These results demonstrate that our approximation of the number of seconds in a year was sufficient to get very good results, well within the 99% degree of accuracy required for this program. We would expect more recent dates of birth to give less accurate results given that there is less time that is approximated. Still, for test case December 31, 2009 the inaccuracy is less than .05 percent. Therefore, we were able to develop a program that gave very accurate results without involving all the program logic that would be needed to consider all the details required to give an exact result.

## CHAPTER SUMMARY

### General Topics

Numeric and String Literals
Limitations of Floating-Point Representation
Arithmetic Overflow and Underflow
Character Representation Schemes (Unicode/ASCII)
Control Characters
String Formatting Implicit and Explicit Line Joining/Variables and Variable Use/ Keyboard Input/Identifier Naming/ Keywords
Arithmetic Operators/Expressions/Infix Notation
Operator Precedence and Associativity
Data Types/Static vs. Dynamic Typing
Mixed-Type Expressions/Coercion and Type Conversion

### Python-Specific Programming Topics

Numeric Literal and String Literal Values in Python
Built-in `format` Function in Python
Variable Assignment and Storage in Python
Immutable Values in Python
Identifier Naming and Keywords in Python
Arithmetic Operators in Python
Operator Precedence and Associativity in Python
Built-in `int()` and `float()` Type Conversion Functions in Python

## CHAPTER EXERCISES

### Section 2.1

1. Based on the information in Figure 2-1, how many novels can be stored in one terabyte of storage?
2. Give the following values in the exponential notation of Python, such that there is only one significant digit to the left of the decimal point.
   **(a)** `4580.5034` **(b)** `0.00000046004` **(c)** `5000402.000000000006`
3. Which of the floating-point values in question 2 would exceed the representation of the precision of floating points typically supported in Python, as mentioned in the chapter?
4. Regarding the built-in `format` function in Python,
   **(a)** Use the `format` function to display the floating-point value in a variable named `result` with three decimal digits of precision.
   **(b)** Give a modified version of the format function in (a) so that commas are included in the displayed results.
5. Give the string of binary digits that represents, in ASCII code,
   **(a)** The string `'Hi!'`
   **(b)** The literal string `'I am 24'`
6. Give a call to `print` that is provided one string that displays the following address on three separate lines.

```
John Doe
123 Main Street
Anytown, Maryland 21009
```

7. Use the `print` function in Python to output `It's raining today.`

### Section 2.2

**8.** Regarding variable assignment,

**(a)** What is the value of variables `num1` and `num2` after the following instructions are executed?

```
num = 0
k = 5
num1 = num + k * 2
num2 = num + k * 2
```

**(b)** Are the values `id(num1)` and `id(num2)` equal after the last statement is executed?

**9.** Regarding the `input` function in Python,

**(a)** Give an instruction that prompts the user for their last name and stores it in a variable named `last_name`.

**(b)** Give an instruction that prompts the user for their age and stores it as an integer value named `age`.

**(c)** Give an instruction that prompts the user for their temperature and stores it as a float named `current_temperature`.

**10.** Regarding keywords and other predefined identifiers in Python, give the result for each of the following,

**(a)** `'int' in dir(__builtins__)`

**(b)** `'import' in dir(__builtins__)`

### Section 2.3

**11.** Which of the following operator symbols can be used as both a unary operator and a binary operator?

+, −, *, /

**12.** What is the exact result of each of the following when evaluated?

**(a)** `12 / 6.0`

**(b)** `21 // 10`

**(c)** `25 // 10.0`

**13.** If variable `n` contains an initial value of `1`, what is the largest value that will be assigned to `n` after the following assignment statement is executed an arbitrary number of times?

```
n = (n + 1) % 100
```

**14.** Which of the following arithmetic expressions could potentially result in arithmetic overflow, where `n` and `k` are each assigned integer values?

**(a)** `n * k` **(b)** `n ** k` **(c)** `n / k` **(d)** `n + k`

### Section 2.4

**15.** Evaluate the following expressions in Python.

**(a)** `10 - (5 * 4)`

**(b)** `40 % 6`

**(c)** `- (10 / 3) + 2`

**16.** Give all the possible evaluated results for the following arithmetic expression (assuming no rules of operator precedence).

```
2 * 4 + 25 - 5
```

**17.** Parenthesize all of the subexpressions in the following expressions following operator precedence in Python.

**(a)** `var1 * 8 - var2 + 32 / var3`

**(b)** `var1 - 6 ** 4 * var2 ** 3`

**18.** Evaluate each of the expressions in question 17 above for `var1 = 10`, `var2 = 30`, and `var3 = 2`.

**19.** For each of the following expressions, indicate where operator associativity of Python is used to resolve ambiguity in the evaluation of each expression.

**(a)** `var1 * var2 * var3 - var4`

**(b)** `var1 * var2 / var3`

**(c)** `var1 ** var2 ** var3`

**20.** Using the built-in type conversion function `float()`, alter the following arithmetic expressions so that each is evaluated using floating-point accuracy. Assume that `var1`, `var2`, and `var3` are assigned integer values. Use the minimum number of calls to function `float()` needed to produce the results.

**(a)** `var1 + var2 * var3`

**(b)** `var1 // var2 + var3`

**(c)** `var1 // var2 / var3`

## PYTHON PROGRAMMING EXERCISES

**P1.** Write a Python program that prompts the user for two integer values and displays the result of the first number divided by the second, with exactly two decimal places displayed.

