# 11-5 String Manipulation Functions

***Because a string is not a standard type, we cannot use it directly with most C operators. Fortunately, C provides a set of functions to manipulates strings.***

***Topics discussed in this section:***

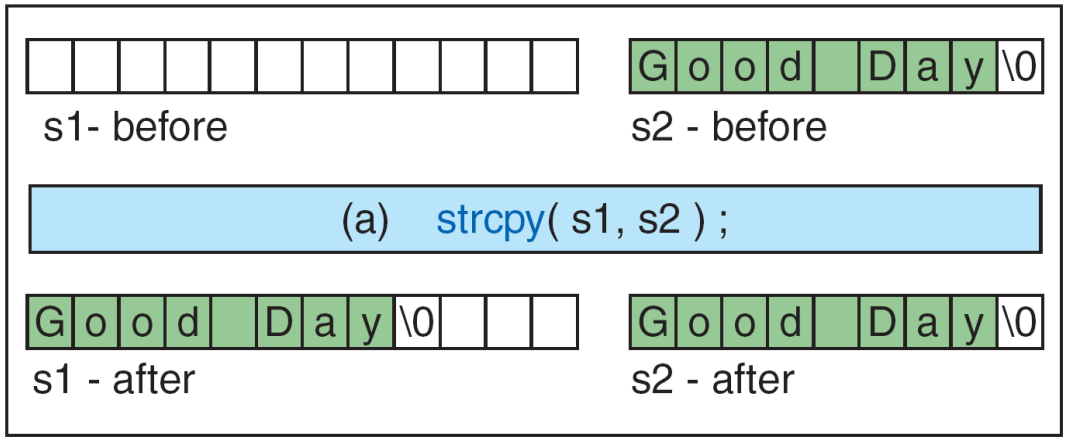
**String Length and String Copy**
**String Compare and String Concatenate**
**Character in String**
**Search for a Substring and Search for Character in Set**
**String Span and String Token**
**String to Number**

# PROGRAM 11-11 Add Left Margin

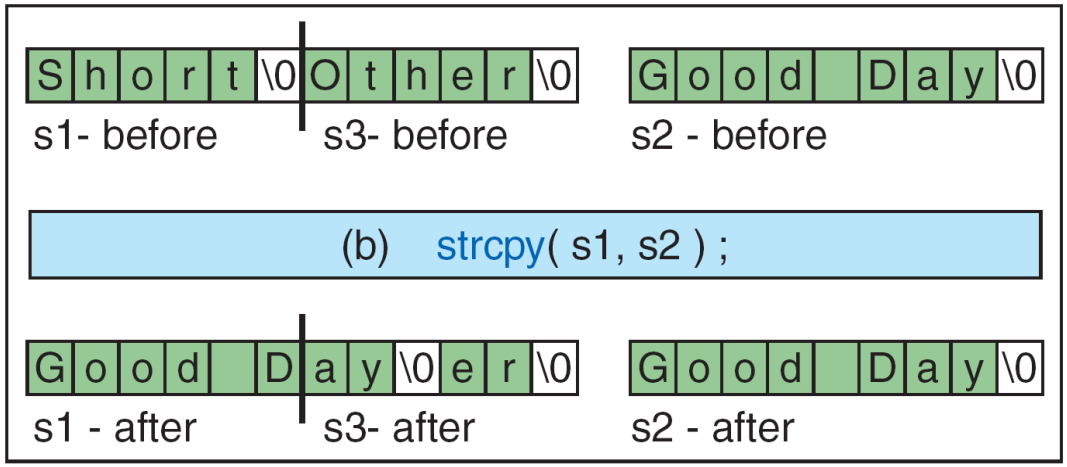
```
1  /* Typewriter program: adds two spaces to the left
2      margin and writes line to file
3        Written by:
4        Date:
5  */
6  #include <stdio.h>
7  #include <stdlib.h>
8  #include <string.h>
9
10 int main (void)
11 {
12 // Local Declarations
13    FILE* spOutFile;
14    char  strng[81];
15
16 // Statements
17    if (!(spOutFile = fopen("P11-11.TXT", "w")))
```

# PROGRAM 11-11 Add Left Margin

```
18          {
19           printf("\aCould not open output file.\n");
20           exit (100);
21          } // if
22
23      while (fgets(strng, sizeof(strng), stdin))
24          {
25           fputc(' ', spOutFile);
26           fputc(' ', spOutFile);
27           fputs(strng, spOutFile);
28           if (strng[strlen(strng) - 1] != '\n')
29               fputs("\n", spOutFile);
30          } // while
31      fclose (spOutFile);
32      return 0;
33  }  // main
```




