



Universidad Católica de Honduras

“Nuestra Señora Reina de la Paz”

Asignatura

Programación Científica

Tema de Exposición

Caracteres y Cadenas

Catedrático: Ing. Henry Pinto

Tegucigalpa, M. D. C.

01 de Marzo

del 2008



Universidad Católica de Honduras
“Nuestra Señora Reina de la Paz”

Asignatura

Programación Científica

Tema de Exposición

Caracteres y Cadenas

Equipo No: 6

Integrantes:

Isis Irias

Tg-200320007

Gustavo Turcios

Tg-200510150

Javier Iscoa

110610418

Tegucigalpa, M. D. C.

01 de Marzo

del 2008

Caracteres Y Cadenas

En matemáticas o en programación, una cadena de caracteres, palabra o frase (String en inglés) es una secuencia ordenada de longitud arbitraria (aunque finita) de elementos que pertenecen a un cierto alfabeto. En general, una cadena de caracteres es una sucesión de caracteres (letras, números u otros signos o símbolos).

En matemáticas es habitual usar las letras w, x, y, \dots para referirnos a las cadenas. Por ejemplo, si tenemos un alfabeto $\Sigma = \{a, b, c\}$, una cadena podría ser: $x = aacbbcba$.

Desde un punto de vista de la programación, si no se ponen restricciones al alfabeto, una cadena podrá estar formada por cualquier combinación finita de todo el juego caracteres disponibles (las letras de la 'a' a la 'z' y de la 'A' a la 'Z', los números del '0' al '9', el espacio en blanco ' ', símbolos diversos '!', '@', '%', etc.). En este mismo ámbito (el de la programación), se utilizan normalmente como un tipo de dato predefinido, para palabras, frases o cualquier otra sucesión de caracteres. En este caso, se almacenan en un vector de datos, o matriz de datos de una sola fila (array en inglés). Las cadenas se pueden almacenar físicamente:

1. Seguidas.
2. Enlazada letra a letra.

Generalmente son guardados un carácter a continuación de otro por una cuestión de eficiencia de acceso.

Un caso especial de cadena es la que contiene cero caracteres, a esta cadena se la llama cadena vacía.

Cadenas

Las cadenas de caracteres son un tipo especial de arreglo pues se trata de un conjunto de datos de tipo char que termina con un caracter nulo, a este tipo de cadenas también se les conoce como "cadenas". Básicamente el manejo de cadenas es muy similar al de los arreglos.

Ejemplo 1:



```
f:\vname00.cpp
#include<stdio.h>
#include<string.h>
main() {
char cadena[6]; /* Define una cadena de caracteres */
cadena[0]='L';
cadena[1]='e';
cadena[2]='t';
cadena[3]='\n';
cadena[4]='a';
cadena[5]='s';
cadena[6]=0;

printf("La cadena es %s\n" , cadena);
printf("La tercera letra de la cadena es: %s\n", &cadena[2]);
printf("Una parte de la cadena es : %sn", &cadena[3]);
return 0;
}
```

Imagen 1

Como se muestra en la imagen 1 podemos observar la codificación necesaria de los tipos de datos para la utilización de las cadenas. A continuación el resultado del ejemplo:

