f \*= 3.

d. All of the above statements are true.

Answer: c. The statement f = f \*\* 3 may be replaced by the more concise augmented assignment statement f \*= 3. c.   
Actually, the statement f = f \*\* 3 may be replaced by the more concise augmented assignment statement f \*\*= 3.

* Program Development: Sequence-Controlled Iteration
* Requirements Statement

No questions.

* Pseudocode for the Algorithm

3.10 Q1: Which of the following statements is false.

a. Experience has shown that the most challenging part of solving a problem on a computer is developing an algorithm for the solution.

b. Once a correct algorithm has been specified, creating a working Python program from the algorithm is typically straightforward.

c. A requirements statement describes how a program should operate but not what the program is supposed to do.

d. Variables for totaling and counting normally are initialized to zero before they’re used.

Answer: c. A requirements statement describes how a program should operate but not what the program is supposed to do. Actually, a requirements statement describes *what* a program is supposed to do, but not *how* the program should do it.

* Coding the Algorithm in Python

3.10 Q2: Which of the following statements is false?

a. Many scripts can be decomposed into initialization, processing and termination phases.

b. The following statement creates the variable grades and initializes it with a list of 10 integer grades.

grades = [98, 76, 71, 87, 83, 90, 57, 79, 82, 94]

c. Iteration is called definite if the number of iterations is first known after the loop finishes executing.

d. The Style Guide for Python Code recommends placing a blank line above and below each control statement.

Answer: c. Iteration is called *definite* if the number of iterations is first known after the loop finishes executing. Actually, with definite iteration, the number of iterations is known *before* the loop begins executing.

* Introduction to Formatted Strings

3.10 Q3: Which of the following statements is false?

a. The following f-string inserts the value of average into a string:

f'Class average is {average}'

b. You specify where to insert values in an f-string by using placeholders delimited by curly braces ({ and }).

c. The placeholder {average} converts the variable average’s value to a string representation, then replaces {average} with that replacement text.

d. Replacement-text expressions may contain values, variables or other expressions, such as calculations, but may not contain function calls.

Answer: d. **Replacement-text expressions may contain values, variables or other expressions, such as calculations, but may not contain function calls. Actually, replacement-text expressions may contain values, variables or other expressions, such as calculations or function calls.**

* Program Development: Sentinel-Controlled Iteration

3.11 Q1: Which of the following statements is false?

a. A sentinel value indicates “end of data entry.”

b. Sentinel-controlled iteration is often called indefinite iteration because the number of iterations is not known before the loop begins executing.

c. A sentinel value must match one acceptable input value.

d. Grades on a quiz are typically nonnegative integers between 0 and 100, so the value –1 is an acceptable sentinel value for grades data.

Answer: c. A sentinel value must match one acceptable input value. Actually, a sentinel value *must not* be confused with any acceptable input value.

3.11 Q2: Which of the following statements is false?

a. In top-down, stepwise refinement, the top is a single statement that conveys the program’s overall function but rarely conveys enough detail from which to write a program.

b. The top specifies what should be done, but not how to implement it.

c. In the refinement process, we decompose the top into a sequence of smaller tasks—a process sometimes called divide and conquer.

d. In the top-down, stepwise refinement process for developing algorithms, each refinement represents another portion of the algorithm.

Answer: d. In the top-down, stepwise refinement process for developing algorithms, each refinement represents another portion of the algorithm. Actually, in the top-down, stepwise refinement process for developing algorithms, each refinement represents the *complete* algorithm—only the level of detail varies.

3.11 Q3: Which of the following statements is false. In an f-string, you can optionally follow a replacement-text expression with a colon (:) and a format specifier that describes how to format the replacement text.

b. The format specifier .2f formats data as a floating-point number (f) with two digits to the right of the decimal point (.2).

c. Formatting floating-point data with .2f rounds it to the tenths position.

d. All of the above statements are true.

Answer: c. Formatting **floating-point** data with .2f *rounds* it to the tenths position. Actually, Formatting data with .2f *rounds* it to the hundredths position.

* Program Development: Nested Control Statements

3.12 Q1: In the following code segment:

# process 10 students

for student in range(10):

# get one exam result

result = int(input('Enter result (1=pass, 2=fail): '))

if result == 1:

passes = passes + 1

else:

failures = failures + 1

a. The if statement is nested in the for statement.

b. The if statement follows the for statement in sequence.

c. The for statement is nested in the if statement.

d. None of the above.

Answer: a. The if statement is nested in the for statement.

