11-2 C Strings

A C string is a variable-length array of characters that is delimited by the null character.

Topics discussed in this section:

Storing Strings
The String Delimiter
String Literals
Strings and Characters
Declaring Strings
Initializing Strings
Strings and the Assignment Operator
Reading and Writing Strings

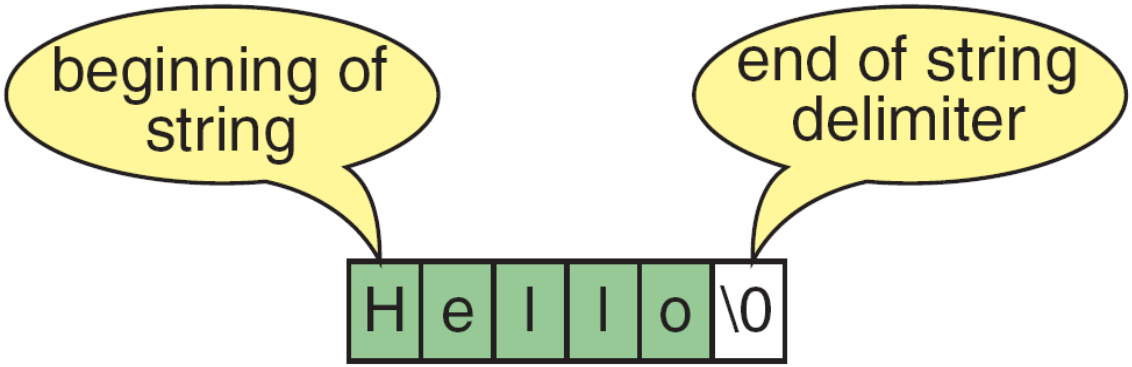

FIGURE 11-3 **Storing Strings**

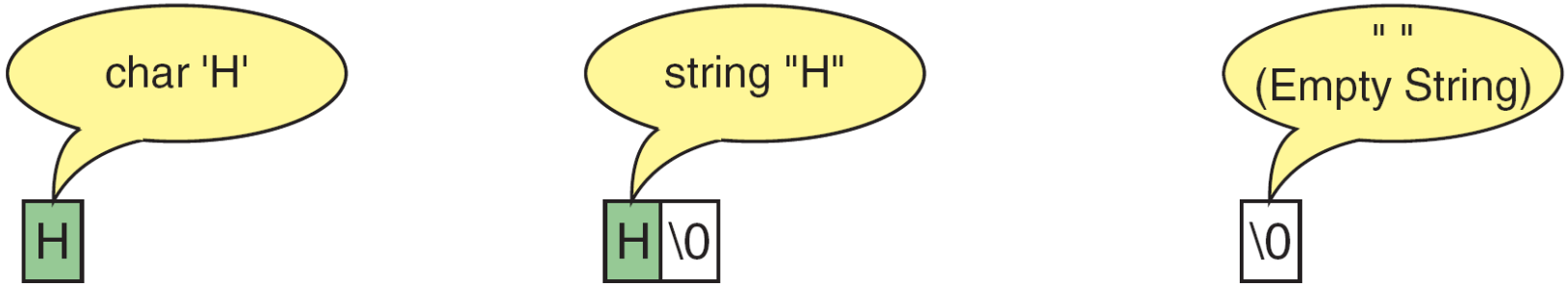

FIGURE 11-4 **Storing Strings and Characters**

FIGURE 11-5 Differences Between Strings and Character Arrays

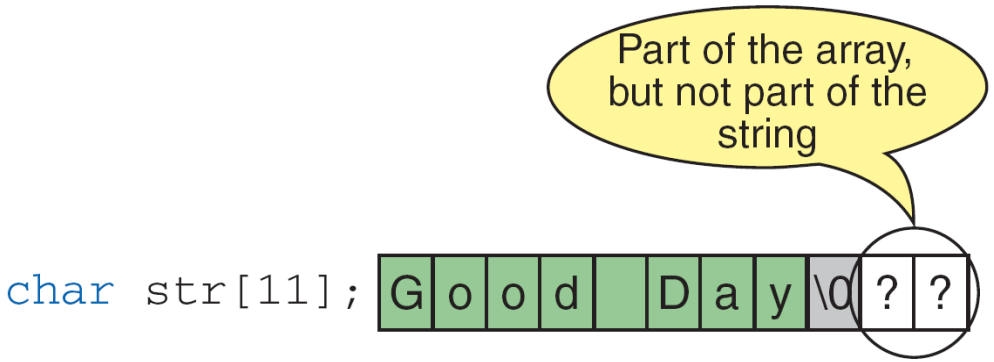

FIGURE 11-6 Strings in Arrays

Note

A string literal is enclosed in double quotes.