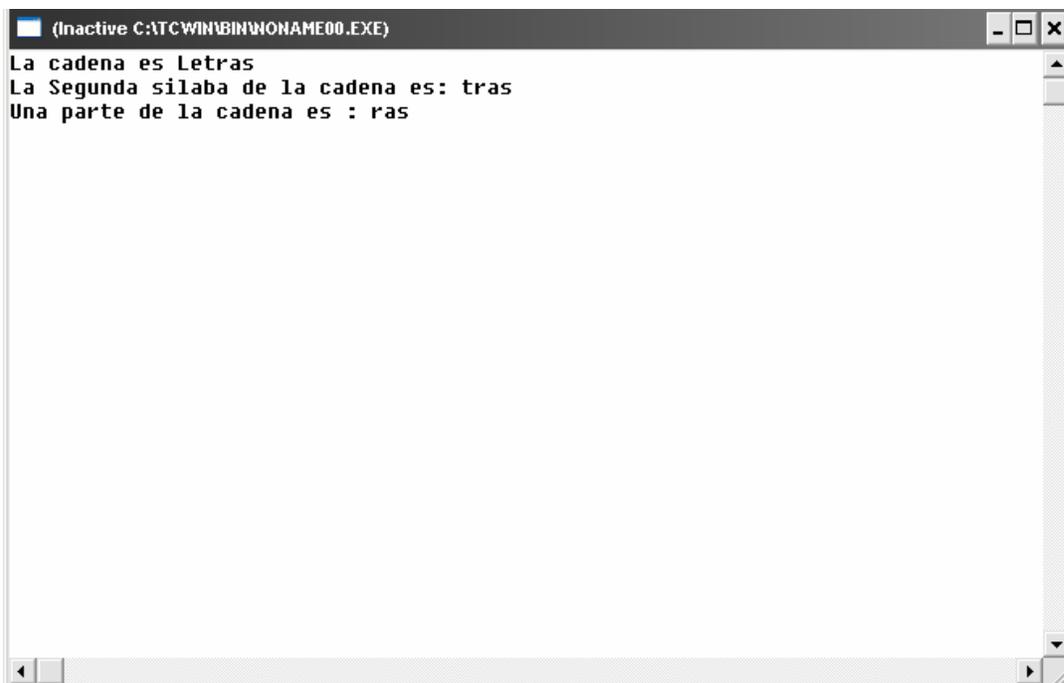


Imagen 2

La variable cadena es una cadena que puede almacenar hasta seis caracteres, tomando en cuenta que se requiere un espacio para almacenar el caracter nulo al final de la cadena. El símbolo %s mostrado en los enunciados printf() le indica al sistema que despliegue una cadena de caracteres empezando con el elemento subíndice cero, que en el código de ejemplo es la letra L, y continuando hasta encontrar el caracter nulo. Observe que en los enunciados printf() cuando se indica la variable cadena sin corchetes indica que se despliegue la totalidad de la cadena, en tanto que al indicar la variable cadena con algún valor entre corchetes se refiere a un solo elemento de la cadena, en este caso debemos utilizar en el enunciado printf() el símbolo %c que le indica al sistema que despliegue un solo carácter. El símbolo & especifica la dirección en memoria de cadena[3], este símbolo lo estudiaremos mas adelante. A continuación se muestra un ejemplo del manejo de cadenas.

Ejemplo2:



```
f:\nname00.cpp
#include<stdio.h>
#include<string.h>

main() {
char cadena1[17], cadena2[13], titulo[26], prueba[29];
strcpy(cadena1, "Pedro Picapiedra ");
strcpy(cadena2, "Pablo Marmol");
strcpy(titulo, "-- Los Picapiedras --");

printf("%s", titulo);
printf("\n\nLos personajes principales son:\n %s", cadena1);
printf("y %s", cadena2);

if(strcmp(cadena1, cadena2) > 0)
strcpy(prueba, cadena1);
else
strcpy(prueba, cadena2);
printf("\n\nLa cadena mas grande es:\n", prueba);

strcpy(prueba, cadena1);
strcat(prueba, " y ");
strcat(prueba, cadena2);
printf("%s son vecinos", prueba);
return 0;
}
```

Imagen 3

Como puede ver, en la imagen 3 se han definido cuatro arreglos de tipo char de diferente longitud, enseguida nos encontramos con la función `strcpy()` que sirve para copiar la cadena especificada en la segunda entidad dentro del paréntesis de la función en un arreglo de tipo char especificado por la primera entidad dentro del paréntesis de la función `strcpy`, de esta forma, por ejemplo, la cadena "Pedro Picapiedra" se copia en el arreglo de tipo char llamado `cadena1`.

Mas adelante en el código nos encontramos con la función `strcmp()` que como es fácil adivinar, sirve para comparar, letra por letra, dos cadenas especificadas dentro del paréntesis. Esta función devuelve 1 si la primera cadena es mayor que la segunda, es decir, si tiene mayor cantidad de letras. Si ambas cadenas son iguales la función devuelve 0, en tanto que si la primera cadena es menor que la segunda entonces el valor devuelto es -1.

Siguiendo en el ámbito de la informática, al considerar las cadenas como un tipo de datos, hay que definir (o conocer) cuales son las operaciones que podemos hacer con ellas, en principio estas podrían ser muchas y llegar a ser muy sofisticadas, pero las que podríamos considerar básicas son:

Concatenación: Consiste en unir dos cadenas o más (o una cadena con un carácter) para formar una cadena de mayor tamaño.

Búsqueda: Consiste en localizar dentro de una cadena una subcadena más pequeña o un carácter.

