A First Book of C++, 4th Edition
Chapter 2 Solutions

Exercises 2.1

1.
   a. `float` or `double`
   b. `int`
   c. `float` or `double`
   d. `int`
   e. `float` or `double`
   f. `char`
2. See solution file pgm2-1ex2.cpp.
3. See solution file pgm2-1ex3.cpp.
4. Students should create a figure similar to Figure 2-3 that includes the following:

8 bytes of storage

```
01001011 01001001 01001110 01000111 01010011 01001100 01000101 01011001
-------- -------- -------- -------- -------- -------- -------- --------
   K        I        N        G        S        L        E        Y
```

5. Answers will vary. Students should create a figure similar to Figure 2-3 using their own last names, as done previously for the name KINGSLEY.
6. See solution file pgm2-1ex6.cpp.
7.
   a. 634000
   b. 195.162
   c. 83.95
   d. 0.00295
   e. 0.0004623
8.
   a. 2670
   b. 0.00267
   c. 1872000000
   d. 0.000000001872
   e. 0.0000000000000000000000000000000066256
9.
   a. 1.26×10^2

b. 6.5623×10^{2}

c. 3.42695×10^{3}

d. 4.8932×10^{3}

e. 3.21×10^{-1}

f. 1.23×10^{-2}

g. 6.789×10^{-3}

10. In some fashion, the program must specify the data type of the different values in the program. Section 2.3 explains how this is accomplished.
11.

a. `h`

b. `r`

c. `m`

d. `A`

e. `i`

f. `f`

g. `F`

12.

a. 536,870,912 bytes

b. 67,108,864 bytes

c. 134,217,728 bytes

d. 2,097,152 bytes

e. 262,144,000 bytes

f. 8,589,934,592 bytes

Exercises 2.2

1.

a. `(2 * 3) + (4 * 5)`

b. `(6 + 18) / 2`

c. `4.5 / (12.2-3.1)`

d. `4.6 * (3.0 + 14.9)`

e. `(12.1 + 18.9) * (15.3 - 3.8)`

2.

a. 27

b. 8

c. 1

d. 220

e. 23

f. 20

g. 6

h. 2

i. 10

j. 1

3.

a. 27.0

b. 8.0

c. 1.0

d. 220.0

e. 23.0

f. 20.0

g. 6.0

h. 2.0

4.

a. 21.3

b. 21.8

c. 9.0

d. 9.67

e. 23.0

f. 65.0

g. 19.67

h. 18.0

i. 16

5.

a. `n / p + 3 = 5`

b. `m / p + n - 10 * amount = 10`

c. `m - 3 * n + 4 * amount = 24`

d. `amount / 5 = 0`

e. `18 / p = 3`

f. `-p * n = -50`

g. `-m / 20 = -2`

h. `(m + n) / (p + amount ) = 10`

i. `m + n / p + amount = 53`

6.

a. `n / p + 3 = 5.0`

b. `m / p + n - 10 * amount = 10.0`

c. `m - 3 * n + 4 * amount = 24.0`

d. `amount / 5 = 0.2`

e. `18 / p = 3.6`

f. `-p * n = -50.0`

g. `-m / 20 = -2.0`

h. `(m + n) / (p + amount ) = 10.0`

i. `m + n / p + amount = 53.0`

Note: These results assume that the preceding variables are the `float` data type.

7. See solution file pgm2-2ex7.cpp.
8. `answer1` is the integer 2, and `answer2` is the integer 5.
9. The remainder of 9 divided by 4 is 1. The remainder of 17 divided by 3 is 2.
10. See solution file pgm2-2ex10.cpp. The manual calculations are as follows:

```
3.0 * 5.0 = 15.0
7.1 * 8.3 - 2.2 = 58.93 - 2.2 = 56.73
3.2 / (6.1 * 5) = 3.2 / 30.5 = 0.104918
```

11. See solution file pgm2-2ex11.cpp. The manual calculations are as follows:

```
15 / 4 = 3
15 % 4 = 3
5 * 3 - (6 * 4) = 5 * 3 - 24 = 15 - 24 = -9
```

Exercises 2.3

1.

`prod_a`	valid
`newbal`	valid
`9ab6`	invalid (begins with a number)
`c1234`	valid
`while`	invalid (C++ keyword)
`sum.of`	invalid (decimal point not allowed)
`abcd`	valid
$`total`	invalid (begins with special character)

`average`	valid
`_c3`	invalid (begins with special character)
`new bal`	invalid (contains a space)
`grade1`	valid
`12345`	invalid (begins with a number)
`a1b2c3d4`	valid
`finGrad`	valid

2.