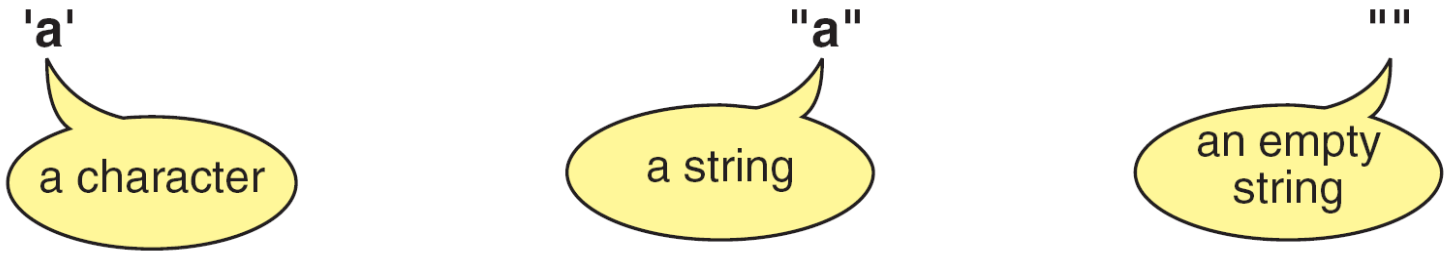

FIGURE 11-7 Character Literals and String Literals

```c
#include <stdio.h>
int main (void)
{
  printf("%c\n", "Hello"[1];
  return 0;
} // main
```

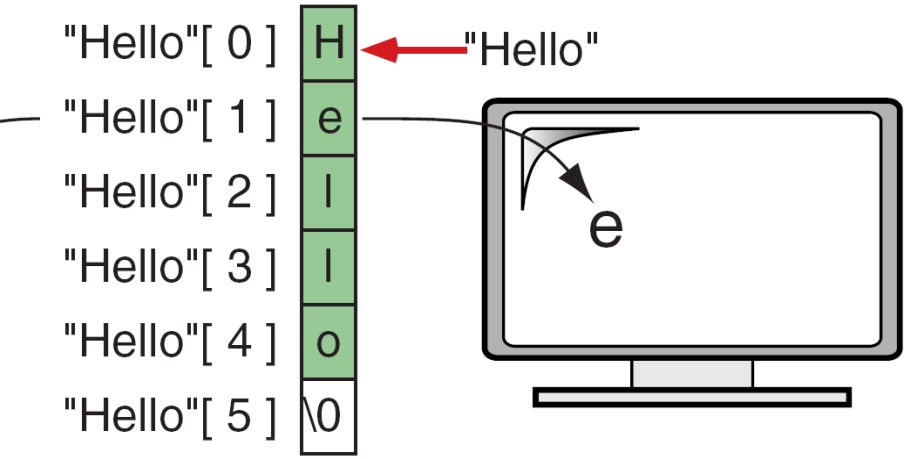

FIGURE 11-8 String Literal References

```
// Local Declarations
   char str[9];
```

(a) String Declaration

```
// Local Declarations
   char* pStr;
```

(b) String Pointer Declaration

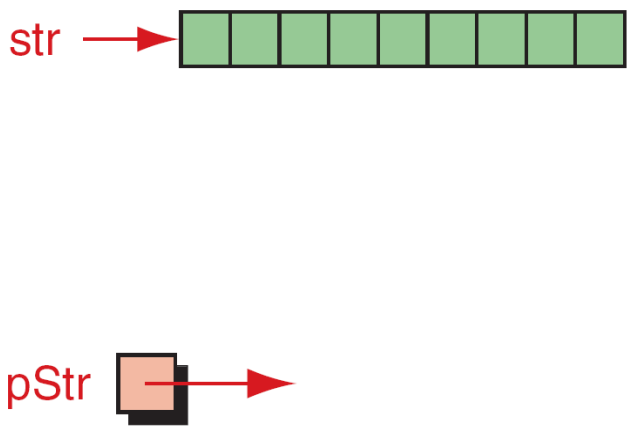

FIGURE 11-9 Defining Strings

Note

Memory for strings must be allocated before the string can be used.