**P2.** Write a Python program that prompts the user for two floating-point values and displays the result of the first number divided by the second, with exactly six decimal places displayed.

**P3.** Write a Python program that prompts the user for two floating-point values and displays the result of the first number divided by the second, with exactly six decimal places displayed in scientific notation.

**P4.** Write a Python program that prompts the user to enter an upper or lower case letter and displays the corresponding Unicode encoding.

**P5.** Write a Python program that allows the user to enter two integer values, and displays the results when each of the following arithmetic operators are applied. For example, if the user enters the values 7 and 5, the output would be,

```
7 + 5 = 12
7 - 5 = 2
7 * 5 = 35
7 / 5 = 1.40
7 // 5 = 1
7 % 5 = 2
7 ** 5 = 16,807
```

All floating-point results should be displayed with two decimal places of accuracy. In addition, all values should be displayed with commas where appropriate.

## PROGRAM MODIFICATION PROBLEMS

**M1.** Modify the Restaurant Tab Calculation program of section 2.2.5 so that, instead of the restaurant tax being hard coded in the program, the tax rate is entered by the user.

**M2.** Modify the Restaurant Tab Calculation program of section 2.2.5 so that, in addition to displaying the total of the items ordered, it also displays the total amount spent on drinks and dessert, as well as the percentage of the total cost of the meal (before tax) that these items comprise. Display the monetary amount rounded to two decimal places.

**M3.** Modify the Your Place in the Universe program in section 2.3.3 for international users, so that the user enters their weight in kilograms, and not in pounds.

**M4.** Modify the Temperature Conversion program in section 2.4.6 to convert from Celsius to Fahrenheit instead. The formula for the conversion is $f = (c * 9/5) + 32$.

**M5.** Modify the Age in Seconds program so that it displays the estimated age in number of days, hours, and minutes.

**M6.** Modify the Age in Seconds program so that it determines the difference in age in seconds of two friends.

## PROGRAM DEVELOPMENT PROBLEMS

**D1.** Losing Your Head over Chess

The game of chess is generally believed to have been invented in India in the sixth century for a ruling king by one of his subjects. The king was supposedly very delighted with the game and asked the subject what he wanted in return. The subject, being clever, asked for one grain of wheat on the first square, two grains of wheat on the second square, four grains of wheat on the third square, and so forth, doubling the amount on each next square. The king thought that this was a modest reward for such an invention. However, the total amount of wheat would have been more than 1,000 times the current world production.

Develop and test a Python program that calculates how much wheat this would be in pounds, using the fact that a grain of wheat weighs approximately 1/7,000 of a pound.

**D2.** All That Talking

Develop and test a Python program that determines how much time it would take to download all instances of every word ever spoken. Assume the size of this information as given in Figure 2-1. The download speed is to be entered by the user in million of bits per second (mbps). To find your actual connection speed, go to the following website (from Intel Corporation) or similar site,

www.intel.com/content/www/us/en/gamers/broadband-speed-test.html

Because connection speeds can vary, run this connection speed test three times. Take the average of three results, and use that as the connection speed to enter into your program. Finally, determine what is an appropriate unit of time to express your program results in: minutes? hours? days? other?

**D3.** Pictures on the Go

Develop and test a Python program that determines how many images can be stored on a given size USB (flash) drive. The size of the USB drive is to be entered by the user in gigabytes (GB). The number of images that can be stored must be calculated for GIF, JPEG, PNG, and TIFF image file formats. The program output should be formatted as given below.

```
Enter USB size (GB): 4

xxxxx images in GIF format can be stored
xxxxx images in JPEG format can be stored
xxxxx images in PNG format can be stored
xxxxx images in TIFF format can be stored
```

The ultimate file size of a given image depends not only on the image format used, but also on the image itself. In addition, formats such as JPEG allow the user to select the degree of compression for the image quality desired. For this program, we assume the image compression ratios given below. Also assume that all the images have a resolution of 800 $\times$ 600 pixels.

Thus, for example, a 800 $\times$ 600 resolution image with 16-bit (2 bytes) color depth would have a total number of bytes of 800 $\times$ 600 $\times$ 2 = 960,000. For a compression rate of 25:1, the total number of bytes needed to store the image would be 960000/25 = 38400.

Finally, assume that a GB (gigabyte) equals 1,000,000,000 bytes, as given in Figure 2.1.

| Format | Full Name | Color Depth | | Compression | |
|---|---|---|---|---|---|
| GIF | Graphics Interchange Format | 256 colors | 8 bits | lossless | 5:1 |
| JPEG | Joint Photographic Experts Group | 16 million colors | 24 bits | lossy | 25:1 |
| PNG | Portable Network Graphics | 16 million colors | 24 bits | lossless | 8:1 |
| TIFF | Tagged Image File Format | 280 trillion colors | 48 bits | lossless | n/a |

Note that a "lossless" compression is one in which no information is lost. A "lossy" compression does lose some of the original information.

**D4.** Life Signs

Develop and test a program that prompts the user for their age and determines approximately how many breaths and how many heartbeats the person has had in their life. The average respiration (breath) rate of people changes during different stages of development. Use the breath rates given below for use in your program:

| | Breaths per Minute |
|---|---|
| Infant | 30–60 |
| 1–4 years | 20–30 |
| 5–14 years | 15–25 |
| adults | 12–20 |

For heart rate, use an average of 67.5 beats per second.

## 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!
>>>
```