Copying Strings




Copying Long Strings

**FIGURE 11-14 String Copy**

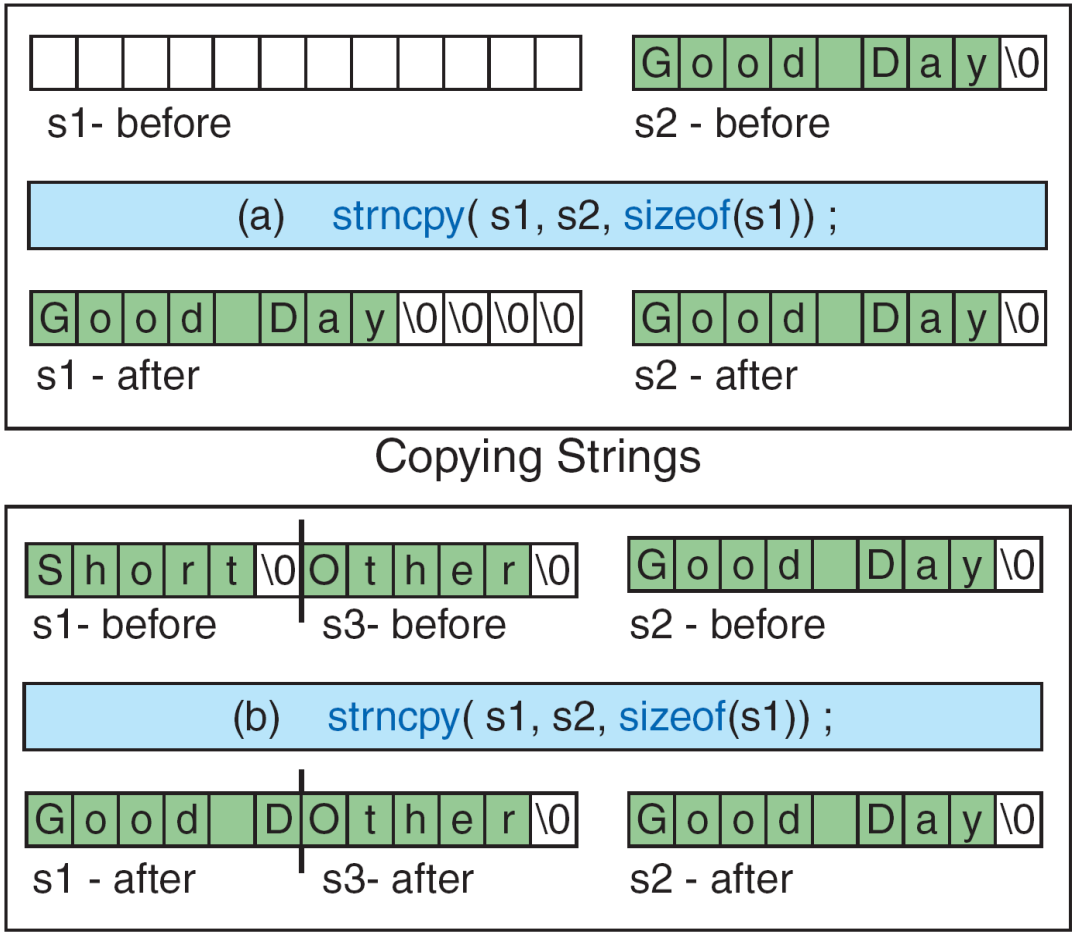



**FIGURE 11-15 String-number Copy**

## *Note*

**Always use strncpy to copy one string to another.**

## PROGRAM 11-12 Build Name Array in Heap

```
1  /* Build a dynamic array of names.
2        Written by:
3        Date:
4  */
5  #include <stdio.h>
6  #include <stdlib.h>
7  #include <string.h>
8
9  #define FLUSH while (getchar() != '\n')
10
11 int main (void)
12 {
13 // Local Declarations
14    char   input[81];
15    char** pNames;          // array of pointers to char
16
17    int size;
18    int namesIndex;
19
```

# PROGRAM 11-12 Build Name Array in Heap

```
20  // Statements
21     printf("How many names do you plan to input? ");
22     scanf ("%d", &size);
23     FLUSH;
24
25     // Allocate array in heap.
26     // One extra element added for loop control
27     pNames = calloc (size + 1, sizeof (char*));
28     printf("Enter names:\n");
29
30     namesIndex = 0;
31     while (namesIndex < size
32         && fgets(input, sizeof(input), stdin))
33         {
34          *(pNames + namesIndex)  = (char*)
35               calloc (strlen(input) + 1, sizeof(char));
36          strcpy (*(pNames + namesIndex), input);
37          namesIndex++;
38        } // while
```

# PROGRAM 11-12 Build Name Array in Heap

```
39
40      *(pNames + namesIndex) = NULL;
41      printf("\nYour names are: \n");
42      namesIndex = 0;
43      while (*(pNames + namesIndex))
44          {
45           printf("%3d: %s",
46                   namesIndex, *(pNames + namesIndex));
47           namesIndex++;
48          } // while
49      return 0;
50  }   // main
```

```
Results:
How many names do you plan to input? 3
Enter names:
Tom
Rico
Huang

Your names are:
  0: Tom
  1: Rico
  2: Huang
```