* Built-In Function **range**: A Deeper Look

3.13 Q1: Which of the following statements is false?

a. Function range’s one-argument version produces a sequence of consecutive integers from 0 up to, but not including, the argument’s value.

b. The following snippet produces the sequence 5 6 7 8 9.

for number in range(5, 10):  
 print(number, end=' ')

c. The following snippet produces the sequence 0 2 4 6 8.

for number in range(0, 10, 2):  
 print(number, end=' ')

d. The following snippet produces the sequence 10 8 6 4 2 0.

for number in range(10, 0, -2):  
 print(number, end=' ')

Answer: d. The following snippet produces the sequence 10 8 6 4 2 0.

**for number in range(10, 0, -2):  
 print(number, end=' ')**

Actually, the snippet produces the sequence 10 8 6 4 2.

* Using Type **Decimal** for Monetary Amounts

3.14 Q1: Which of the following statements is false?

a. Many applications require precise representation of numbers with decimal points.

b. Financial institutions like banks that deal with millions or even billions of transactions per day have to tie out their transactions “to the penny.” Floating-point numbers can represent some but not all monetary amounts with to-the-penny precision.

c. For monetary calculations and other applications that require precise representation and manipulation of numbers with decimal points, the Python Standard Library provides type Decimal, which uses a special coding scheme to solve the problem of "to-the-penny precision." Banks also have to deal with other issues such as using a fair rounding algorithm when they’re calculating daily interest on accounts. Type Decimal offers such capabilities.

d. Floating-point values are stored and represented precisely in binary format.

Answer: **d. Floating-point values are stored and represented precisely in binary format.** Actually, some floating-point values are represented only approximately when they’re converted to binary.

3.14 Q2: Which of the following statements is false?

a. The Python Standard Library is divided into modules—groups of related capabilities.

b. The decimal module defines type Decimal and its capabilities.

c. To use Decimal, you must first import the its module, as in

import decimal

and refer to the Decimal type as decimal.Decimal, or you must indicate a specific capability to import using from…import, as in:

from decimal import Decimal

which imports only the type Decimal from the decimal module so that you can use it in your code.

d. All of the above statements are true.

Answer: d. All of the above statements are *true*.

3.14 Q3: Which of the following statements is false?

a. You typically create a Decimal from a string.

b. Decimals support the standard arithmetic operators +, -, \*, /, //, \*\* and %, as well as the corresponding augmented assignments:

c. You may perform arithmetic between Decimals and integers.

d. You may perform arithmetic between Decimals and floating-point numbers.

Answer: d. You may perform arithmetic between Decimals and floating-point numbers. Actually, you may **not** perform arithmetic between Decimals and floating-point numbers.

3.14 Q4: Which of the following statements is false?

a. The following statement uses an f-string with two placeholders to format year and amount.

print(f'{year:>2}{amount:>10.2f}')

b. The placeholder {year:>2} uses the format specifier >2 to indicate that year’s value should be right aligned (>)in a field of width 2—the field width specifies the number of character positions to use when displaying the value.

c. For single-digit year values 1–9, the format specifier >2 displays a value followed by the space character, thus right aligning the years in the first column.

d. The format specifier 10.2f in the placeholder {amount:>10.2f} formats amount as a floating-point number (f) right aligned (>) in a field width of 10 with a decimal point and two digits to the right of the decimal point (.2). Formatting a column of amounts this way aligns their decimal points vertically, as is typical with monetary amounts.

Answer: **c. For single-digit year values 1–9, the format specifier >2 displays a value followed by the space character, thus right aligning the years in the first column. Actually, for single-digit year values 1–9, the format specifier >2 displays a space character followed by the value, thus right aligning the years in the first column.**

* **break** and **continue** Statements

3.15 Q1: Which of the following statements is false?

a. Executing a break statement in a while or for immediately exits that statement.

b. The following snippet produces the integer sequence 0 1 2 3 4 5 6 7 8 9.

for number in range(100):   
 if number == 10:  
 break  
 print(number, end=' ')

c. The while and for statements each have an optional else clause that executes only if the loop terminates normally—that is, not as a result of a break.

d. The following code snippet produces the sequence 0 1 2 3 4 5 5 6 7 8 9.

for number in range(10):   
 if number == 5:  
 continue  
 print(number, end=' ')

Answer: **d. The following code snippet produces the sequence 0 1 2 3 4 5 5 6 7 8 9.**

**for number in range(10):   
 if number == 5:  
 continue  
 print(number, end=' ')**

Actually, the code snippet produces the sequence **0 1 2 3 4 6 7 8 9**.

