# COP 3223: C Programming
# Spring 2009

## Strings In C – Part 4

Instructor : Dr. Mark Llewellyn
markl@cs.ucf.edu
HEC 236, 407-823-2790
http://www.cs.ucf.edu/courses/cop3223/spr2009/section1

School of Electrical Engineering and Computer Science
University of Central Florida

# The Character Handling Library

- Although we are primarily dealing with strings, processing of character data within the strings is an important and often utilized function of many application programs.
- The character handling functions are found in the `<ctype.h>` standard library.
- Some of the more common character handling functions are shown in the tables on the next two pages.
- Following the tables are several example programs that illustrate some of the character handling functions in `<ctype.h>`.

# Some Of The Functions In `<ctype.h>`

| Function Prototype | Function Description |
|---|---|
| `int isdigit (int c);` | Returns a true (non-zero) value if `c` is a digit and 0 (false) otherwise. |
| `int isalpha (int c);` | Returns a true value if `c` is a letter and 0 (false) otherwise. |
| `int isalnum (int c);` | Returns a true value if `c` is a digit or a letter and 0 (false) otherwise. |
| `int islower (int c);` | Returns a true value if `c` is a lowercase letter and 0 (false) otherwise. |
| `int isupper (int c);` | Returns a true value if `c` is an uppercase letter and 0 (false) otherwise. |
| `int tolower (int c);` | If `c` is an uppercase letter, this function returns `c` as a lowercase letter. Otherwise, the function returns the argument unchanged. |
| `int toupper (int c);` | If c is a lowercase letter, this function returns c as an uppercase letter. Otherwise, the function returns the argument unchanged. |

# Some Of The Functions In `<ctype.h>`

| Function Prototype | Function Description |
|---|---|
| `int isspace (int c);` | Returns a true (non-zero) value if `c` is a white-space character. This includes: newline ('\n'), space (' '), form feed ('\F'), carriage return ('\r'), horizontal tab ('\t'), or vertical tab ('\v'). Otherwise a value of 0 (false) is returned. |
| `int iscntrl (int c);` | Returns a true value if `c` is a control character and 0 (false) otherwise. |
| `int ispunct (int c);` | Returns a true value if `c` is a printing character other than a space, a digit, or a letter and 0 (false) otherwise. (This function is basically returning true for punctuation marks.) |
| `int isprint (int c);` | Returns a true value if `c` is a printing character including a space (' ') and 0 (false) otherwise. |
| `int isgraph (int c);` | Returns a true value if `c` is a printing character other than a space (' ') and 0 (false) otherwise. |

[*] character handling example 1.c

```c
//Strings In C - Part 4 - Character Handling Example 1
//Using functions isdigit, isalpha, isalnum
//April 2, 2009   Written by: Mark Llewellyn
#include <stdio.h>
#include <ctype.h>

int main()
{
   printf( "\n%s\n%s%s\n%s%s\n\n", "According to isdigit: ",
       isdigit( '8' ) ? "8 is a " : "8 is not a ", "digit",
       isdigit( '#' ) ? "# is a " : "# is not a ", "digit" );

   printf( "%s\n%s%s\n%s%s\n%s%s\n%s%s\n\n",
       "According to isalpha:",
       isalpha( 'A' ) ? "A is a " : "A is not a ", "letter",
       isalpha( 'b' ) ? "b is a " : "b is not a ", "letter",
       isalpha( '&' ) ? "& is a " : "& is not a ", "letter",
       isalpha( '4' ) ? "4 is a " : "4 is not a ", "letter" );

   printf( "%s\n%s%s\n%s%s\n%s%s\n\n\n",
       "According to isalnum:",
       isalnum( 'A' ) ? "A is a " : "A is not a ",
       "digit or a letter",
       isalnum( '8' ) ? "8 is a " : "8 is not a ",
       "digit or a letter",
       isalnum( '#' ) ? "# is a " : "# is not a ",
       "digit or a letter" );

   system("PAUSE");
   return 0; // indicates successful termination
}//end main function
```

C:\Courses\COP 3223 - C Programming\Spring ...

```
According to isdigit:
8 is a digit
# is not a digit

According to isalpha:
A is a letter
b is a letter
& is not a letter
4 is not a letter

According to isalnum:
A is a digit or a letter
8 is a digit or a letter
# is not a digit or a letter


Press any key to continue . . .
```

# An aside on the conditional operator ?

The previous example program utilizes the conditional operator ?. This C operator is a sometimes useful shorthand replacement for an if-else control statement.