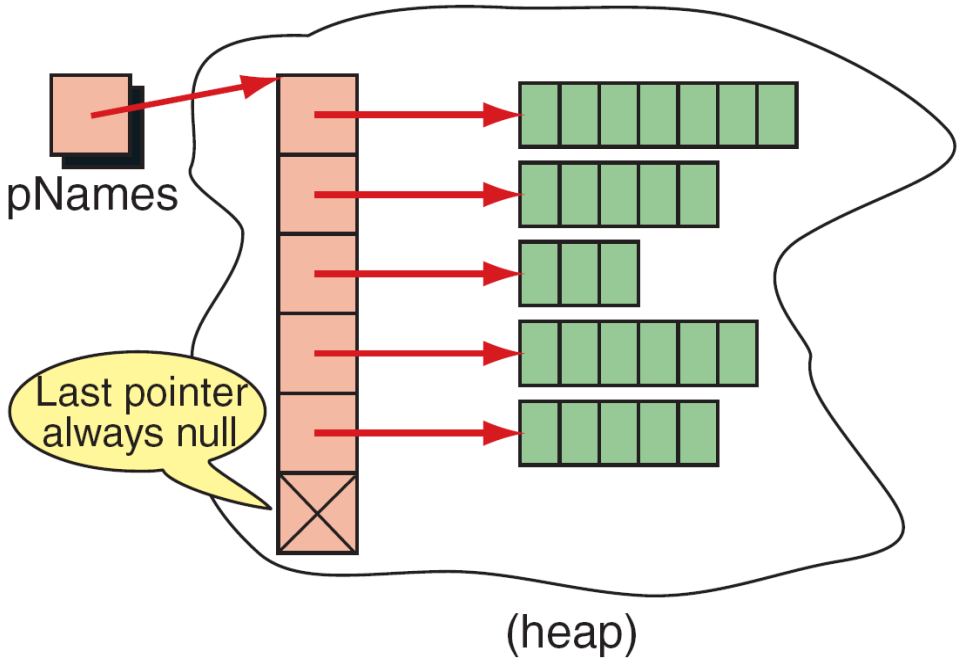



**FIGURE 11-16 Structure for Names Array**

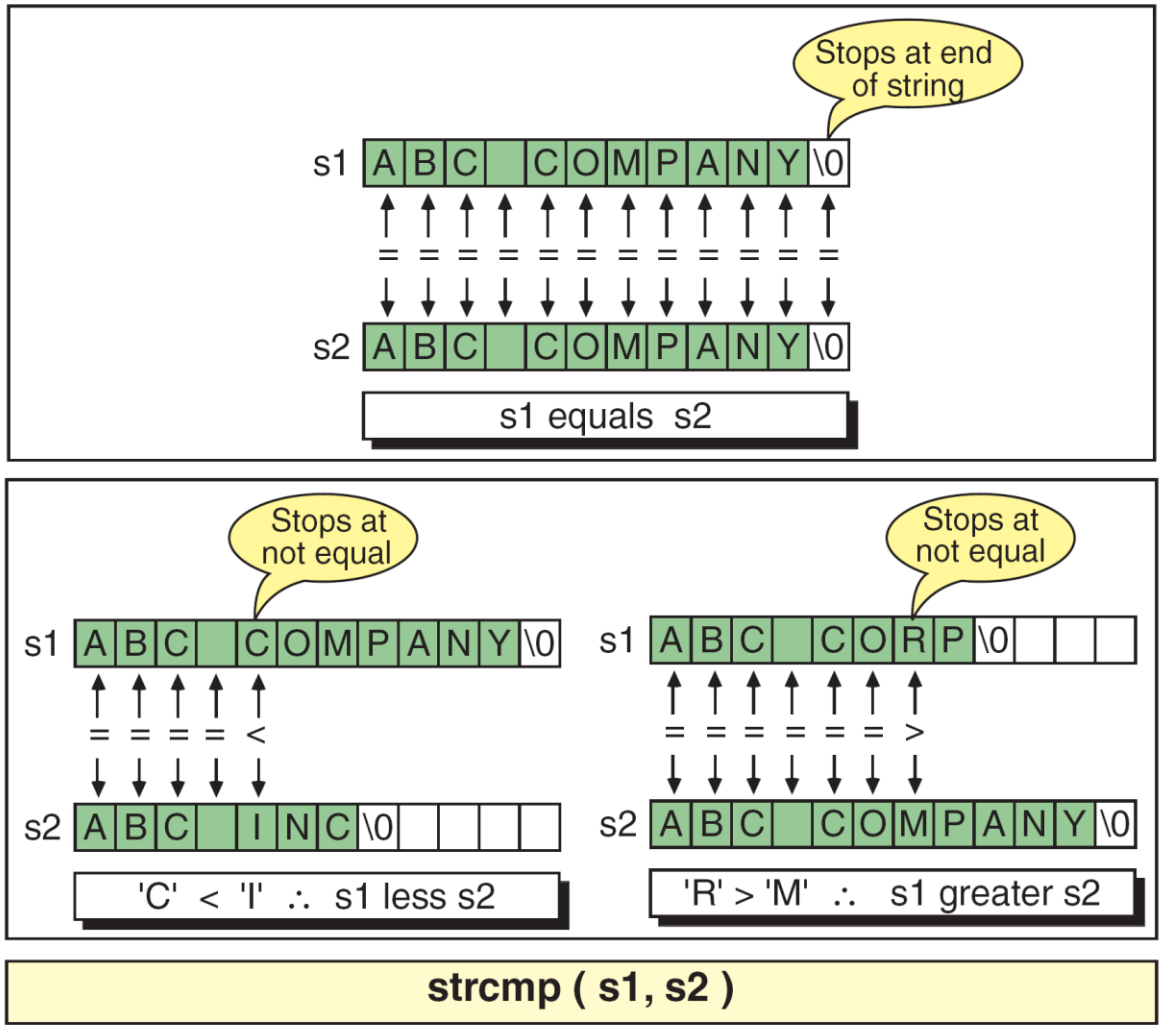



**FIGURE 11-17 String Compares**

| string1 | string2 | Size | Results | Returns |
|---|---|---|---|---|
| "ABC123" | "ABC123" | 8 | equal | 0 |
| "ABC123" | "ABC456" | 3 | equal | 0 |
| "ABC123" | "ABC456" | 4 | string1 < string2 | < 0 |
| "ABC123" | "ABC" | 3 | equal | 0 |
| "ABC123" | "ABC" | 4 | string1 > string2 | > 0 |
| "ABC" | "ABC123" | 3 | equal | 0 |
| "ABC123" | "123ABC" | –1 | equal | 0 |

