Laboratorio de Programación

Lenguaje C : Argumentos de Línea de Comandos

Luis Garreta
luis.garreta@javerianacali.edu.co

Ingeniería de Sistemas y Computación
IlPontificia Universidad Javeriana – Cali

24 de febrero de 2017

Opciones de la línea de comandos: parámetros de “main”

- Es muy frecuente que un programa que usamos desde la “línea de comandos” tenga ciertas opciones que le indicamos como argumentos.
- Por ejemplo, bajo Linux, podemos ver la lista detallada de ficheros que terminan en .c llamando al comando ”ls” con dos parámetros: ”-l” y ”*.c”:

```
1 $ ls -l *.c
```

- O cuando compilas un programa con ”gcc” este se llama con tres parámetros: ”taller.c”, ”-o”, y ”taller.bin”

```
1 $ gcc taller.c -o taller.bin
```

Argumentos de main: argc y argv

Estas opciones que se le pasan al programa se pueden leer desde C. La forma de hacerlo es con dos parámetros: **argc y argv**.

- El primero: **argc:** es un número entero que indica cuantos argumentos se han tecleado.
- El segundo: **argv:** es una tabla de cadenas de texto, que contiene cada uno de esos argumentos.

Tenga en cuenta que los elementos de argv son todos cadenas

Si necesita envíar un número como argumento, este se tomará como cadena. Lo que debe hacer dentro de su programa es convertir la cadena a entero (o flotante), usando las funcione *atoi y atof*, así

```
char *cadValor = argv [2];
int valor = atoi (cadValor);
```

Ejemplo de llamado

Por ejemplo, si bajo Windows o MsDos tecleamos la orden “DIR *.EXE /P”, tendríamos que:

```
1 C:\Documentos> DIR *.EXE /P
```

Lo que tendríamos en los parámetros argc y argv sería:

- argc es la cantidad de parámetros, incluyendo el nombre del propio programa (3, en este ejemplo).
- argv[0] es la cadena con el nombre del programa "DIR"
- argv[1] es el primer argumento con la cadena "*.EXE"
- argv[2] es el segundo argumento con la cadena "/P"

Ejemplo de Programa para la funcion copiarNCaractateres

```
1  char *copiarNCaracteres (char *cadOrigen, char *cadDestino,
       int n) {
2    int i;
3    for (i=0; i <= n; i++)
4      cadDestino [i] = cadOrigen [i];
5
6    return cadDestino;
7  }
8
9  int main (int argc, char *argv[]) {
10   char *cad1 = argv [1];
11   char *cadN = argv [2];
12   char *cad2;
13   int n = atoi (cadN);
14
15   cad2 = (char *)malloc (n);
16   copiarNCaracteres (cad1, cad2, n);
17
18   printf ("\n%s\n", cad2);
19   return 0;
20 }
```