The conditional operator is the only ternary operator (ternary means 3-way) in C. The conditional operator requires 3 operands. The operands together with the conditional operator form a conditional expression.

The first operand is a condition, the second operand is the value for the entire conditional expression if the condition is true and the third operand is the value for the entire conditional expression if the condition is false.

The general form is: `condition ? operand : operand;`

An example is: `grade >= 60 ? printf("passed\n"): printf('failed\n");`

character handling example 1.c | [*] character handling example 2.c

```c
int main()
{
   printf( "\n%s\n%s%s\n%s%s\n%s%s\n%s%s\n\n",
           "According to islower:",
           islower( 'p' ) ? "p is a " : "p is not a ",
           "lowercase letter",
           islower( 'P' ) ? "P is a " : "P is not a ",
           "lowercase letter",
           islower( '5' ) ? "5 is a " : "5 is not a ",
           "lowercase letter",
           islower( '!' ) ? "! is a " : "! is not a ",
           "lowercase letter" );

   printf( "%s\n%s%s\n%s%s\n%s%s\n%s%s\n\n",
           "According to isupper:",
           isupper( 'D' ) ? "D is an " : "D is not an ",
           "uppercase letter",
           isupper( 'd' ) ? "d is an " : "d is not an ",
           "uppercase letter",
           isupper( '8' ) ? "8 is an " : "8 is not an ",
           "uppercase letter",
           isupper( '$' ) ? "$ is an " : "$ is not an ",
           "uppercase letter" );

   printf( "%s%c\n%s%c\n%s%c\n%s%c\n\n\n",
           "u converted to uppercase is ", toupper( 'u' ),
           "7 converted to uppercase is ", toupper( '7' ),
           "$ converted to uppercase is ", toupper( '$' ),
           "L converted to lowercase is ", tolower( 'L' ) );
   system("PAUSE");
   return 0;
```



C:\Courses\COP 3223 - C Programming\S...

```
According to islower:
p is a lowercase letter
P is not a lowercase letter
5 is not a lowercase letter
! is not a lowercase letter

According to isupper:
D is an uppercase letter
d is not an uppercase letter
8 is not an uppercase letter
$ is not an uppercase letter

u converted to uppercase is U
7 converted to uppercase is 7
$ converted to uppercase is $
L converted to lowercase is l


Press any key to continue . . .
```

```
7  int main()
8  {
9     printf( "\n%s\n%s%s%s\n%s%s%s\n%s%s\n\n",
10        "According to isspace:", "Newline", isspace( '\n' ) ? " is a " :
11        " is not a ", "whitespace character", "Horizontal tab",
12        isspace( '\t' ) ? " is a " : " is not a ", "whitespace character",
13        isspace( '%' ) ? "% is a " : "% is not a ","whitespace character" );
14
15    printf( "%s\n%s%s%s\n%s%s\n\n", "According to iscntrl:",
16        "Newline", iscntrl( '\n' ) ? " is a " : " is not a ",
17        "control character", iscntrl( '$' ) ? "$ is a " :
18        "$ is not a ", "control character" );
19
20    printf( "%s\n%s%s\n%s%s\n%s%s\n\n",
21        "According to ispunct:", ispunct( ';' ) ? "; is a " : "; is not a ",
22        "punctuation character", ispunct( 'Y' ) ? "Y is a " : "Y is not a ",
23        "punctuation character", ispunct( '#' ) ? "# is a " : "# is not a ",
24        "punctuation character" );
25
26    printf( "%s\n%s%s\n%s%s%s\n\n", "According to isprint:",
27        isprint( '$' ) ? "$ is a " : "$ is not a ", "printing character",
28        "Alert", isprint( '\a' ) ? " is a " : " is not a ",
29        "printing character" );
30
31    printf( "%s\n%s%s\n%s%s%s\n\n\n",  "According to isgraph:",
32        isgraph( 'Q' ) ? "Q is a " : "Q is not a ",
33        "printing character other than a space",
34        "Space", isgraph( ' ' ) ? " is a " : " is not a ",
35        "printing character other than a space" );
36
37    system("PAUSE");
```