**Table 11-1 Results for String Compare**

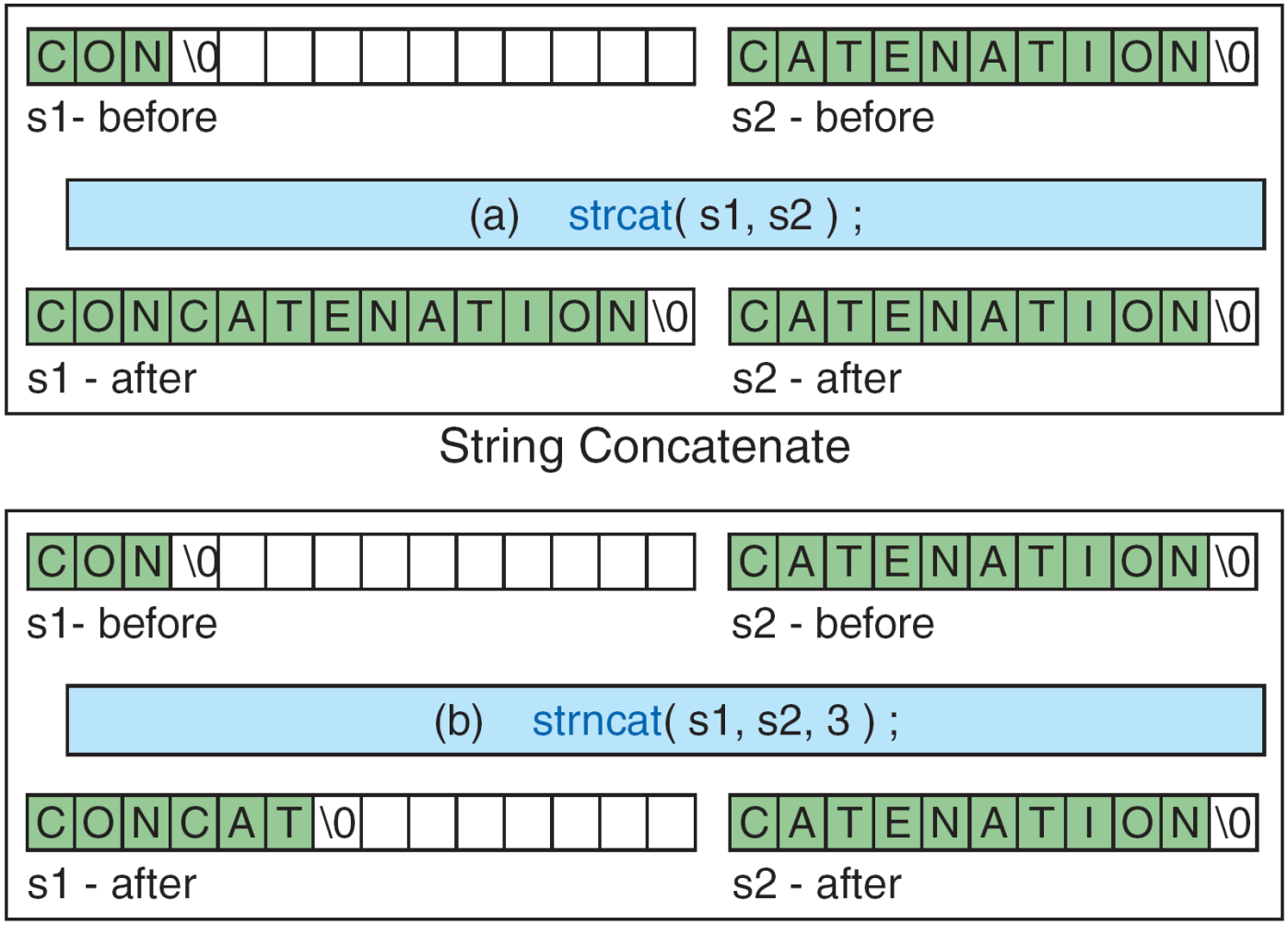



**FIGURE 11-18 String Concatenation**

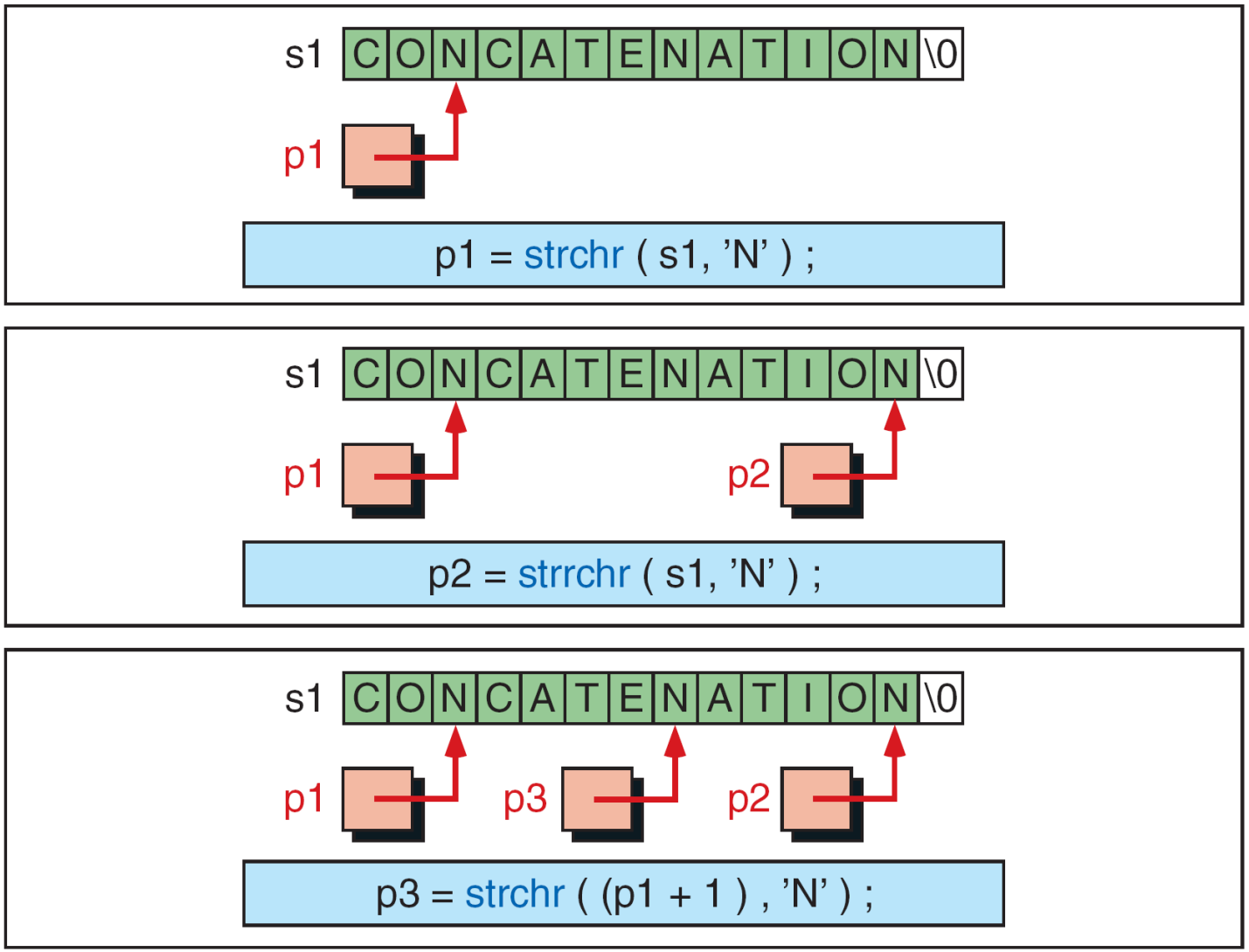



**FIGURE 11-19 Character in String (*strchr*)**

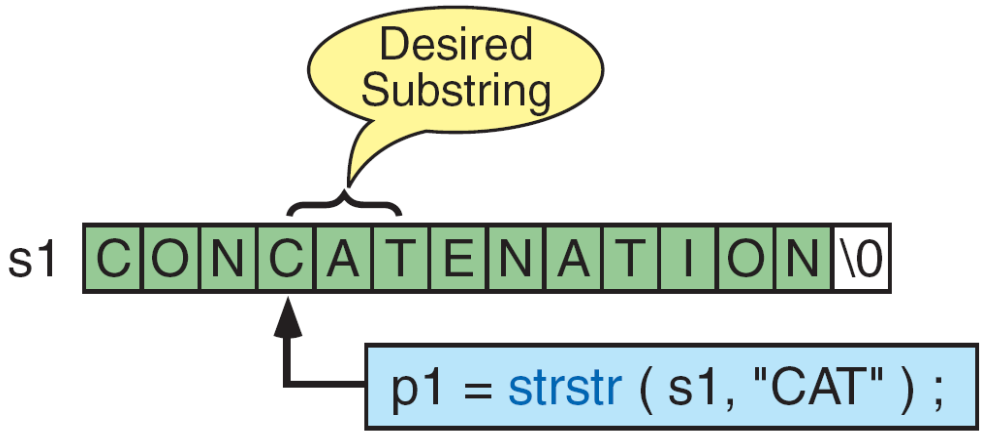



**FIGURE 11-20 String in String**

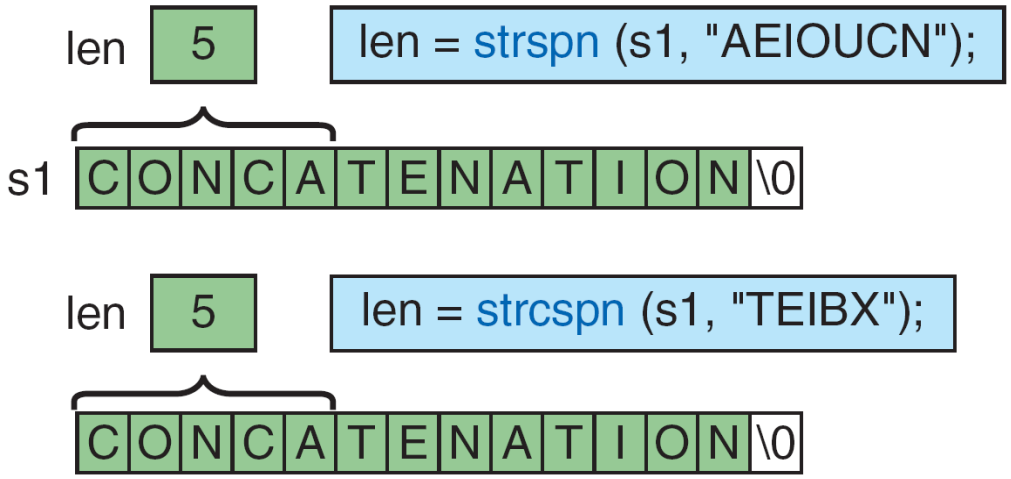



**FIGURE 11-21 String Span**

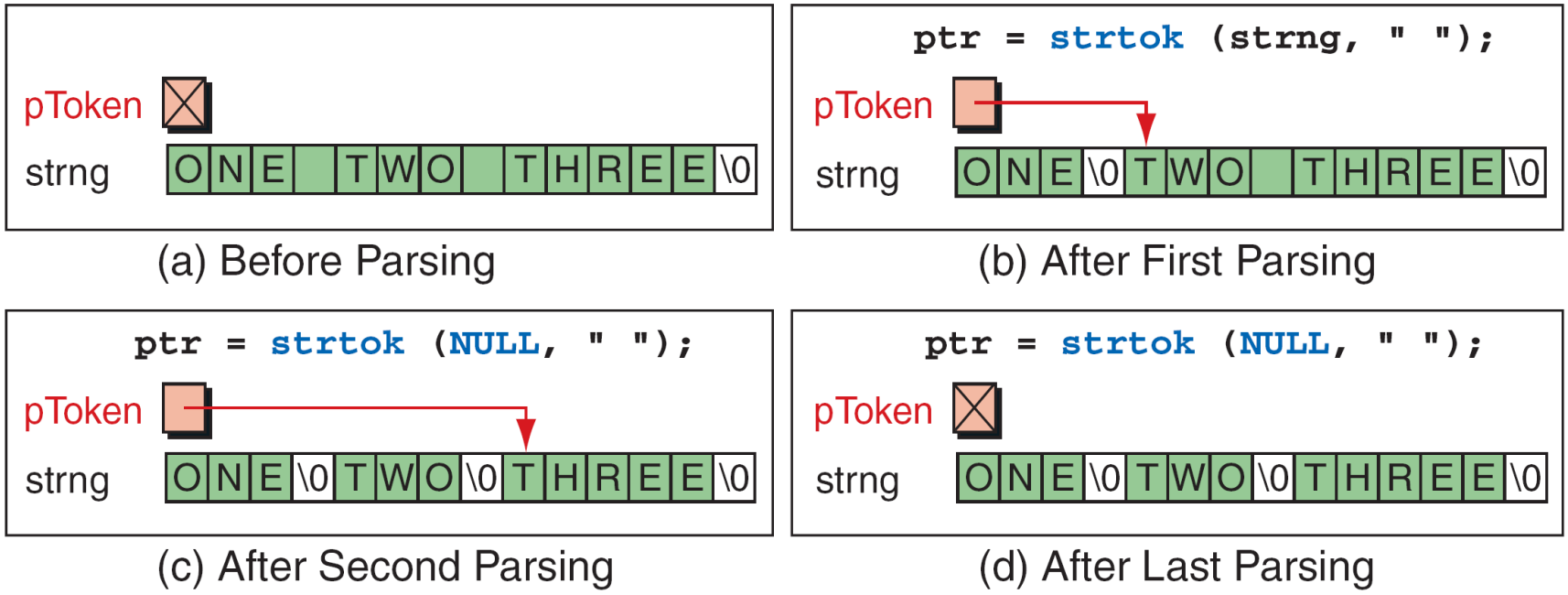



**FIGURE 11-22 Streams**

## PROGRAM 11-13 Demonstrate String to Long

```
1  /* Demonstrate string to long function.
2        Written by:
3        Date:
4  */
5  #include <stdio.h>
6  #include <stdlib.h>
7
8  int main (void)
9  {
10 // Local Declarations
11    long  num;
12    char* ptr;
13
14 // Statements
