PROGRAMACION I

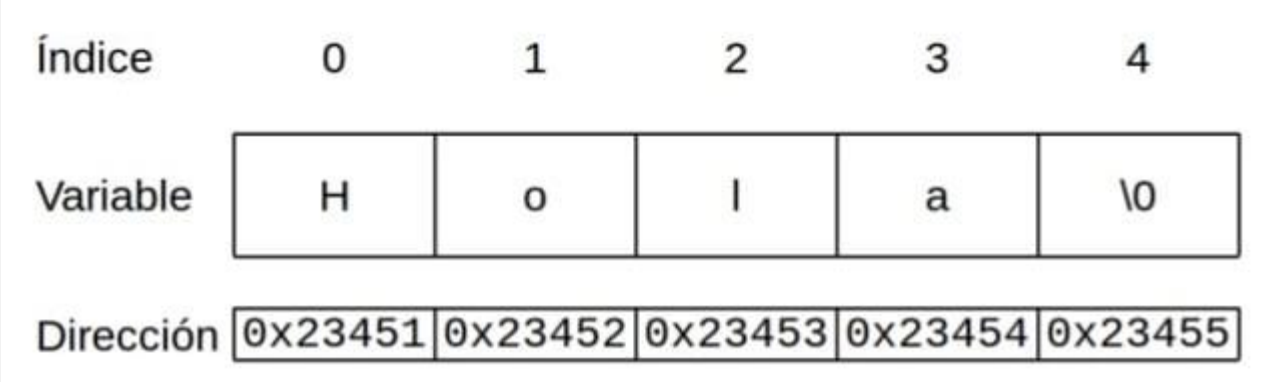
ING. OTTO PARRA GONZÁLEZ, PhD

FUNCIONES PARA EL MANEJO DE CADENAS DE CARACTERES EN C

PROGRAMACION I

INTRODUCCION

- C no dispone de un tipo de dato String para trabajar con cadenas de texto como en otros lenguajes. En su lugar, una cadena está representada por un array de caracteres (char) habitualmente terminado con el caracter especial de fin de cadena \0-
- Este caracter especial no se visualiza por pantalla, podríamos decir que está pero "no es visible" a través de la pantalla.



INTRODUCCIÓN

- Las funciones para pedir datos al usuario scanf y gets introducen automáticamente un fin de cadena (\0) al final del texto capturado. En estos casos el array de caracteres tiene en C un caracter adicional además del texto visible, el correspondiente al final de cadena.
- Por ejemplo la longitud en caracteres de la palabra “abc” introducida por el usuario con scanf o gets no es tres, sino cuatro, ya que a los tres caracteres se suma el caracter especial de final de cadena representado con \0.

LONGITUD DE UNA CADENA

- Algunas formas de declarar una cadena de caracteres:
  - char pruebaCadena [3]; pruebaCadena [0]= 'a'; pruebaCadena [1]= 'b'; pruebaCadena [2]= 'c';
  - char pruebaCadena [] = "abc";
- En el primer caso el array es de tres caracteres, el de índice cero es a, el de índice 1 es b, el de índice 2 es c. En el segundo caso el array es de cuatro caracteres al introducir C el carácter de fin de cadena \0.
- Se llama tamaño (length) de un array al número de elementos de que consta. En el caso del array de caracteres “abc” con el carácter de fin de cadena se dice que consta de 4 elementos (contando el carácter especial de fin de cadena).
- Se puede conocer la longitud de un array de caracteres usando la función sizeof, siendo la sintaxis a emplear: **sizeof (nombreDelArray).**

MOSTRAR UNA CADENA EN PANTALLA

- A diferencia de los arrays de tipos de datos numéricos (arrays de enteros, de números con punto decimal, etc.), en donde cada elemento del array se debe considerar como una variable independiente de los demás, los arrays de caracteres (cadenas) se pueden manipular de dos maneras: de forma conjunta o separada.
- Por ejemplo, para mostrar en pantalla un array de caracteres se puede hacer dentro de un bucle, desde el primer caracter (indice 0) hasta el último carácter (lo que nos devuelve la función strlen):

```
for(i=0; i<strlen(cadena); i++)
  printf("%c",cadena[i]);
```

- Existe una mejor manera de mostrar en pantalla una cadena, y es utilizando el carácter de conversión %s:

```
printf("%s",cadena);
```

