CHAPTER EXERCISES

Section 3.1

1. Which of the three forms of control is an implicit form of control?
2. What is meant by a "straight-line" program?
3. What is the difference between a control statement and a control structure?

Section 3.2

4. The Boolean data type contains two literal values, denoted as ________________ and ________________ in Python.
5. Which of the following relational expressions evaluate to `True`?
   (a) `5 < 8` **(b)** `'5' < '8'` **(c)** `'10' < '8'` **(d)** `'Jake' < 'Brian'`
6. Which of the following relational expressions evaluate to `False`?
   (a) `5 <= 5` **(b)** `5 >= 5` **(c)** `5 == 5` **(d)** `5 != 5` **(e)** `5 != 10`
7. Give an appropriate expression for each of the following.
   (a) To determine if the number 24 does *not* appear in a given list of numbers assigned to variable `nums`.
   (b) To determine if the name `'Ellen'` appears in a list of names assigned to variable `names`.
   (c) To determine if a single last name stored in variable `last_name` is either `'Morris'` or `'Morrison'`.
8. Evaluate the following Python expressions.
   (a) `(12 * 2) == (3 * 8)`
   (b) `(14 * 2) != (3 * 8)`
9. What value for x makes each of the following Boolean expressions true?
   (a) `x or False`
   (b) `x and True`
   (c) `not (x or False)`
   (d) `not (x and True)`
10. Evaluate the Boolean expressions below for `n = 10` and `k = 20`.
    (a) `(n > 10) and (k == 20)`
    (b) `(n > 10) or (k == 20)`
    (c) `not((n > 10) and (k == 20))`
    (d) `not(n > 10) and not(k == 20)`
    (e) `(n >10) or (k == 10 or k != 5)`
11. Give an appropriate Boolean expression for each of the following.
    (a) Determine if variable `num` is greater than or equal to 0, and less than 100.
    (b) Determine if variable `num` is less than 100 and greater than or equal to 0, or it is equal to 200.
    (c) Determine if either the name `'Thompson'` or `'Wu'` appears in a list of names assigned to variable `last_names`.
    (d) Determine if the name `'Thomson'` appears and the name `'Wu'` does not appear in a list of last names assigned to variable `last_names`.
12. Evaluate the following Boolean expressions for `num1 = 10` and `num2 = 20`.
    (a) `not (num1 < 1) and num2 < 10`
    (b) `not (num1 < 1) and num2 < 10 or num1 + num3 < 100`

13. Give a logically equivalent expression for each of the following.

(a) `num != 25 or num == 0`

(b) `1 <= num and num <= 50`

(c) `not num > 100 and not num < 0`

(d) `(num < 0 or num > 100)`

Section 3.3

14. Give an appropriate if statement for each of the following.

(a) An if statement that displays `'within range'` if num is between 0 and 100, inclusive.

(b) An if statement that displays `'within range'` if num is between 0 and 100, inclusive, and displays `'out of range'` otherwise.

15. Rewrite the following if-else statements using a single if statement and `elif` headers.

```
if temperature >= 85 and humidity > 60:
    print('muggy day today')
else:
    if temperature >= 85:
        print('warm, but not muggy today')
    else:
        if temperature >= 65:
            print('pleasant today')
        else:
            if temperature <= 45:
                print('cold today')
            else:
                print('cool today')
```

16. Regarding proper indentation,

(a) Explain the change in indentation needed in order for the following code to be syntactically correct.

(b) Indicate other changes in the indentation of the code that is not strictly needed, but would make the code more readable.

```
if level <= 1:
    print('Value is well within range')
    print('Recheck in one year')
elif level <= 2:
      print('Value is within range')
      print('Recheck within one month')
elif level <= 3:
      print('Value is slightly high)
        print('Recheck in one week')
elif level <= 4:
      print('Value abnormally high')
      print('Shut down system immediately')
```

Section 3.4

17. Write a program segment that uses a while loop to add up all the even numbers between 100 and 200, inclusive.

18. The following while loop is meant to multiply a series of integers input by the user, until a sentinel value of 0 is entered. Indicate any errors in the code given.

```
product = 1
num = input('Enter first number: ')

while num != 0:
      num = input('Enter first number: ')
      product = product * num
   print('product = ', product)
```

19. For each of the following, indicate which is a definite loop, and which is an indefinite loop.

(a)
```
num = input('Enter a non-zero value: ')
while num == 0:
   num = input('Enter a non-zero value: ')
```

(b)
```
num = 0
while n < 10:
   print 2 ** n
   n = n + 1
```

PYTHON PROGRAMMING EXERCISES

P1. Write a Python program in which the user enters either `'A'`, `'B'`, or `'C'`. If `'A'` is entered, the program should display the word `'Apple'`; if `'B'` is entered, it displays `'Banana'`; and if `'C'` is entered, it displays `'Coconut'`. Use nested if statements for this as depicted in Figure 3-13.

P2. Repeat question P1 using an if statement with `elif` headers instead.