15    num = strtol ("12345 Decimal constant: ", &ptr, 0);
16    printf("%s %ld\n", ptr, num);
17
18    num = strtol ("11001 Binary constant : ", &ptr, 2);
19    printf("%s %ld\n", ptr, num);
```

## PROGRAM 11-13 Demonstrate String to Long

```
20
21      num = strtol ("13572 Octal constant  : ", &ptr, 8);
22      printf("%s %ld\n", ptr, num);
23
24      num = strtol (" 7AbC Hex constant    : ", &ptr, 16);
25      printf("%s %ld\n", ptr, num);
26
27      num = strtol ("11001 Base 0-Decimal  : ", &ptr, 0);
28      printf("%s %ld\n", ptr, num);
29
30      num = strtol ("01101 Base 0-Octal    : ", &ptr, 0);
31      printf("%s %ld\n", ptr, num);
32
33      num = strtol ("0x7AbC Base 0-Hex      : ", &ptr, 0);
34      printf("%s %ld\n", ptr, num);
35
36      num = strtol ("Invalid input    : ", &ptr, 0);
37      printf("%s %ld\n", ptr, num);
```

# PROGRAM 11-13 Demonstrate String to Long

```
38
39      return 0;
40  }   // main
```

```
Results:
    Decimal constant:  12345
    Binary constant :  25
    Octal constant  :  6010
    Hex constant    :  31420
    Base 0-Decimal  :  11001
    Base 0-Octal    :  577
    Base 0-Hex      :  31420
    Invalid input   :  0
```

| Numeric Format | ASCII Function | Wide-character Function |
|---|---|---|
| double | strtod | wcstod |
| float | strtof | wcstof |
| long double | strtold | wcstold |
| long int | strtol | wcstol |
| long long int | strtoll | wcstoll |
| unsigned long int | strtoul | wcstoul |
| unsigned long long int | strtoull | wcstoull |