Extracción: Se trata de sacar fuera de una cadena una porción de la misma según su posición dentro de ella. Se verá a continuación en la imagen 4 el resultado de la imagen 3:

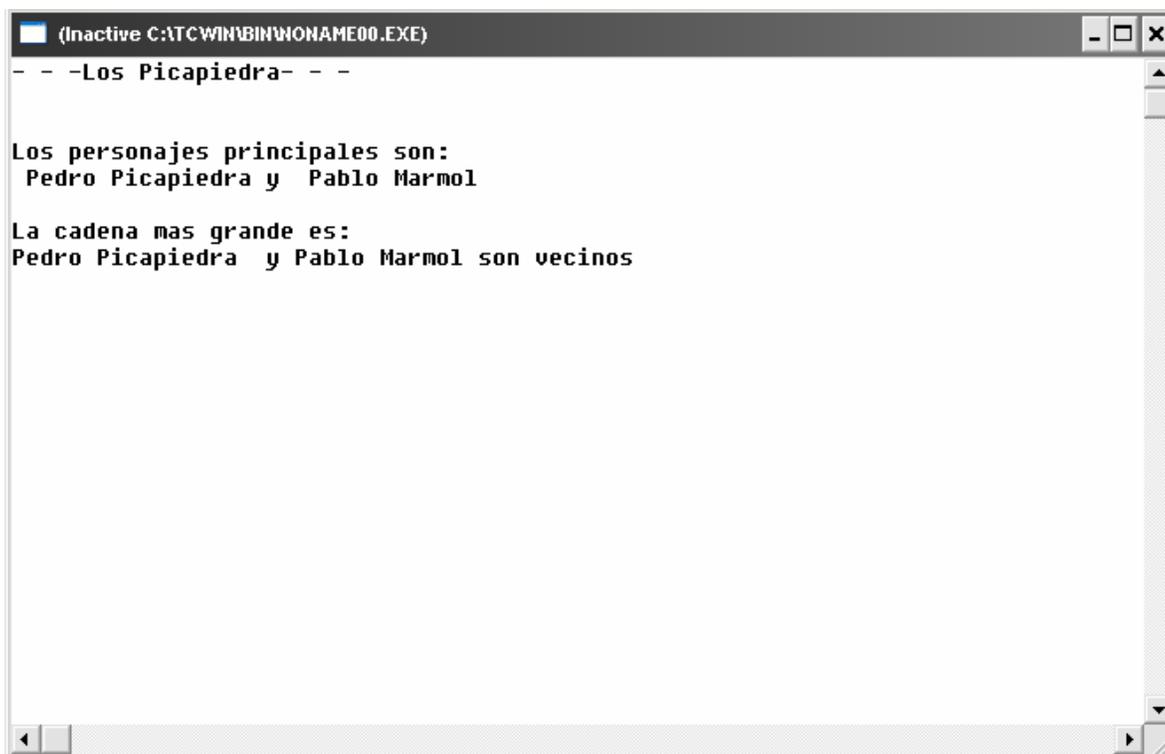


Imagen 4

(Operaciones con cadenas en el lenguaje C)

Representación

Una cadena suele ser representada entre comillas dobles superiores ("palabra"), mientras que un carácter de esa cadena (un char en inglés) suele ser representado entre comillas simples ('p'). Por ejemplo, en C:

```
char c = 'a';  
char str[5] = "hola";
```

Generalmente para acceder a un carácter en una posición determinada se suele usar la forma `variable[posición]` como cuando se accede a un vector.

Para poder mostrar una comilla (") dentro de la cadena y no tener problemas con las comillas que la delimitan, se usan secuencias de escape. Esto se aplica a otros caracteres reservados o no imprimibles como el retorno de carro. No obstante, las expresiones para producir estas secuencias de escape dependen del lenguaje de programación que se esté usando. Una forma común, en muchos lenguajes, de escapar un carácter es anteponiéndole un «\» (sin comillas), p. e.: «\"» (sin comillas).

Cadenas dinámicas y estáticas

Las cadenas pueden ser de naturaleza dinámica (pueden alterar su longitud durante el tiempo de ejecución), o de naturaleza estática (su longitud es fija a lo largo del tiempo de ejecución). En este segundo caso el programador debe prever que al recorrer la cadena los índices no se vayan de los límites previstos (C no permite que las cadenas crezcan automáticamente de forma explícita, mientras que C# sí).