LONGITUD DE UNA CADENA

```
#include <stdio.h>
#include <stdlib.h>
int main() {
    int i = 0; int j=0; char pruebaCadena [] = "abc";
    while (pruebaCadena[i] != '\0') { printf ("%c", pruebaCadena[i]); i=i+1; }
    printf ("%cUsando \\0: ", '\n');
    while (pruebaCadena[j] != '\0') { printf ("Iteracion %d - ", j+1); j=j+1; }
    printf ("%cUsando sizeof: ", '\n');
    j= 0; //Reinicializar j
    while (j < sizeof(pruebaCadena)) { printf ("Iteracion %d - ", j+1); j=j+1; }
    printf ("\nsizeof pruebaCadena es %d", sizeof(pruebaCadena));
    return 0; // Ejemplo aprenderaprogramar.com
}
```

STRING.H

- string.h es un archivo de la Biblioteca estándar del lenguaje de programación C que contiene la definición de macros, constantes, funciones y tipos y algunas operaciones de manipulación de memoria.
- Las funciones declaradas en string.h se han hecho muy populares, por lo que están garantizadas para cualquier plataforma que soporte C.
- Las funciones para cadenas de caracteres sólo trabajan con conjuntos de caracteres ASCII o exten.siones ASCII compatibles

STRING.H

- A través de la librería string.h se puede usar distintas funciones relacionadas con cadenas. Para usar estas funciones se debe escribir **#include <string.h>** en la cabecera del programa. A continuación algunas de las funciones disponibles:

Función	Significado y ejemplo aprenderaprogramar.com
strcpy (arg1, arg2)	Copia arg2 en arg1. Ejemplo: strcpy (cadena, "control");
strlen (arg1)	Devuelve la longitud del texto representado por arg1. Ejemplo: strlen(cadena1)
strcat (arg1, arg2)	Concatena las cadenas representadas por arg1 y arg2. Ejemplo: strcat(cadena1, " unidades")
strcmp (arg1, arg2)	Devuelve 0 si las cadenas representadas por arg1 y arg2 son iguales, o un valor menor que cero si arg1 precede alfabéticamente a arg2. Ejemplo: resComparacion = strcmp (cadena4, cadena2);

EJEMPLO DE LAS FUNCIONES PARA MANEJO DE CADENAS

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
int main() {
   char cadena1 [] ="aprenderaprogramar.com"; char cadena2 [sizeof(cadena1)];
   char cadena3[] = " ok!";    char cadena4[50];
   int resComparacion;
   printf ("Cadena1 vale %s y Cadena2 vale %s \n", cadena1, cadena2);
   strcpy (cadena2, cadena1); //Copia cadena1 en cadena2
   printf ("Cadena1 vale %s y Cadena2 vale %s \n", cadena1, cadena2);
   printf ("Cadena1 tiene longitud %d (uno mas contando \\0) \n", strlen(cadena1));
   strcpy (cadena4, cadena1) ;
   printf ("Concatenacion %s \n", strcat(cadena4, cadena3));
   resComparacion = strcmp (cadena4, cadena2);
   printf ("Cadena1 vale %s y Cadena2 vale %s \n", cadena1, cadena2);
   printf ("Cadena3 vale %s y Cadena4 vale %s \n", cadena3, cadena4);
   if (resComparacion ==0) {
      puts ("Las cadenas 1 y 2 son iguales");
   }
   resComparacion = strcmp (cadena1, cadena3);
   if (resComparacion >0) {
      puts ("La cadena 1 precede alfabeticamente a la cadena 3");
   }
   return 0; // Ejemplo aprenderaprogramar.com
}
```

FUNCIONES PARA MANEJO DE CADENAS EN C

- Las cadenas en C no se pueden copiar escribiendo
  - cadena1 = cadena2.
- En su lugar, debe usarse la función strcpy.
- De igual manera, no se puede comparar dos cadenas usando ==, en vez de ello, se debe usar strcmp.

MAS FUNCIONES DE STRING.H

Nombres	Descripción
`memcpy`	copia n bytes entre dos áreas de memoria que no deben solaparse
`memmove`	copia n bytes entre dos áreas de memoria; al contrario que `memcpy` las áreas pueden solaparse
`memchr`	busca un valor a partir de una dirección de memoria dada y devuelve un puntero a la primera ocurrencia del valor buscado o NULL si no se encuentra
`memcmp`	compara los n primeros caracteres de dos áreas de memoria
`memset`	sobre escribe un área de memoria con un patrón de bytes dado
`strcat`	añade una cadena al final de otra
`strncat`	añade los n primeros caracteres de una cadena al final de otra
`strchr`	localiza un carácter en una cadena, buscando desde el principio
`strrchr`	localiza un carácter en una cadena, buscando desde el final
`strcmp`	compara dos cadenas alfabéticamente ('a'!='A')
`strncmp`	compara los n primeros caracteres de dos cadenas numéricamente ('a'!='A')
`strcoll`	compara dos cadenas según la colación actual ('a'=='A')
`strcpy`	copia una cadena en otra
`strncpy`	copia los n primeros caracteres de una cadena en otra
`strerror`	devuelve la cadena con el mensaje de error correspondiente al número de error dado

MAS FUNCIONES DE STRING.H

`strlen`	devuelve la longitud de una cadena
`strspn`	devuelve la posición del primer carácter de una cadena que no coincide con ninguno de los caracteres de otra cadena dada
`strcspn`	devuelve la posición del primer carácter que coincide con alguno de los caracteres de otra cadena dada
`strpbrk`	encuentra la primera ocurrencia de alguno de los caracteres de una cadena dada en otra
`strstr`	busca una cadena dentro de otra
`strtok`	parte una cadena en una secuencia de tokens
`strxfrm`	transforma una cadena en su forma de colación (??)
`strrev`	invierte una cadena

EJEMPLOS DE USO DE STRCMP