**Table 11-2 String-to-Number Functions**

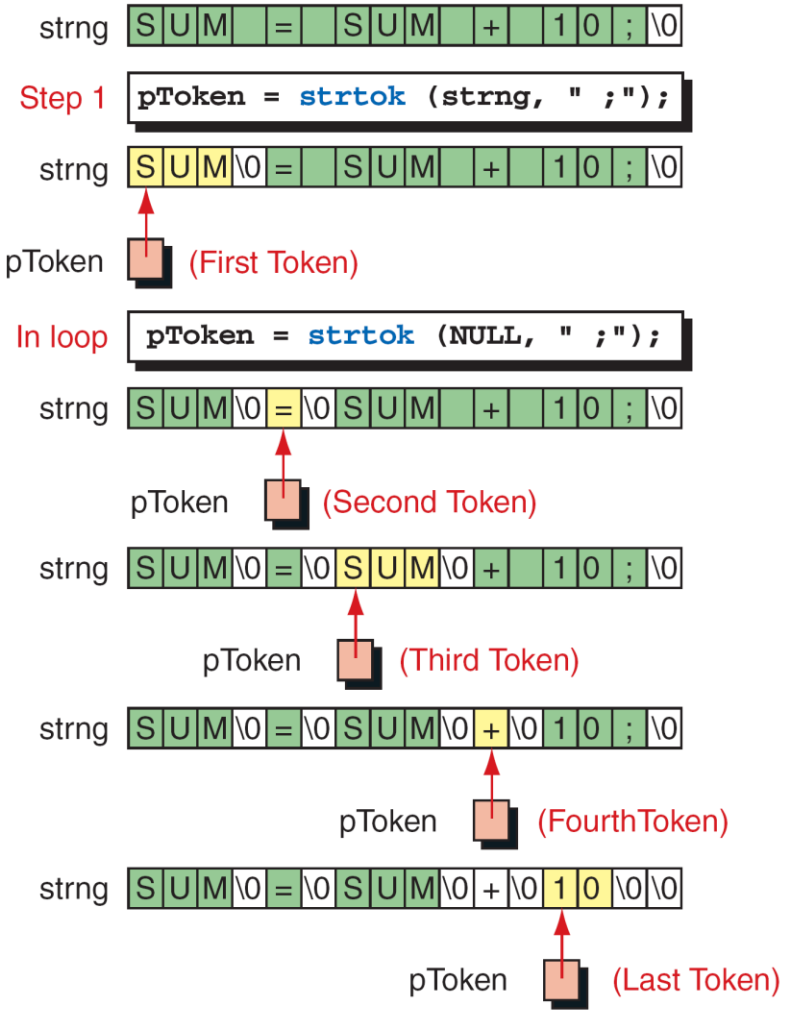



**FIGURE 11-23 Parsing with String Token**

# PROGRAM 11-14 Parsing a String with String Token

```
1  /* Parse a simple algebraic expression.
2        Written by:
3        Date:
4  */
5  #include <stdio.h>
6  #include <string.h>
7
8  int main (void)
9  {
10 // Local Declarations
11    char  strng [16] = "sum = sum + 10;";
12    char* pToken;
13    int   tokenCount;
14
15 // Statements
16    tokenCount = 0;
17    pToken = strtok (strng, " ;");
18
```

# PROGRAM 11-14 Parsing a String with String Token

```
19      while (pToken)
20          {
21           tokenCount++;
22           printf("Token %2d contains %s\n",
23                    tokenCount, pToken);
24           pToken = strtok (NULL, " ;");
25          } // while
26
27      printf("\nEnd of tokens\n");
28      return 0;
29  }   // main
```

```
Results:
Token   1 contains sum
Token   2 contains =
Token   3 contains sum
Token   4 contains +
Token   5 contains 10

End of tokens
```

# PROGRAM 11-15 Compare Packed String Function

```
1  /* This program packs and compares a string.
2        Written by:
3        Date:
4  */
5  #include <stdio.h>
6  #include <string.h>
7
8  #define ALPHA \
9   "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz"
10
11 // Function Declarations
12 int  strCmpPk (char* S1, char* S2);
13 void strPk    (char* s1, char* s2);
14
15 int main (void)
16 {
17 // Local Declarations
18    int  cmpResult;
19    char s1[80];
20    char s2[80];
```

# PROGRAM 11-15 Compare Packed String Function

```
21
22  // Statements
23      printf("Please enter first string:\n");
24      fgets (s1, 80, stdin);
25      s1[strlen(s1) - 1] = '\0';
26
27      printf("Please enter second string:\n");
28      fgets (s2, 80, stdin);
29      s2[strlen(s2) - 1] = '\0';
30
31      cmpResult = strCmpPk (s1, s2);
