Solutions to Chapter 7

Review Questions

1. a. True

3. b. False

5. a. True

7. c. `stdfile`

9. a. Opens the file for reading and sets the file marker at the beginning.

11. a. Conversion code

13. a. *scanf* implicitly casts data to match the conversion code and its corresponding address parameter

15. e. Terminates the *scanf* function and leaves the invalid character in the input stream.

Exercises

17.

```
 i1 = 14  i2 = 67  f1 = 67.9  c1 = '.'  c2 = 2
```

19.

```
000123, -00234, -00007
-234   , %, ", \t, 123   A 10 a H 0xf
```

Problems

21. See Program 7-1.

Program 7-1 Solution to Problem 21

```
/* ==================== appendFile ====================
   This function appends one file to the other.
     Pre   Files exist
     Post  File 2 appended to file 1.
           Returns 0 if successful
                   1 if file 1 cannot be opened
                   2 if file 2 cannot be opened
*/
int appendFile (const char* file1, const char* file2)
{
// Local Declarations
   char c;
   FILE* sp1;
   FILE* sp2;

// Statements
   if (!(sp1 = fopen (file1, "a")))
      {
       printf ("\nError opening %s for appending.\n",
               file1);
       return (1);
      } // if
```

Program 7-1 Solution to Problem 21 (continued)

```
   if (!(sp2 = fopen (file2, "r")))
      {
       printf ("\nError opening %s for reading.\n",
                file2);
       return (2);
      } // if

   while ((c = fgetc (sp2)) != EOF)
       fputc (c, sp1);

   fclose (sp1);
   fclose (sp2);
   return 0;
}  // appendFile
```

23. See Program 7-2.

Program 7-2 Solution to Problem 23

```
/* ==================== formatLine ====================
   This function reformats a file to 60 characters per
   line.
      Pre   File exists
      Post  File reformatted 60 characters per line
*/

int formatLine (const char* filename)
{
// Local Declarations
   char  c;
   char* tempfile = "TEMP.DAT";
   int   count    = 0;
   FILE* sp;
   FILE* sptemp;

// Statements
   if (!(sp = fopen (filename, "r")))
      {
       printf ("\nError opening %s for reading.\n",
               filename);
       return (1);
      } // if open error

   if (!(sptemp = fopen (tempfile, "w")))
      {
       printf ("\nError opening %s for writing.\n",
               tempfile);
       return (2);
      } // if open error

   while ((c = fgetc (sp)) != EOF)
         {
          if (count == 60)
              {
               fputc ('\n', sptemp);
               count = 0;
              } // if count
          if (c != '\n')
              {
               fputc (c, sptemp);
               count++;
              } // if not \n
         } // while
   fputc ('\n', sptemp);
```

Program 7-2 Solution to Problem 23 (continued)

```
   fclose (sp);
   fclose (sptemp);

   if (!(sp = fopen (filename, "w")))
      {
       printf ("\nError opening %s for writing.\n",
                filename);
       return (1);
      } // if open error

   if (!(sptemp = fopen (tempfile, "r")))
      {
       printf ("\nError opening %s for reading.\n",
                tempfile);
       return (2);
      } // if open error

   while ((c = fgetc (sptemp)) != EOF)
       fputc (c, sp);

   fclose (sp);
   fclose (sptemp);
   return 0;
}  // formatLine
```

25. See Program 7-3.

Program 7-3 Solution to Problem 25

```
/* ==================== delLastLine ===================
   This function deletes the last line of any file.
      Pre   File exists
      Post  Last line of file deleted.
*/
int delLastLine (const char* filename)
{
// Local Declarations
   char  c;
   char* tempfile    = "TEMP.DAT";
   int   line_count  = 1;
   int   total_lines = 1;
   FILE* sp;
   FILE* sptemp;

// Statements
   if (!(sp = fopen (filename, "r")))
      {
       printf ("\nError opening %s.\n", filename);
       return (1);
      } // if open error

   if (!(sptemp = fopen (tempfile, "w")))
      {
       printf ("\nError opening %s.\n", tempfile);
       return (2);
      } // if open error

   // count the number of lines and build temp file
   while ((c = fgetc (sp)) != EOF)
      {
       if (c == '\n')
           total_lines++;
       fputc(c, sptemp);
      } // while

   fclose (sp);
```

Program 7-3 Solution to Problem 25 (continued)

```
   fclose (sptemp);

   if (!(sp = fopen (filename, "w")))
      {
       printf ("\nError opening %s for writing.\n",
                filename);
       return (1);
      } // if open error

   if (!(sptemp = fopen (tempfile, "r")))
      {
       printf ("\nError opening %s for reading.\n",
                tempfile);
       return (2);
      } // if open error

   // Write all but last line to the original file.
   while ((c = fgetc (sptemp)) != EOF
       &&  line_count < total_lines - 1)
      {
       fputc (c, sp);
       if (c == '\n')
          line_count++;
      } // while

   fclose (sp);
   fclose (sptemp);

   return 0;
}  // delLastLine
```