C:\Courses\COP 3223 - C Programming\Spring 2009\COP 3223 Progr...

```
According to isspace:
Newline is a whitespace character
Horizontal tab is a whitespace character
% is not a whitespace character

According to iscntrl:
Newline is a control character
$ is not a control character

According to ispunct:
; is a punctuation character
Y is not a punctuation character
# is a punctuation character

According to isprint:
$ is a printing character
Alert is not a printing character

According to isgraph:
Q is a printing character other than a space
Space is not a printing character other than a space
```

# Using String Conversion Functions

- Another sometimes useful operation that needs to be performed on strings is their conversion into numeric formats.
- For example, converting the string "1234" into the integer value 1234.
- Again, the C standard library has several functions that perform this task depending on the numeric type of the conversion.
- These functions are included in the `<stdlib.h>` standard library.
- The table on the next page lists some of the more common string conversion functions in this library.
- Care should be used when dealing with such functions, particularly in converting to integer types to ensure that overflow does not occur. If it is possible to do so, the application should normally read numeric types as numeric input rather than convert using these functions.

# Some Of The String Conversion Functions In `<stdlib.h>`

| Function Prototype | Function Description |
|---|---|
| `double atof (const char *nPtr);` | Converts the string pointed to by `nPtr` to a `double`. |
| `int atoi (const char *nPtr);` | Converts the string pointed to by `nPtr` to an `int`. |
| `long atol (const char *nPtr);` | Converts the string pointed to by `nPtr` to a `long int`. |

- Example program using each of these three functions are shown on the next page.
- Notice in each of the programs that after the string is converted to a numeric type, the converted numeric value is used in an expression to illustrate the conversion.

atof example.c

```c
//Strings In C - Part 4 - Using atof
//April 2, 2009    Written by: Mark Llew

#include <stdio.h>
#include <stdlib.h>

int main()
{
   double d; // variable to hold converted string

   d = atof( "99.0" );

   printf( "\n%s%.3f\n%s%.3f\n\n\n",
           "The string \"99.0\" converted to double is ", d,
           "The converted value divided by 2 is ",
           d / 2.0 );
   system("PAUSE");
   return 0;
}//end main function

```

C:\Courses\COP 3223 - C Programming\Spring 2009\COP 322...

```
The string "99.0" converted to double is 99.000
The converted value divided by 2 is 49.500


Press any key to continue . . .
```

atof example.c | atoi example.c

```c
//Strings In C - Part 4 - atoi
//April 2, 2009  Written by: Ma

#include <stdio.h>
#include <stdlib.h>

int main()
{
   int i; // variable to hold converted string

   i = atoi( "2593" );

   printf( "\n%s%d\n%s%d\n\n\n",
           "The string \"2593\" converted to int is ", i,
           "The converted value minus 593 is ", i - 593 );
   //this one will cause an overflow
   i = atoi( "123456789123" );

   printf( "\n%s%d\n%s%d\n\n\n",
           "The string \"123456789123\" converted to int is ", i,
           "The converted value minus 500 is ", i - 500 );
   system("PAUSE");
   return 0;
}//end main function
```

C:\Courses\COP 3223 - C Programming\Spring 2009\COP 3223 Progra...

```
The string "2593" converted to int is 2593
The converted value minus 593 is 2000


The string "123456789123" converted to int is -1097262461
The converted value minus 500 is -1097262961
```

atof example.c | atoi example.c | atol example.c

```c
//Strings In C - Part 4 - atol example
//April 2, 2009  Written by: Mark Llewellyn

#include <stdio.h>
#include <stdlib.h>

int main()
{
   long l; /* variable to hold converted string */

   l = atol( "1000000" );

   printf( "\n%s%ld\n%s%ld\n\n\n",
           "The string \"1000000\" converted to long int is ", l,
           "The converted value divided by 2 is ", l / 2 );

   system("PAUSE");
   return 0;
}//end main function
```

C:\Courses\COP 3223 - C Programming\Spring 2009\COP 3223 Program Files\Strings...

```
 "1000000" converted to long int is 1000000
ted value divided by 2 is 500000
```