32      if (cmpResult < 0)
33          printf("string1 < string2\n");
34      else if (cmpResult > 0)
35          printf("string1 > string2\n");
36      else
37          printf("string1 == string2\n");
38
39      return 0;
40  }  // main
```

# PROGRAM 11-15 Compare Packed String Function

```
41
42  /* ==================== strCmpPk ====================
43     Packs two strings and then compares them.
44        Pre   s1 and s2 contain strings
45        Post  returns result of strcmp of packed strings
46  */
47  int strCmpPk (char* s1, char* s2)
48  {
49  // Local Declarations
50     char s1In [80];
51     char s1Out[81];
52     char s2In [80];
53     char s2Out[81];
54
55  // Statements
56     strncpy (s1In, s1, sizeof(s1In) - 1);
57     strncpy (s2In, s2, sizeof(s2In) - 1);
58     strPk (s1In, s1Out);
59     strPk (s2In, s2Out);
60     return (strcmp (s1Out, s2Out));
61  }  // strCmpPk
62
```

# PROGRAM 11-15 Compare Packed String Function

```
63  /* ==================== strPk ====================
64     Deletes all non-alpha characters from s1 and
65     copies to s2.
66        Pre   s1 is a string
67        Post  packed string in s2
68              s1 destroyed
69  */
70  void strPk (char* s1, char* s2)
71  {
72  // Local Declarations
73     int strSize;
74
75  // Statements
76     *s2 = '\0';
77     while (*s1 != '\0')
78        {
79         // Find non-alpha character & replace
80         strSize = strspn(s1, ALPHA);
81         s1[strSize] = '\0';
82         strncat (s2, s1, 79 - strlen(s2));
83         s1 += strSize + 1;
```

# PROGRAM 11-15 Compare Packed String Function

```
84          } // while
85       return;
86    }  // strPk
```

```
Results:
Please enter first string:
a b!c 234d
Please enter second string:
abcd
string1 == string2

Please enter first string:
abcd
Please enter second string:
aabb
string1 > string2
```