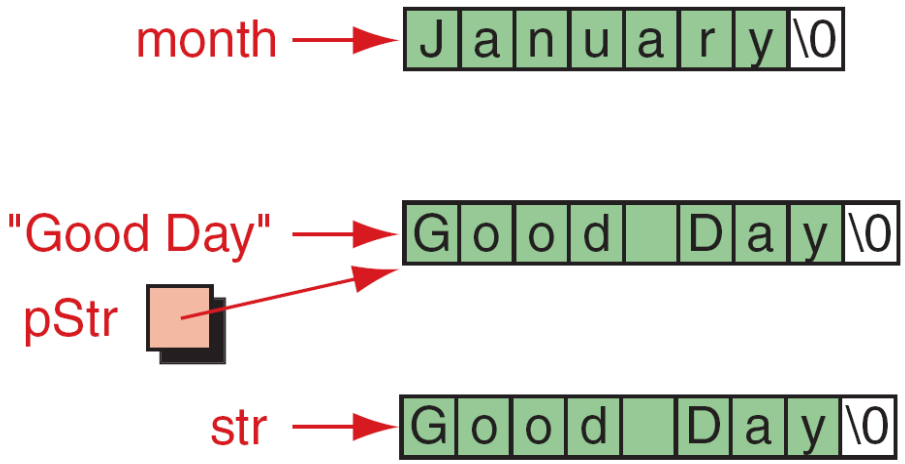

FIGURE 11-10 **Initializing Strings**

11-3 String Input/Output Functions

C provides two basic ways to read and write strings. First, we can read and write strings with the formatted input/output functions, *scanf/fscanf* ***and*** *printf/fprintf.* ***Second, we can use a special set of string-only functions, get string*** *(gets/fgets)* ***and put string*** *(puts/fputs).*

Topics discussed in this section:

Formatted String Input/Output

String Input/Output

Note

The string conversion code(s) skips whitespace.

PROGRAM 11-1 Reading Strings

```
1  {  // Read Month
2     #define FLUSH while (getchar() != '\n')
3     char month[10];
4
5     printf("Please enter a month. ");
6     scanf("%9s", month);
7     FLUSH;
8  }  // Read Month
```

Note

Always use a width in the field specification when reading strings.

Note

The maximum number of characters to be printed is specified by the precision in the format string of the field specification.

PROGRAM 11-2 Demonstrate String Scan Set

```
 1  /* Read only second integer.
 2        Written by:
 3        Date:
 4  */
 5  #include <stdio.h>
 6  #include <stdlib.h>
 7
 8  int main (void)
 9  {
10  // Local Declarations
11     int   amount;
12     FILE* spData;
13
14  // Statements
15     if (!(spData = fopen ("P11-03.TXT", "r")))
16        {
17         printf("\aCould not open input file.\n");
18         exit (100);
19        } // if
```

PROGRAM 11-2 Demonstrate String Scan Set

```
20      // Read and print only second integer
21      while (fscanf(spData,
22             " %*d%d%*[^\n]", &amount) != EOF)
23          printf("Second integer: %4d\n", amount);
24
25      printf("End of program\n");
26      fclose (spData);
27      return 0;
28  }   // main
```

```
Input:
123 456 7.89
987 654 3.21

Results:
Second integer:  456
Second integer:  654
End of program
```

PROGRAM 11-3 Delete Leading Whitespace

```
1  /* Delete leading spaces at beginning of line.
2        Written by:
3        Date:
4  */
5  #include <stdio.h>
6  #include <ctype.h>
7
8  int main (void)
9  {
10 // Local Declarations
11    char line[80];
12
13 // Statements
14    printf("Enter data:  ");
15    while ((fscanf(stdin, "%*[\t\v\f ]%79[^\n]", line))
16                          != EOF)
17       {
18        printf("You entered: %s\n", line);
19
```

PROGRAM 11-3 Delete Leading Whitespace

```
20          // Discard newline and set line to null string
21          fgetc (stdin);
22          *(line) = '\0';
23          printf("Enter data:  ");
24        } // while
25
26     printf("\nThank you\n");
27     return 0;
28  }  // main
```

```
Results:
Enter data:  No whitespace here.
You entered: No whitespace here.
Enter data:   Only one whitespace character.
You entered: Only one whitespace character.
Enter data:         Tabs and spaces here.
You entered: Tabs and spaces here.
Enter data:  Next line is only one space.
You entered: Next line is only one space.
Enter data:
You entered:
Enter data:  ^d
Thank you
```