* Boolean Operators **and**, **or** and **not**

3.16 Q1: Which of the following statements about the IPython session below is true?

In [1]: gender = 'Female'

In [2]: age = 70

In [3]: if gender == 'Female' and age >= 65:

...: print('Senior female')

...:

Senior female

a. The session defines two variables, then tests a condition that’s True if and only if both simple conditions are True—if either (or both) of the simple conditions is False, the entire and expression is False.

b. The right side of the and operator evaluates only if the left side is True.

c. The combined condition can be made clearer by adding redundant (unnecessary) parentheses

(gender == 'Female') and (age >= 65)

d. All of the above statements are true.

Answer: d. All of the above statements are true.

3.16 Q2: Which of the following statements is false?

a. Python stops evaluating an and-expression as soon as it knows whether the entire condition is False. Python stops evaluating an or-expression as soon as it knows whether the entire condition is True.

b. The following condition stops evaluating immediately if gender is not equal to 'Female' because the entire expression must be False. If gender is equal to 'Female', execution continues, because the entire expression will be True if the age is greater than or equal to 65.

gender == 'Female' and age >= 65

c. The following condition stops evaluating immediately if semester\_average is greater than or equal to 90 because the entire expression must be True.

semester\_average >= 90 or final\_exam >= 90

d. In operator expressions that use and, make the condition that’s more likely to be True the leftmost condition—in or operator expressions, make the condition that’s more likely to be False the leftmost condition—each of these tactics can reduce a program’s execution time.

Answer: **d. In operator expressions that use and, make the condition that’s more likely to be True the leftmost condition—in or operator expressions, make the condition that’s more likely to be False the leftmost condition—each of these tactics can reduce a program’s execution time.** Actually, in operator expressions that use and, make the condition that’s more likely to be **False** the leftmost condition—in or operator expressions, make the condition that’s more likely to be **True** the leftmost condition—each of these tactics can reduce a program’s execution time.

3.16 Q3: Which of the following statements is false

a. You place the not operator before a condition to choose a path of execution if the original condition (without the not operator) is True.

b. The if statement

if not grade == -1:   
 print('The next grade is', grade)

can also be written as follows:

if grade != -1:  
 print('The next grade is', grade)

c. The Boolean not operator reverses the meaning of a condition—True becomes False and False becomes True.

d. The not operator is a unary operator—it has only one operand.

Answer: a. You place the not operator before a condition to choose a path of execution if the original condition (without the not operator) is True. Actually, you place the not operator before a condition to choose a path of execution if the original condition (without the not operator) is False.

* Intro to Data Science: Measures of Central Tendency—Mean, Median and Mode

3.17 Q1: Which of the following statements is false?

a. The descriptive statistics mean, median and mode are measures of central tendency—each is a way of producing a single value that is in some sense typical of the others.

b. The following session creates a list called grades, then uses the built-in sum and len functions to calculate the median “by hand”—sum calculates the total of the grades (397) and len returns the number of grades (5):

In [1]: grades = [85, 93, 45, 89, 85]

In [2]: sum(grades) / len(grades)

Out[2]: 79.4

c. Like functions min and max, sum and len are both examples of functional-style programming reductions—they reduce a collection of values to a single value.

d. The Python Standard Library’s statistics module provides functions for calculating the mean, median and mode—these, too, are reductions.

Answer: b. The following session creates a list called grades, then uses the built-in sum and len functions to calculate the median “by hand”—sum calculates the total of the grades (397) and len returns the number of grades (5).

**In [1]: grades = [85, 93, 45, 89, 85]**

**In [2]: sum(grades) / len(grades)**

**Out[2]: 79.4**

Actually, the session uses the built-in sum and len functions to calculate the *mean* “by hand.”

3.17 Q2: Which of the following statements is false?

a. The argument of each of the functions mean, median and mode must be an iterable.

b. To help confirm the median and mode values of a grades list, you can use the built-in sorted function to get a copy of grades with its values arranged in increasing order, as in the following session, which makes it clear that both the median and the mode are 85:

In [1]: grades = [85, 93, 45, 89, 85]

In [2]: sorted(grades)

Out[2]: [45, 85, 85, 89, 93]

c. If a list’s number of values is even, median returns the mode of the two middle values.

d. The mode function causes a StatisticsError for lists like

[85, 93, 45, 89, 85, 93]

in which there are two or more “most frequent” values. Such a set of values is said to be bimodal.

Answer: c. If a list’s number of values is even, median returns the *mode* of the two middle values. Actually, if a list’s number of values is even, median returns the *average (mean)* of the *two* middle values.