`salestax`	valid
`harry`	valid (not meaningful)
`maximum`	valid
`3sum`	invalid (begins with a number)
`a243`	valid (not meaningful)
`sue`	valid (not meaningful)
`okay`	valid
`for`	invalid (C++ keyword)
`r2d2`	valid (not meaningful)
`c3p0`	valid (not meaningful)
`a`	valid (not meaningful)
`tot.al`	invalid (contains decimal point)
`firstNum`	valid
`average`	valid
`awesome`	valid (not meaningful)
`c$five`	invalid (contains a special character)
`cc_al`	valid (not meaningful)
`sum`	valid
`goforit`	valid (not meaningful)
`netpay`	valid

3.

a. `int count;`

b. `float grade;`

c. `double yield;`

d. `char initial;`

4.

a. `int num1, num2, num3;`

b. `double grade 1, grade2, grade3, grade4;`

c. `double temp1, temp2, temp3;`

d. `char let1, let2, let3, let4;`

5.

a. `int firstnum, secnum;`

b. `double price, yield, coupon;`

c. `double average;`

6.

a.
```
int month;
int day = 30;
int year;
```

b.
```
double hours;
double volt;
double power = 15.62;
```

c.
```
double price;
double amount;
double taxes;
```

d.
```
char inKey;
char ch;
char choice = 'f';
```

7.

a.
```
#include <iostream>     // preprocessor directive
using namespace std;    // location of header file

int main()                  // header line for main() function
{                           // opening brace for main() function
  int num1, num2, total;   // integer variable declarations

num1 = 25;     // assign the value 25 to the variable num1
  num2 = 30;   // assign the value 30 to the variable num2
  total = num1 + num2;    // assign the sum of num1 and num2 to total
  cout    <<  "The total of  "  << num1 << " and "
          << num2  << " is " << total << endl;    // output statement
  return 0;    // return 0 to calling program
}              // closing brace for main() function
```

b. The total of 25 and 30 is 55.

8. See solution file pgm2-3ex8.cpp
9. See solution file pgm2-3ex9.cpp
10. See solution file pgm2-3ex10.cpp
11. See solution file pgm2-3ex11.cpp. The average is 16.5, but the program stores 16 in the variable `average`. To ensure that the correct answer is displayed, the variable `average` must be declared as a `float` or `double` value.
12. See solution file pgm2-3ex12.cpp. This program computes an incorrect average of 16. The reason is that although `average` is declared as a `double`, the expression `total / 2` is an integer expression that yields a result of 16. Assigning 16 to a `double` produces an answer of 16.0. The solution is to declare both `total` and `average` as `double`s or leave `total` as an integer and divide by the real value 2.0.
13. See solution file pgm2-3ex13.cpp
14.

    a. All definition statements are declaration statements, but not all declaration statements are definition statements.

    b. The definition statement sets up the required storage area in memory for the variable. In other words, the variable comes into existence because of the definition statement and can't be used until it comes into existence.

15.

    a. Address: 159 160 161 162 163 164 165 166

    | | | | | M | E | L | T |

    rate ch1 ch2 ch3 ch4

    Address: 167 168 169 170 171 172 173 174

    | | | | | | | | |

    taxes

    Address: 175 176 177 178 179 180 181 182

    | | | | | 0 | | | |

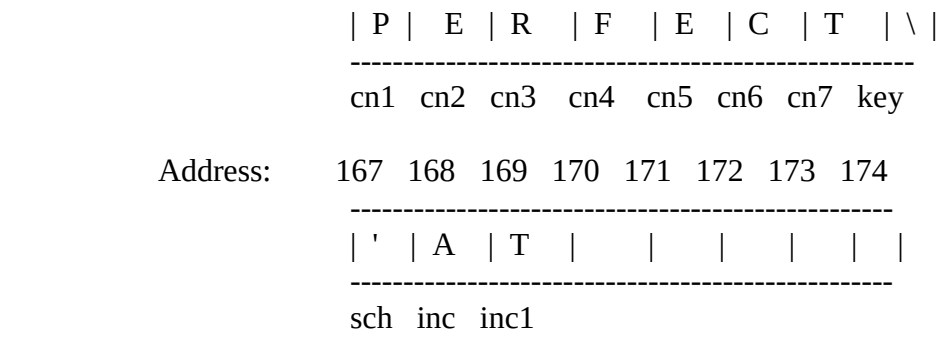
    num count

    b.

```
ch1 = 01001101
ch2 = 01000101
ch3 = 01001100
ch4 = 01010100
```

16.

    a. Address: 159 160 161 162 163 164 165 166



b.

```
cn1=01010000
cn2=01000101
cn3=01010010
cn4=01010010
cn5=01000110
cn6=01100011
cn7=01101000
key=01011100
sch=01011100
inc=01101111
inc1=01100110
```

17.