El final de la cadena se delimita de diferente manera en uno u otro caso:

Mediante un carácter de fin de cadena ("\0" en C) para las cadenas de tipo dinámico.

Mediante una propiedad de la cadena que delimite su longitud (Count en C#) para las de tipo estático.

Algunas operaciones comunes

Concatenación: unir dos cadenas de caracteres.

```
$pareja = "Joshua"." y "."Lidia" # en Perl y PHP;  
pareja = "Luisa" & " y " & "Carmen" # en Visual Basic;  
pareja = "Luisa" + " y " + "Carmen"; # en C++ y Java con la clase String.
```

Multiplicar una cadena: repetir una cadena un número de veces

```
$puntos = "." x 5 # pone 5 puntos en Perl
```

Por último tenemos la función `strcat()` que ejecuta una concatenación de cadenas, es decir, copia la segunda cadena especificada dentro del paréntesis de la función enseguida de la primera cadena especificada, agregando un carácter nulo al final de la cadena resultante.

Como ya se mencionó se debe especificar la terminación de la cadena con el carácter nulo `"\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];
```

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";
```

Por ejemplo, el siguiente fragmento inicializa cadena con `"\0hola"`:

```
char cadena[5]="hola";
```

El código anterior es equivalente a:

```
char cadena[5]={'h','o','l','a',' '};
```

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.

Por ejemplo:



```
f:\noname00.cpp
#include<stdio.h>
#include<string.h>
main() {
char nombre[15], apellidos[45],segnombre[30], segapellidos[60];

printf("Introduce tu nombre: ");
scanf("%s",nombre);
printf("Introduce tu segundo nombre: ");
scanf("%s",segnombre);
printf("Introduce tu apellidos: ");
scanf("%s",apellidos);
printf("Introduce tu segundo apellido: ");
scanf("%s",segapellidos);
printf("\nUsted es %s %s %s %s",nombre, segnombre, apellidos ,segapellidos );

return 0;
}
```

Imagen 4



Imagen 4

Arreglos

Un array (también conocido como arreglo, vector o matriz) es una colección de variables relacionadas a las que se hace referencia por medio de un nombre en común.

Es un modo de manejar una gran cantidad de datos del mismo tipo bajo un mismo nombre o identificador. En esta práctica utilizaremos cadenas unidimensionales (vectores y cadenas de caracteres).

- Su forma general es: tipo nombre[tamaño]

Ejemplo double datos[10] : En esta sentencia se reserva espacio para 10 variables de tipo double, las cuales se van a manejar por medio del nombre datos y un índice, el cual en C++ siempre empieza por "cero".

- Los elementos se enumeran desde 0 hasta (n-1). Hay que tener mucho cuidado de no sobre-pasar las dimensiones del array, en cuyo caso daría error el programa.

Si queremos acceder al primer elemento del ejemplo anterior: datos[0]=2.5; Al segundo: datos[1]=4.5; Y así sucesivamente hasta el último valor: datos[9]=3.5

Arreglos Unidimensionales Y Bidimensionales

Los arreglos permiten agrupar datos usando un mismo identificador. Todos los elementos de un arreglo son del mismo tipo, y para acceder a cada elemento se usan subíndices. En el siguiente capítulo se presentan los arreglos y las cadenas. Las cadenas se consideran como un arreglo de tipo char.

Los arreglos son una colección de variables del mismo tipo que hacen referencia utilizando un nombre común. Un arreglo consta de posiciones de memoria contigua. La dirección más baja corresponde al primer elemento y la más alta al último. Un arreglo puede tener una o varias dimensiones a los arreglos unidimensionales se les conocen como vectores y a los bidimensionales como matrices. Para acceder a un elemento en particular de un arreglo se usa un índice. El formato para declarar un arreglo unidimensional es:

```
tipo nombre_arr [ tamaño ]
```

Por ejemplo, para declarar un arreglo de enteros llamado listanum con diez elementos se hace de la siguiente forma:

```
int listanum[10];
```

En C, todos los arreglos usan cero como índice para el primer elemento. Por tanto, el ejemplo anterior declara un arreglo de enteros con diez elementos desde listanum[0] hasta listanum[9]. La forma como pueden ser accesados los elementos de un arreglo, es de la siguiente forma:

```
listanum[2] = 15; /* Asigna 15 al 3er elemento del arreglo listanum*/
```

```
num = listanum[2]; /* Asigna el contenido del 3er elemento a la variable num */
```

El lenguaje C no realiza comprobación de contornos en los arreglos. En el caso de que sobrepase el final durante una operación de asignación, entonces se asignarán valores a otra variable o a un trozo del código el cual es una condición no deseada. Como programador se es responsable de asegurar que todos los arreglos sean lo suficientemente grandes para guardar lo que pondrá en ellos el programa.