P3. Write a Python program in which a student enters the number of college credits earned. If the number of credits is greater than 90, `'Senior Status'` is displayed; if greater than 60, `'Junior Status'` is displayed; if greater than 30, `'Sophomore Status'` is displayed; else, `'Freshman Status'` is displayed.

P4. Write a program that sums a series of (positive) integers entered by the user, excluding all numbers that are greater than 100.

P5. Write a program, in which the user can enter any number of positive and negative integer values, that displays the number of positive values entered, as well as the number of negative values.

P6. Write a program containing a pair of nested while loops that displays the integer values 1–100, ten numbers per row, with the columns aligned as shown below,

```
 1  2  3  4  5  6  7  8  9 10
11 12 13 14 15 16 17 18 19 20
21 22 23 24 25 26 27 28 29 30
.
.
91 92 93 94 95 96 97 98 99100
```

P7. Display the integer values 1–100 as given in question P6 using only *one* while loop.

PROGRAM MODIFICATION PROBLEMS

M1. Temperature Conversion Program: Input Error Checking
Modify the Temperature Conversion program in Figure 3-19 to perform input error checking of entered temperatures. On the Fahrenheit scale, absolute zero is −459.67. Therefore, all valid Fahrenheit temperatures start at that value (with no upper limit). On the Celsius scale, absolute zero is −273.15. The program should reprompt the user for any invalid entered temperatures.

M2. Temperature Conversion Program: Addition of Kelvin Scale
Modify the Temperature Conversion program in Figure 3-19 to add an additional option of converting to and from degrees Kelvin. The formula for conversion to Kelvin (K) from Celsius (C) is $K = C + 273.15$.

M3. Number of Days in Month Program: Input Error Checking
Modify the Number of Days in Month Program of section 3.3.4 so that the program prompts the user to re-enter any month (not in the range 1–12) or year that is an invalid value.

M4. Number of Days in Month Program: Indication of Leap Years
Modify the Number of Days in Month program of section 3.3.4 so that the program displays, in addition to the number of days in the month, that the year is a leap year or not as shown below.

```
Enter the month (1-12): 2
Please enter the year (e.g., 2010): 2000
There are 29 days in the month (a leap year)
```

M5. Oil Change Notification Program: Number of Miles before Change
Modify the Oil Change Notification program in Figure 3-21 so that the program displays the number of miles left before the next oil change, or the number of miles overdue for an oil change, as appropriate.

M6. Coin Change Exercise Program: Addition of Half-Dollar Coins
Modify the Coin Change Exercise program in section 3.4.6 to allow for the use of half-dollar coins. Make all necessary changes in the program.

M7. Coin Change Exercise Program: Raising the Challenge
Modify the Coin Change Exercise program in section 3.4.6 so that the least possible number of coins must be entered. For example, the least number of coins that total to 43 cents is 6 (one quarter, one dime, one nickel, and three pennies).

M8. Calendar Month Program: Indication of Leap Year
Modify the final version of the Calendar Month program in section 3.5 so that for leap years, the month heading is displayed as in the following,

```
June   1984 (leap year)
                      1   2
  3   4   5   6   7   8   9
 10  11  12  13  14  15  16
 17  18  19  20  21  22  23
 24  25  26  27  28  29  30
```

M9. Calendar Month Program: User Entry of Month Name
Modify the final version of the Calendar Month program to allow the user to enter a month's name (e.g., `'January'`) rather than a number (e.g., `1`). Make all appropriate changes in the program as a result of this change.

M10. Calendar Month Program: Day of the Week Headings
Modify the final version of the Calendar Month program in section 3.5 so that there is day heading for each of the columns as shown below.

```
January   1800

  Su  Mo  Tu  We  Th  Fr  Sa
               1   2   3   4
   5   6   7   8   9  10  11
  12  13  14  15  16  17  18
  19  20  21  22  23  24  25
  26  27  28  29  30  31
```

M11. Sage Program Modification
Following is the output of an "all knowing" Sage program that replies with random responses to questions posed by the user. The responses generated have no meaningful connection to the questions asked.

```
What is your question?:
When will I finally finish the book?

You ask me "When will I finally finish the book?" . . .
You know the answer to that already, don't you?

Do you have another question? (y/n): y
What is your question?:
Do you really know what I am asking?

You ask me "Do you really know what I am asking?" . . .
I would focus my thoughts on something else.

Do you have another question? (y/n): y
What is your question?:
Can't you answer my questions directly?

You ask me "Can't you answer my questions directly?" . . .
The probabilities are in your favor.

Do you have another question? (y/n): y
What is your question?:
So, what's the difference between an orange?

You ask me "So, what's the difference between an orange?" . . .
It is close to certainty.

Do you have another question? (y/n): y
What is your question?:
I think you are bogus, aren't you?

You ask me "I think you are bogus, aren't you?" . . .
Someone you would not expect can be most helpful about this.

Do you have another question? (y/n): n
Goodbye ... hope I was of some help!
>>>
```

(b) How many elements would be looked at when the list is traversed (from top to bottom) until the value 19 was found?