27. See Program 7-4.

Program 7-4 Solution to Problem 27

```
/* This program prints itself.
      Written by:
      Date:
*/
#include <stdio.h>

int main (void)
{
// Local Declarations
   char  c;
   FILE* sp;

// Statements
   printf ("Start of Program\n\n");
   if (!(sp = fopen ("P07-27.c", "r")))
      {
       printf ("\nError opening P07-27.c.\n");
       return (1);
      } // if open error

   while ((c = fgetc(sp)) != EOF)
          printf ("%c", c);

   fclose (sp);
   printf ("End of Program\n\n");
   return 0;
}  // main
```

29. See Program 7-5.

Program 7-5 Solution to Problem 29

```
/* This program copies only the line started with a
   specified character.
      Written by:
      Date:
*/
#include <stdio.h>

#define FILE_1 "FILE1.DAT"
#define FILE_2 "FILE2.DAT"

int main (void)
{
// Local Declarations
   char  target;
   char  cur;
   char  pre = '\n';
   int   writeLn = 0;
   FILE* sp1;
   FILE* sp2;

// Statements
   printf ("Start of Program\n\n");

   printf ("Please enter the target character:  ");
   scanf  ("%c", &target);

   if (!(sp1 = fopen (FILE_1, "r")))
      {
       printf ("\nError opening %s.\n", FILE_1);
       return (1);
      } // if open error

   if (!(sp2 = fopen (FILE_2, "w")))
      {
       printf ("\nError opening %s.\n", FILE_2);
       return (2);
      } // if open error

   while ((cur = fgetc(sp1)) != EOF)
      {
       if (!writeLn && pre == '\n' && cur == target)
          writeLn = 1;
       if (writeLn)
          fputc (cur, sp2);
       if (writeLn && cur == '\n')
          writeLn = 0;
       pre = cur;
      } // while

   fclose (sp1);
   fclose (sp2);
   printf ("\nEnd of Program\n");
   return 0;
}  // main
```

31. See Program 7-6.

Program 7-6 Solution to Problem 31

```
/* ==================== handleError ===================
   This function reads 3 pieces of numeric data, and
   checks if the input is correct or not.
      Pre   nothing
```

Program 7-6 Solution to Problem 31 (continued)

```
      Post  Valid input printed
*/
#define FLUSH while (getchar() != '\n')

void handleError (void)
{
// Local Declarations
   int num1;
   int num2;
   int num3;

// Statements
   printf ("\nPlease enter 3 integers: ");
   while  (scanf ("%d %d %d", &num1, &num2, &num3) < 3)
      {
       FLUSH;
       printf ("\nInvalid input. Please re-enter: ");
      } // while

   printf ("\nThe 3 input data is %d, %d, %d\n",
                num1, num2, num3);
   return;
}  // handleError
```

33. See Program 7-7.

Program 7-7 Solution to Problem 33

```
/* This program deletes the sixth line in a file.
      Written by:
      Date:
*/
#include <stdio.h>

#define FILE_1 "FILE1.DAT"
#define FILE_2 "TEMP.DAT"

int main (void)
{
// Local Declarations
   char  ch;
   int   line_cnt = 0;
   FILE* sp1;
   FILE* sp2;

// Statements
   printf ("Start of Program\n\n");

   if (!(sp1 = fopen (FILE_1, "r")))
      {
       printf ("\nError opening %s.\n", FILE_1);
       return (1);
      } // if open error

   if (!(sp2 = fopen (FILE_2, "w")))
      {
       printf ("\nError opening %s.\n", FILE_2);
       return (2);
      } // if open error

   while ((ch = fgetc(sp1)) != EOF)
      {
       if (ch == '\n')
           line_cnt++;
       if (line_cnt != 5)
           fputc (ch, sp2);
```

Program 7-7 Solution to Problem 33 (continued)

```
      } // while

   fclose (sp1);
   fclose (sp2);

   if (!(sp1 = fopen (FILE_1, "w")))
      {
       printf ("\nError opening %s for writing.\n",
                FILE_1);
       return (1);
      } // if open error

   if (!(sp2 = fopen (FILE_2, "r")))
      {
       printf ("\nError opening %s for reading.\n",
                FILE_2);
       return (2);
      } // if open error

   while ((ch = fgetc (sp2)) != EOF)
       fputc (ch, sp1);

   fclose (sp1);
   fclose (sp2);
   printf ("\nEnd of Program\n");
   return 0;
}  // main
```

35. See Program 7-8.

Program 7-8 Solution to Problem 35

```
/* This program duplicates the fourth line in a file.
      Written by:
      Date:
*/
#include <stdio.h>

#define FILE_1 "FILE1.DAT"
#define FILE_2 "TEMP1.DAT"
#define FILE_3 "TEMP2.DAT"

int main (void)
{
// Local Declarations
   char  c;
   int   line_cnt = 0;
   FILE* sp1;
   FILE* sp2;
   FILE* sp3;

// Statements
   printf ("Start of Program\n\n");

   if (!(sp1 = fopen (FILE_1, "r")))
      {
       printf ("\nError opening %s.\n", FILE_1);
       return (1);
      } // if open error

   if (!(sp2 = fopen (FILE_2, "w")))
      {
       printf ("\nError opening %s.\n", FILE_2);
       return (2);
      } // if open error
```