PROGRAM 11-4 Read Student Names and Scores

```
 1  /* Demonstrate reading names from a file.
 2        Written by:
 3        Date:
 4  */
 5  #include <stdio.h>
 6  #include <stdlib.h>
 7  #include <string.h>
 8
 9  int main (void)
10  {
11  // Local Declarations
12     char  first[80];
13     char  last[80];
14     int   score;
15     FILE* spStuScores;
16
17  // Statements
18     if (!(spStuScores = fopen ("P11-04.TXT", "r")))
19        {
```

PROGRAM 11-4 Read Student Names and Scores

```
20          printf("\aCould not open student file.\n");
21          exit (100);
22        } // if
23
24     // Read and print first name, last name, and score
25     while (fscanf(spStuScores, " %s %s %d",
26              first, last, &score) == 3)
27        printf("%s %s %3d\n", first, last, score);
28
29     printf("End of Student List\n");
30     fclose (spStuScores);
31     return 0;
32  } // main
```

```
Results:
George Washington  95
Benedict Arnold  53
Mary Todd-Lincoln  91
End of Student List
```

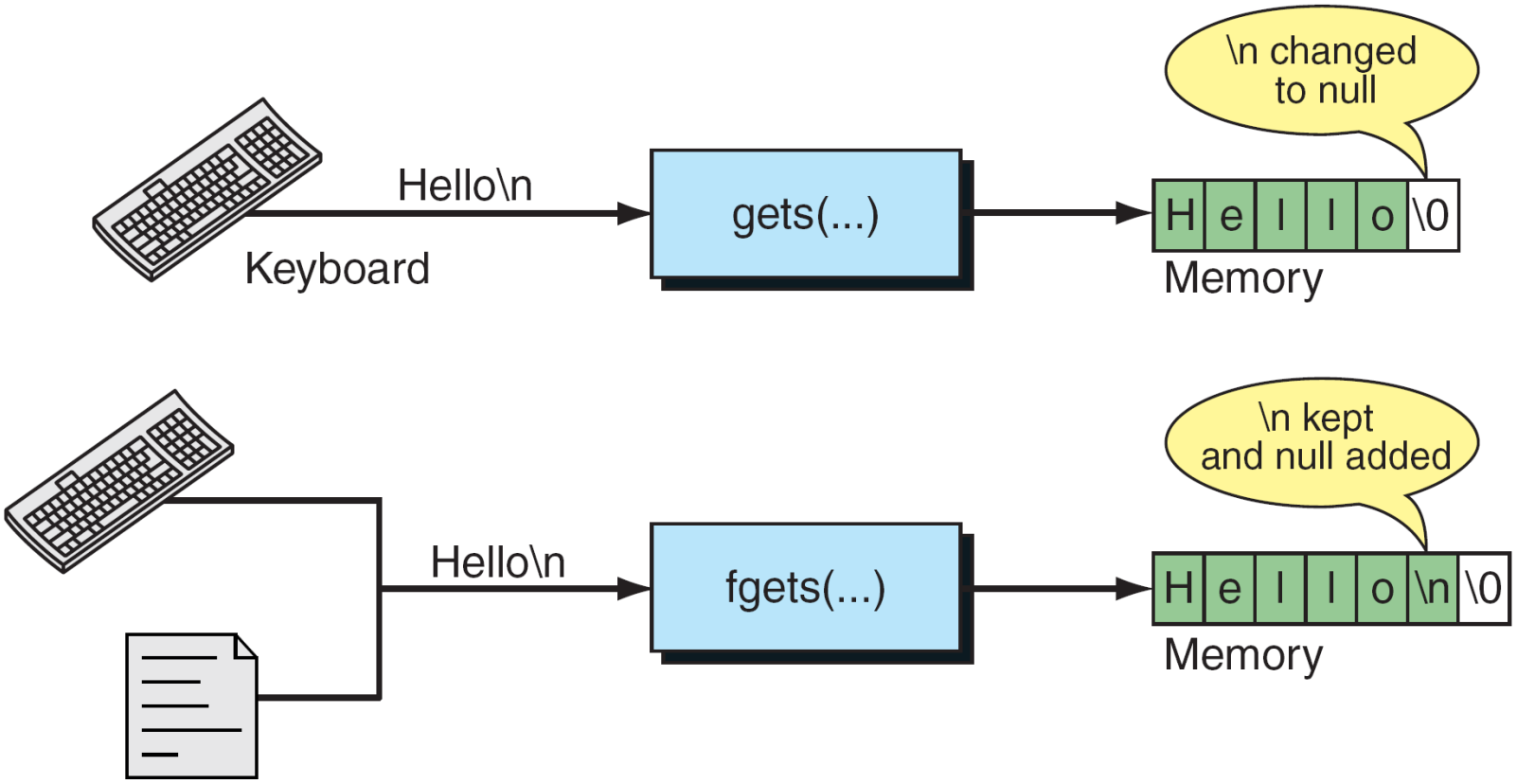

FIGURE 11-11 ***gets*** **and** ***fgets*** **Functions**

PROGRAM 11-5 Demonstrate *fgets* Operation

```
 1  /* Demonstrate the use of fgets in a program
 2        Written by:
 3        Date:
 4  */
 5  #include <stdio.h>
 6
 7  int main (void)
 8  {
 9  // Local Declarations
10     char str[81];
11
12  // Statements
13     printf("Please enter a string: ");
14     fgets (str, sizeof (str), stdin);
15     printf("Here is your string: \n\t%s", str);
16     return 0;
17  }  // main
```

PROGRAM 11-5 Demonstrate *fgets* Operation

```
Results:
Please enter a string: Now is the time for all students
Here is your string:
    Now is the time for all students
```

