Lenguaje de Programación: C++ Arreglos y Strings

José Luis Alonzo Velázquez

Universidad de Guanajuato

Octubre 2010

Cadenas

A diferencia de otros lenguajes de programación que emplean un tipo denominado cadena string para manipular un conjunto de símbolos, en C, se debe simular mediante un arreglo de caracteres, en donde la terminación de la cadena se debe indicar con nulo. Un nulo se especifica como '\0'. Por lo anterior, cuando se declare un arreglo de caracteres se debe considerar un carácter adicional a la cadena más larga que se vaya a guardar. Por ejemplo, si se quiere declarar un arreglo cadena que guarde una cadena de diez caracteres, se hará como:

```
char cadena[11];
```

Inicializaciones

Se pueden hacer también inicializaciones de arreglos de caracteres en donde automáticamente C asigna el carácter nulo al final de la cadena, de la siguiente forma:
char nombre_arr[tam]=“cadena”;

Inicializaciones

Se pueden hacer también inicializaciones de arreglos de caracteres en donde automáticamente C asigna el carácter nulo al final de la cadena, de la siguiente forma:
char nombre_arr[tam]=“cadena”;

Ejemplo

Por ejemplo, el siguiente fragmento inicializa cadena con “hola”:
char cadena[5]=“hola”;
El código anterior es equivalente a:
char cadena[5]='h','o','l','a','\0';

Para asignar la entrada estándar a una cadena se puede usar la función scanf con la opción %s (observar que no se requiere usar el operador &), de igual forma para mostrarlo en la salida estándar.

Ejemplo

```
int main(){
    char nombre[15], apellidos[30];

    printf("Introduce tu nombre: ");
    scanf("%s",nombre);
    printf("Introduce tus apellidos: ");
    scanf("%s",apellidos);
    printf("Usted es %s %s\n",nombre,apellidos);
}
```

El lenguaje C no maneja cadenas de caracteres, como se hace con enteros o flotantes, por lo que lo siguiente no es válido:

```
int main(){
    char nombre[40], apellidos[40], completo[80];

     nombre="Angelina";                    /* Ilegal */
 apellidos="Jolie";          /* Ilegal */
 completo="Actriz"+nombre+appellidos; /* Ilegal */
}
```

Ejemplos de declaración de cadenas de caracteres

```
char *cadena_hola="Hola";//Igual al siguiente
char cadena_hola[]="Hola";//Igual al siguiente
char otro_hola[]={'H','o','l','a','\0'};
char vector[]={'H','o','l','a'}; /*Un vector de 4 elementos
con los elementos 'H','o','l' y 'a' */
char espacio_cadena[1024]="Una cadena en C";
char cadena_vacia[]="";
```

Cómo vimos anteriormente al declarar un arreglo se define la cantidad de elementos que puede contener, en el caso de las cadenas se debe tener en cuenta el espacio adicional necesario para el \0. Viendo el ejemplo, tanto cadena_hola y otro_hola tienen un largo 5 y cadena_vacia tiene un largo de 1.

Buscar las siguientes funciones y ver para que sirven, i.e., crear ejemplos de cada una de ellas.

- strcat
- strcpy
- strlen
- strcmp

Entre las funciones que provee la biblioteca estándar de C, las más importantes son:

- largo = strlen(cadena) // Para obtener el largo de una cadena

Entre las funciones que provee la biblioteca estándar de C, las más importantes son:

- largo = strlen(cadena) // Para obtener el largo de una cadena
- strcpy(destino, origen) // Copia el contenido de origen en destino // destino debe ser lo suficientemente grande

Entre las funciones que provee la biblioteca estándar de C, las más importantes son:

- largo = strlen(cadena) // Para obtener el largo de una cadena
- strcpy(destino, origen) // Copia el contenido de origen en destino // destino debe ser lo suficientemente grande
- strcat(destino, origen) // Agrega el contenido de origen al final de destino // destino debe ser lo suficientemente grander

Entre las funciones que provee la biblioteca estándar de C, las más importantes son:

- largo = strlen(cadena) // Para obtener el largo de una cadena
- strcpy(destino, origen) // Copia el contenido de origen en destino // destino debe ser lo suficientemente grande
- strcat(destino, origen) // Agrega el contenido de origen al final de destino // destino debe ser lo suficientemente grander
- resultado = strcmp(cadena1, cadena2) // Compara dos cadenas // devuelve un valor menor, igual o mayor que 0 según si cadena1 es menor, // igual o mayor que cadena2, respectivamente.

Entre las funciones que provee la biblioteca estándar de C, las más importantes son:

- largo = strlen(cadena) // Para obtener el largo de una cadena
- strcpy(destino, origen) // Copia el contenido de origen en destino // destino debe ser lo suficientemente grande
- strcat(destino, origen) // Agrega el contenido de origen al final de destino // destino debe ser lo suficientemente grander
- resultado = strcmp(cadena1, cadena2) // Compara dos cadenas // devuelve un valor menor, igual o mayor que 0 según si cadena1 es menor, // igual o mayor que cadena2, respectivamente.
- posicion = strchr(cadena, caracter) // Devuelve la posición en memoria de la primer // aparición de caracter dentro de cadena

Entre las funciones que provee la biblioteca estándar de C, las más importantes son:

- largo = strlen(cadena) // Para obtener el largo de una cadena
- strcpy(destino, origen) // Copia el contenido de origen en destino // destino debe ser lo suficientemente grande
- strcat(destino, origen) // Agrega el contenido de origen al final de destino // destino debe ser lo suficientemente grander
- resultado = strcmp(cadena1, cadena2) // Compara dos cadenas // devuelve un valor menor, igual o mayor que 0 según si cadena1 es menor, // igual o mayor que cadena2, respectivamente.
- posicion = strchr(cadena, caracter) // Devuelve la posición en memoria de la primer // aparición de caracter dentro de cadena
- posicion = strstr(cadena,subcadena) // Devuelve la posición en memoria de la primer // aparición de subcadena dentro de cadena

Ejemplo: Función que compara cadenas
Librería necesaria string.h

```
bool compara_cadenas(char s1[],char s2[]){
    int n=strlen(s1);
    int m=strlen(s2);
    if(n!=m){
        return false;
    }
    for(int i=0;i<n;i++){
        if(s1[i]!=s2[i]){
            return false;
        }
    }
    return true;
}
```

 Programming Principles and Practice Using C++, Bjarne Stroustrup.

 http://www.codeblocks.org

 http://www.wxwidgets.org

 (O'Reilly) Practical C Programming (3rd Edition)