Para declarar arreglos con más de una dimensión, el formato general es:

```
tipo nombre_arr [ tam1 ][ tam2 ] ... [ tamN];
```

Por ejemplo un arreglo de enteros bidimensionales se escribirá como:
`int tabladenums[50][50];`

Observar que para declarar cada dimensión lleva sus propios paréntesis cuadrados. Para acceder los elementos se procede de forma similar al ejemplo del arreglo unidimensional.

`tabladenums[2][3] = 15; /* Asigna 15 al elemento de la 3ª fila y la 4ª columna*/`
`num = tabladenums[25][16];`

Si se desea la inicializar arreglos, es decir asignar valores al arreglo en el momento de ser declarados debe hacerse con el siguiente formato:
`tipo nombre_arr[tam1][tam2] ... [tamN] = {lista-valores};`

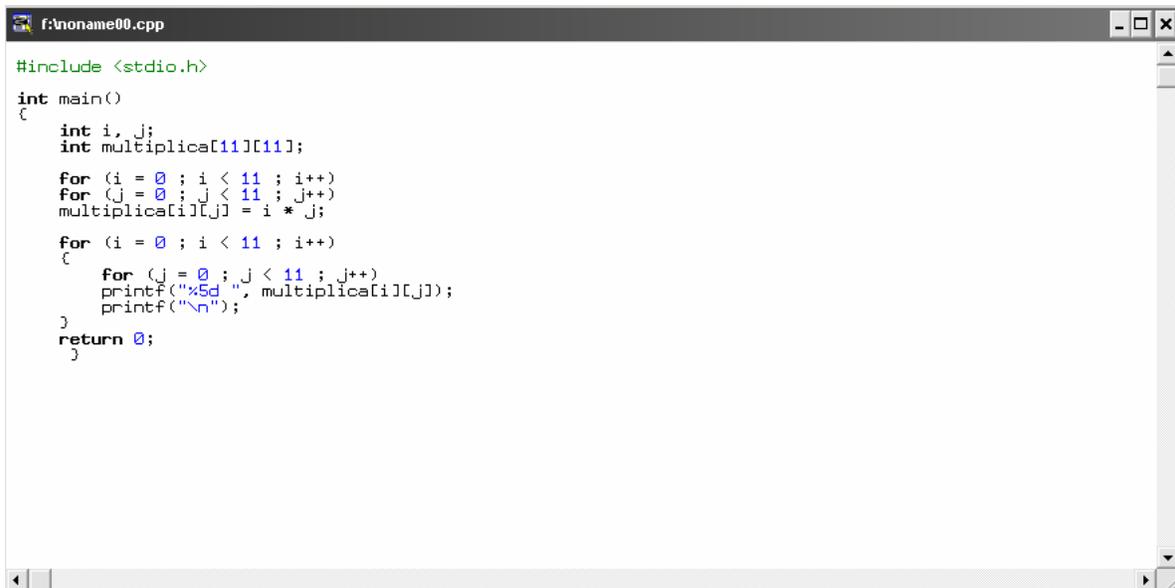
Ejemplo:

`int i[10] = {1,2,3,4,5,6,7,8,9,10};`

`int num[3][4]= {0,1,2,3,4,5,6,7,8,9,10,11};`

Los valores para el número de elementos deben ser constantes, y se pueden usar tantas dimensiones como queramos, limitado sólo por la memoria disponible.