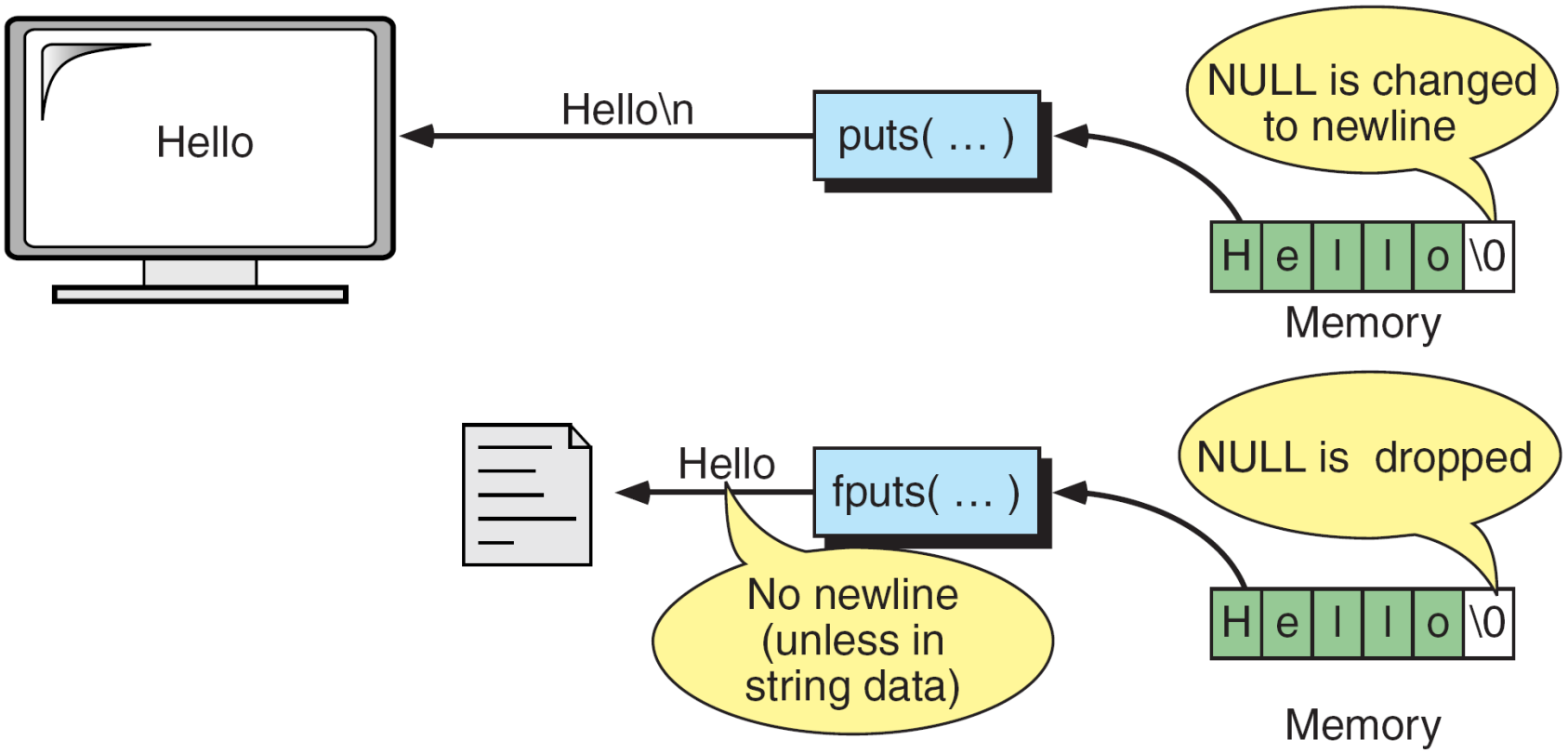

FIGURE 11-12 ***puts*** **and** ***fputs*** **Operations**

PROGRAM 11-6 Demonstration of Put String

```
1  /* Demonstrate fput string
2        Written by:
3        Date:
4  */
5  #include <stdio.h>
6
7  int main (void)
8  {
9  // Local Definitions
10    char str[] = "Necessity Is the Mother of Invention.";
11    char* pStr = str;
12
13 // Statements
14    fputs(pStr, stdout);
15    fputs("\n", stdout);
16    fputs(pStr + 13, stdout);
17    return 0;
18 }  // main
```

PROGRAM 11-6 Demonstration of Put String

```
Results:
Necessity Is the Mother of Invention
the Mother of Invention.
```

PROGRAM 11-7 Typewriter Program

```
1  /* This program creates a text file from the keyboard.
2        Written by:
3        Date:
4  */
5  #include <stdio.h>
6  #include <stdlib.h>
7
8  int main (void)
9  {
10 // Local Declarations
11    char  str[100];
12    FILE* spOut;
13
14 // Statements
15    if (!(spOut = fopen ("P11-07.TXT", "w")))
16       {
17        printf("\aCould not open output file.\n");
18        exit (100);
19       } // if
```

PROGRAM 11-7 Typewriter Program

```
20      while (fgets(str, sizeof (str), stdin))
21          fputs(str,  spOut);
22      fclose (spOut);
23      return 0;
24  }   // main
```

PROGRAM 11-8 Print Selected Sentences

```
1  /* Echo keyboard input that begins with capital letter.
2        Written by:
3        Date written:
4  */
5  #include <ctype.h>