```c
#include <string.h>
#include <stdio.h>

int main(void)
{
        char *buf1 = "aaa", *buf2 = "bbb", *buf3 = "ccc";
        int ptr;

        ptr = strcmp(buf2, buf1);
        if (ptr > 0)
                printf("buffer 2 is greater than buffer 1\n");
        else
                printf("buffer 2 is less than buffer 1\n");

   ptr = strcmp(buf2, buf3);
   if (ptr > 0)
        printf("buffer 2 is greater than buffer 3\n");
   else
        printf("buffer 2 is less than buffer 3\n");

   return 0;
}
```

EJEMPLO DE USO DE STRCMPI

- strcmpi realiza una comparación del contenido de s1 con s2, sin distinción entre mayúsculas y minúsculas (al igual que stricmp - implementado como una macro).

```c
/* strncmpi example */

#include <string.h>
#include <stdio.h>

int main(void)
{
   char *buf1 = "BBB", *buf2 = "bbb";
   int ptr;

   ptr = strcmpi(buf2, buf1);

   if (ptr > 0)
      printf("buffer 2 is greater than buffer 1\n");

   if (ptr < 0)
      printf("buffer 2 is less than buffer 1\n");

   if (ptr == 0)
      printf("buffer 2 equals buffer 1\n");

   return 0;
}
```

EJEMPLO DE USO DE STRREV

- strrev cambia todos los caracteres de una cadena para invertir el orden, excepto el terminación de carácter nulo. (Por ejemplo, podría cambiar de *string\0* a *gnirts\0*).

  strrev devuelve un puntero a la cadena invertida.

```c
#include <string.h>
#include <stdio.h>

int main(void)
{
   char *forward = "string";

   printf("Before strrev(): %s\n", forward);
   strrev(forward);
   printf("After strrev():  %s\n", forward);
   return 0;
}
```

EJEMPLO DE USO DE STRCHR

- *strchr devuelve un puntero a la primera ocurrencia del caracter c en s, si c no ocurre en s, strchr devuelve un valor nulo.*

```c
#include <string.h>
#include <stdio.h>

int main(void)
{
   char string[15];
   char *ptr, c = 'r';

   strcpy(string, "This is a string");
   ptr = strchr(string, c);
   if (ptr)
      printf("The character %c is at position: %d\n", c, ptr-string);
   else
      printf("The character was not found\n");
   return 0;
}
```

```c
#include <stdio.h>
#include <conio.h>
#include <string.h>

int main()
{
   char s[20] = "Hola amigos";
   char c = 'a';

   printf( "s=%s\t", s );
   printf( "c=%c\n", c );
   printf( "strchr=%s\n", strchr( s, c ) );
   getch();
   return 0;
}
```