Ejemplo4:



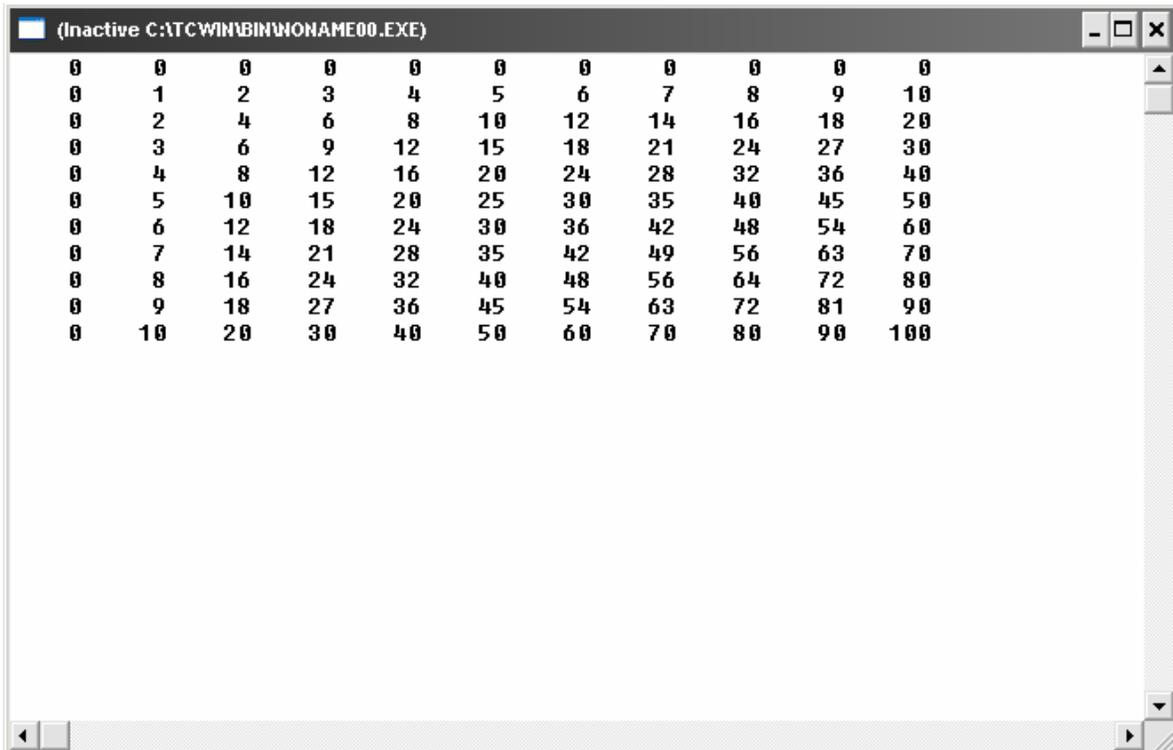
```
#include <stdio.h>

int main()
{
    int i, j;
    int multiplica[11][11];

    for (i = 0 ; i < 11 ; i++)
        for (j = 0 ; j < 11 ; j++)
            multiplica[i][j] = i * j;

    for (i = 0 ; i < 11 ; i++)
    {
        for (j = 0 ; j < 11 ; j++)
            printf("%5d ", multiplica[i][j]);
        printf("\n");
    }
    return 0;
}
```

Imagen 5



The image shows a screenshot of a Windows command prompt window. The title bar reads "(Inactive C:\TC\WIN\BIN\WONAME00.EXE)". The window contains a multiplication table of numbers from 0 to 100, arranged in 11 rows and 11 columns. The numbers are displayed in a monospaced font, with each number occupying a fixed width. The table is as follows:

0	0	0	0	0	0	0	0	0	0	0
0	1	2	3	4	5	6	7	8	9	10
0	2	4	6	8	10	12	14	16	18	20
0	3	6	9	12	15	18	21	24	27	30
0	4	8	12	16	20	24	28	32	36	40
0	5	10	15	20	25	30	35	40	45	50
0	6	12	18	24	30	36	42	48	54	60
0	7	14	21	28	35	42	49	56	63	70
0	8	16	24	32	40	48	56	64	72	80
0	9	18	27	36	45	54	63	72	81	90
0	10	20	30	40	50	60	70	80	90	100

Imagen 6

Bibliografía

- Categorías: [Wikipedia:Esbozo matemática](#) | [Wikipedia:Esbozo programación](#) | [Tipos de datos básicos](#) | [Lenguajes formales](#)
- [Cool C-C++ Program](#)
- [Acción en C-C++ - Tutoriales.mht](#)
- [Departament d'Informàtica](#)
[Departamento de Informática, PRÀCTICA 11, Curso 2002-2003](#)
- [FACULTAT DE MATEMATIQUES, UNIVERSITAT DE VALENCIA](#)

Anexos

<http://www.yousendit.com>

www.monografias.com

<http://fileupyours.com>

<http://www.rincondelvago.com/>

<http://www.transferbigfiles.com>