6  #include <stdio.h>
7
8  int main (void)
9  {
10 // Local Declarations
11    char strng[81];
12
13 // Statements
14    while (fgets (strng, sizeof(strng), stdin))
15        if (isupper(*strng))
16            fputs(strng, stdout);
17    return 0;
18 }  // main
```

PROGRAM 11-8 Print Selected Sentences

```
Results:
Now is the time
Now is the time
for all good students
to come to the aid
of their school.
Amen
Amen
```

PROGRAM 11-9 Print File Double spaced

```
 1  /* Write file double spaced.
 2        Written by:
 3        Date:
 4  */
 5  #include <stdio.h>
 6  #include <stdlib.h>
 7
 8  int main (void)
 9  {
10  // Local Declarations
11     char  strng[81];
12     FILE* textIn;
13
14  // Statements
15     if (!(textIn = fopen("P11-07.TXT", "r")))
16        {
17         printf("\aCan't open textdata\n");
18         exit (100);
19        } // if
```

PROGRAM 11-9 Print File Double spaced

```
20      while (fgets(strng, sizeof(strng), textIn))
21          {
22           fputs(strng, stdout);
23           putchar ('\n');
24          } // while
25      return 0;
26  }   // main
```

11-4 Arrays of Strings

When we discussed arrays of pointers in Chapter 10, we introduced the concept of a ragged array. Ragged arrays are very common with strings. Consider, for example, the need to store the days of the week in their textual format. We could create a two-dimensional array of seven days by ten characters, but this wastes space.