Section 4.2

2. Which of the following lists are syntactically correct in Python?
(a) `[1, 2, 3, 'four']` **(b)** `[1, 2, [3, 4]]` **(c)** `[[1, 2, 3]['four']]`

3. For `lst = [4, 2, 9, 1]`, what is the result of each of the following list operations?
(a) `lst[1]` **(b)** `lst.insert(2, 3)` **(c)** `del lst[3]` **(d)** `lst.append(3)`

4. For `fruit = ['apple', 'banana', 'pear', 'cherry']`, use a list operation to change the list to `['apple', 'banana', 'cherry']`.

5. For a list of integers, `lst`, give the code to retrieve the maximum value of the second half of the list.

6. For variable `product_code` containing a string of letters and digits,
(a) Give an if statement that outputs `"Verified"` if `product_code` contains both a `"Z"` and a `"9"`, and outputs `"Failed"` otherwise.
(b) Give a Python instruction that prints out just the last three characters in `product_code`.

7. Which of the following are valid operations on tuples (for tuples `t1` and `t2`)?
(a) `len(t1)` **(b)** `t1 + t2` **(c)** `t1.append(10)` **(d)** `t1.insert(0, 10)`

8. For `str1 = 'Hello World'`, answer the following,
(a) Give an instruction that prints the fourth character of the string.
(b) Give an instruction that finds the index location of the first occurrence of the letter `'o'` in the string.

9. For a nested list `lst` that contains sublists of integers of the form `[n1, n2, n3]`,
(a) Give a Python instruction that determines the length of the list.
(b) Give Python code that determines how many total integer values there are in list `lst`.
(c) Give Python code that totals all the values in list `lst`.
(d) Given an assignment statement that assigns the third integer of the fourth element (sublist) of `lst` to the value 12.

Section 4.3

10. For a list of integers named `nums`,
(a) Write a while loop that adds up all the values in `nums`.
(b) Write a for loop that adds up all the values in `nums` in which the loop variable is assigned each value in the list.
(c) Write a for loop that adds up all the elements in `nums` in which the loop variable is assigned to the index value of each element in the list.
(d) Write a for loop that displays the elements in `nums` backwards.
(e) Write a for loop that displays every other element in `nums`, starting with the first element.

Section 4.4

11. For `list1 = [1, 2, 3, 4]` and `list2 = [5, 6, 7, 8]`, give the values of `list1[0]` and `list2[0]` where indicated after the following assignments.

(a) `list1[0] = 10`				
`list2[0] = 50`	`list1[0]`	_______	`list2[0]`	_______
(b) `list2 = list1`	`list1[0]`	_______	`list2[0]`	_______
(c) `list2[0] = 15`	`list1[0]`	_______	`list2[0]`	_______
(d) `list1[0] = 0`	`list1[0]`	_______	`list2[0]`	_______

12. Give an appropriate list comprehension for each of the following.

(a) Producing a list of consonants that appear in string variable `w`.

(b) Producing a list of numbers between 1 and 100 that are divisible by 3.

(c) Producing a list of numbers, `zero_values`, from a list of floating-point values, `data_values`, that are within some distance, `epsilon`, from 0.

PYTHON PROGRAMMING EXERCISES

P1. Write a Python program that prompts the user for a list of integers, stores in another list only those values between 1–100, and displays the resulting list.

P2. Write a Python program that prompts the user for a list of integers, stores in another list only those values that are in tuple `valid_values`, and displays the resulting list.

P3. Write a Python program that prompts the user for a list of integers and stores them in a list. For all values that are greater than 100, the string `'over'` should be stored instead. The program should display the resulting list.

P4. Write a Python program that prompts the user to enter a list of first names and stores them in a list. The program should display how many times the letter `'a'` appears within the list.

P5. Write a Python program that prompts the user to enter a list of words and stores in a list only those words whose first letter occurs again within the word (for example, `'Baboon'`). The program should display the resulting list.

P6. Write a Python program that prompts the user to enter types of fruit, and how many pounds of fruit there are for each type. The program should then display the information in the form *fruit, weight* listed in alphabetical order, one fruit type per line as shown below,

```
Apple, 6 lbs.
Banana, 11 lbs.
etc.
```

P7. Write a Python program that prompts the user to enter integer values for each of two lists. It then should displays whether the lists are of the same length, whether the elements in each list sum to the same value, and whether there are any values that occur in both lists.

PROGRAM MODIFICATION PROBLEMS

M1. Chinese Zodiac Program: Japanese and Vietnamese Variations

Modify the Chinese Zodiac program in the chapter to allow the user to select the Chinese Zodiac, the Japanese Zodiac, or the Vietnamese Zodiac. The Japanese Zodiac is the same as the Chinese Zodiac, except that "Pig" is substituted with "Wild Boar." The Vietnamese Zodiac is also the same except that the "Ox" is substituted with "Water Buffalo" and "Rabbit" is replaced with "Cat."

M2. Chinese Zodiac Program: Improved Accuracy

The true Chinese Zodiac does not strictly follow the year that a given person was born. It also depends on the month and date as well, which vary over the years. Following are the correct range of dates for each of the Zodiac symbols for the years 1984 to 2007 (which includes two full cycles of the zodiac). Modify