# Arrays Of Strings

- As a final example dealing with strings, we'll look at a program that uses a two-dimensional array holding strings. In other words, an array of strings.
- In the version shown here, we'll treat the 2-dimensional array in much the same was that we have treated other arrays, using implicit pointers being passed to functions needing access to the array.
- We'll look in some more detail at pointer arithmetic later and we'll revisit this problem and use explicit pointer references.
- The following program reads lines of text, the number of which is unknown in advance, from a file and stores the text in a 2-dimensional array of strings.

lyrics - Notepad

File Edit Format View Help

# The data file

```
It's never been easy for me
To find words to go along with a melody
But this time there's actually something, on my mind
So please forgive these few brief awkward lines

Since I met you, my whole life has changed
It's not just my furniture you?ve rearranged
I was living in the past, but somehow you've brought me back
And I haven't felt like this since before Frankie said relax

And while I know, based on my track record,
I might not seem like the safest bet
All I'm asking you is don't write me off, just yet

For years I've been telling myself the same old story
That I'm happy to live off my so-called former glories
But you've given me a reason to take another chance
Now I need you, despite the fact that you've killed all my plants

And though I know, I've already blown more chances
Than anyone should ever get
All I'm asking you is don?t write me off, just yet
Don't write me off just yet
```

arrays of strings - music lyrics.c

```c
2  //This program reads in several lines of text from a file and stores the text
3  //in a two-dimensional array.
4  //NOTE:  This version of the program treats the lyrics as a 2-dimensional array
5  //       of strings.
6  //March 20, 2009     Written by: Mark Llewellyn
7  #include <stdio.h>
8  #include <string.h>
9  #define MAX_LENGTH 81
10 #define MAX_LINES 40
11
12 int readLyrics(char music[MAX_LINES][MAX_LENGTH])
13 {
14     FILE *inFilePtr;  //declare input file pointer
15     char line[MAX_LENGTH]; //a line of the lyrics
16     int index = 0; //a line counter
17     char *discard; //return pointer not used
18
19     if ( (inFilePtr = fopen("lyrics.dat","r") ) == NULL ) {
20        printf("Sorry, the file could not be opened\n");
21     }
22     else {
23         fgets(line, MAX_LENGTH, inFilePtr);
24         while( !feof(inFilePtr) ){
25             discard = strcpy(music[index], line);
26             index++;
27             fgets(line, MAX_LENGTH, inFilePtr);
28         }//end while stmt
29         return index;
30     }//end else
31 }//end function readLyrics
```

[*] arrays of strings - music lyrics.c

```c
32
33
34
35
36
37 void printLyrics(char music[MAX_LINES][MAX_LENGTH], int counter)
38 {
39      char line[MAX_LENGTH]; //a line of lyrics
40      int index; //loop control
41
42      for (index = 0; index < counter; index++){
43          puts(music[index]);
44      }//end for stmt
45      return;
46 }//end printLyrics function
47
48 int main()
49 {
50     int numberOfRows;  //the number of rows in the lyrics.
51     char music[MAX_LINES][MAX_LENGTH];  //the lyrics
52
53     numberOfRows = readLyrics(music);
54     printLyrics(music, numberOfRows);
55
56     printf("\n\n");
57     system("PAUSE");
58     return 0;
59 }//end main function
```

K:\COP 3223 - Spring 2009\COP 3223 Program Files\Strings In C - Part 4\arrays of strings -...

```
It's never been easy for me
To find words to go along with a melody
But this time there's actually something, on my mind
So please forgive these few brief awkward lines

Since I met you, my whole life has changed
It's not just my furniture you?ve rearranged
I was living in the past, but somehow you've brought me back
And I haven't felt like this since before Frankie said relax

And while I know, based on my track record,
I might not seem like the safest bet
All I'm asking you is don't write me off, just yet

For years I've been telling myself the same old story
That I'm happy to live off my so-called former glories
But you've given me a reason to take another chance
Now I need you, despite the fact that you've killed all my plants

And though I know, I've already blown more chances
Than anyone should ever get
All I'm asking you is don?t write me off, just yet
Don't write me off just yet

Press any key to continue . . .
```

# The output

# Practice Problems

1. Write a program that will read in three lines of text from the keyboard (put the lines into a 2-dimensional array). Once the lines are in the array, process the strings so that for each character in the alphabet you record the number of times that character appeared in total in the three lines of text. The output of the program should show the total number of times each letter appeared for all 26 letters in the alphabet.