PROGRAM 11-10 Print Days of the Week

```
1  /* Demonstrates an array of pointers to strings.
2        Written by:
3        Date written:
4  */
5  #include <stdio.h>
6
7  int main (void)
8  {
9  // Local Declarations
10    char*  pDays[7];
11    char** pLast;
12
13 // Statements
14    pDays[0] = "Sunday";
15    pDays[1] = "Monday";
16    pDays[2] = "Tuesday";
17    pDays[3] = "Wednesday";
18    pDays[4] = "Thursday";
```

PROGRAM 11-10 Print Days of the Week

```
19      pDays[5] = "Friday";
20      pDays[6] = "Saturday";
21
22      printf("The days of the week\n");
23      pLast = pDays + 6;
24      for (char** pWalker = pDays;
25                  pWalker <= pLast;
26                  pWalker++)
27           printf("%s\n", *pWalker);
28      return 0;
29  } // main
```

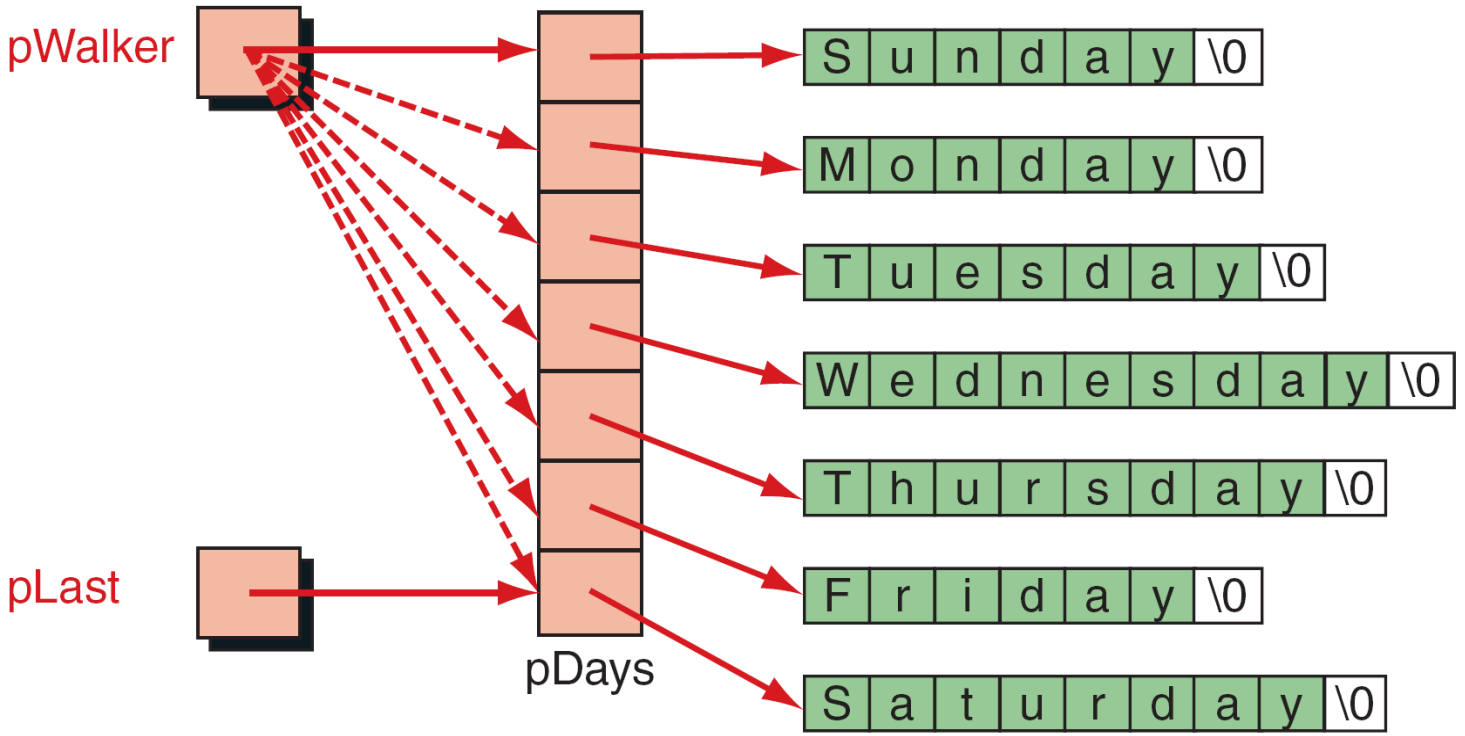

FIGURE 11-13 Pointers to Strings