Program 7-8 Solution to Problem 35 (continued)

```
   if (!(sp3 = fopen (FILE_3, "w")))
      {
       printf ("\nError opening %s for writing.\n",
                FILE_3);
       return (3);
      } // if open error
   while ((c = fgetc(sp1)) != EOF)
      {
       if (c == '\n')
           line_cnt++;
       if (line_cnt == 3)
           fputc (c, sp3);
       if (line_cnt == 4 && c == '\n')
          {
           fclose (sp3);
           if (!(sp3 = fopen (FILE_3, "r")))
              {
               printf ("\nError 4 opening %s.\n",
                        FILE_3);
               return (4);
              } // if
           while ((c = fgetc(sp3)) != EOF)
               fputc (c, sp2);
           c = '\n';
           fclose(sp3);
          } // if line_cnt
       fputc (c, sp2);
      } // while

   fclose (sp1);
   fclose (sp2);

   if (!(sp1 = fopen (FILE_1, "w")))
      {
       printf ("\nError opening %s for writing.\n",
                FILE_1);
       return (1);
      } // if open error

   if (!(sp2 = fopen (FILE_2, "r")))
      {
       printf ("\nError opening %s for reading.\n",
                FILE_2);
       return (2);
      } // if open error

   while ((c = fgetc (sp2)) != EOF)
       fputc (c, sp1);

   fclose (sp1);
   fclose (sp2);
   printf ("\nEnd of Program\n");
   return 0;
}  // main
```

37. See Program 7-9.

Program 7-9 Solution to Problem 37

```
/* This program copies a file, inserting two space
   characters at the beginning of each line.
      Written by:
      Date:
*/
#include <stdio.h>
```

Program 7-9 Solution to Problem 37 (continued)

```
#define FILE_1 "FILE1.DAT"
#define FILE_2 "FILE2.DAT"

int main (void)
{
// Local Declarations
   char  cur;
   char  pre = '\n';
   FILE* sp1;
   FILE* sp2;

// Statements
   printf ("Start of Program\n\n");

   if (!(sp1 = fopen (FILE_1, "r")))
      {
       printf ("\nError opening %s.\n", FILE_1);
       return (1);
      } // if open error

   if (!(sp2 = fopen (FILE_2, "w")))
      {
       printf ("\nError opening %s.\n", FILE_2);
       return (2);
      } // if open error

   while ((cur = fgetc(sp1)) != EOF)
      {
       if (pre == '\n')
          {
           fputc (' ', sp2);
           fputc (' ', sp2);
          } // if
       fputc (cur, sp2);
       pre = cur;
      } // while

   fclose (sp1);
   fclose (sp2);
   printf ("\nEnd of Program\n");
   return 0;
}  // main
```

39. See Program 7-10.

Program 7-10 Solution to Problem 39

```
/* This program writes the odd numbers between 300
   and 500 to a text file.
      Written by:
      Date:
*/
#include <stdio.h>

#define FILE_1 "FILE1.DAT"

int main (void)
{
// Local Declarations
   int   number;
   FILE* sp1;

// Statements
   printf ("Start of Program\n\n");

   if (!(sp1 = fopen (FILE_1, "w")))
```

Program 7-10 Solution to Problem 39 (continued)

```
        {
         printf ("\nError opening %s.\n", FILE_1);
         return (1);
        }  // if open error

     for (number = 301; number < 500; number += 2)
        fprintf (sp1, "%d\n", number);

     fclose (sp1);
     printf ("\nEnd of Program\n");
     return 0;
  }  // main
```

41. See Program 7-11.

Program 7-11 Solution to Problem 41

```
/* This program reads a file of scores and creates a new
   file of all of the scores over 90.
      Written by:
      Date:
*/
#include <stdio.h>

#define FILE_1 "FILE1.DAT"
#define FILE_2 "FILE2.DAT"

int main (void)
{
// Local Declarations
   int   score;
   int   count = 0;
   FILE* sp1;
   FILE* sp2;

// Statements
   printf ("Start of Program\n\n");

   if (!(sp1 = fopen (FILE_1, "r")))
      {
       printf ("\nError opening %s.\n", FILE_1);
       return (1);
      }  // if open error

   if (!(sp2 = fopen (FILE_2, "w")))
      {
       printf ("\nError opening %s.\n", FILE_2);
       return (2);
      }  // if open error

   while ((fscanf(sp1, "%d", &score)) != EOF)
      {
       if (score >= 90)
          {
           fprintf (sp2, "%d\n", score);
           count++;
          } // if
      } // while

   printf ("\nThere were %d scores over 90.\n", count);

   fclose (sp1);
   fclose (sp2);
   printf ("\nEnd of Program\n");
   return 0;
}  // main
```