



INFORMATICA I

Arreglos o " **vectores** "

Ing. Juan Carlos Cuttitta

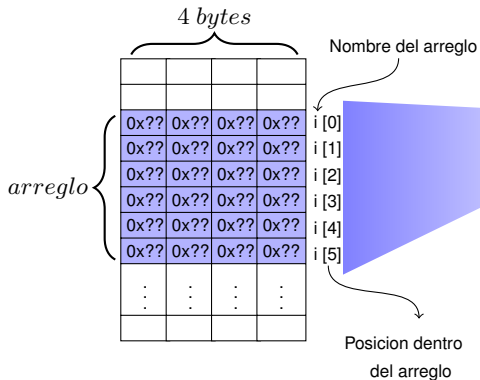
*Universidad Tecnológica Nacional
Facultad Regional Buenos Aires
Departamento de Ingeniería Electrónica*

24 de mayo de 2020

Declaración y disposición en memoria

Arquitectura X86-32 bits

Disposición de la variable *i*
en memoria



Código en programa fuente

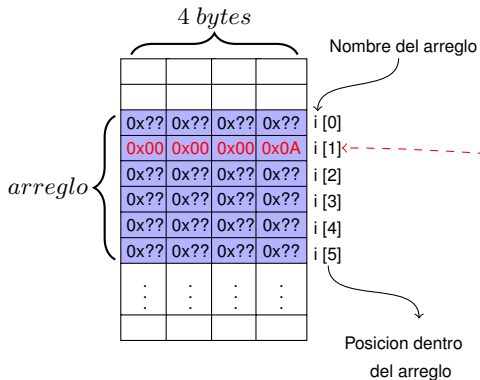
```
1  int main ()
2  {
3      int i [6]; ← Declara arreglo
4      .....
5      i [1]=10;
6      .....
7  }
```

Declara arreglo de 6 enteros

Acceso al contenido

Arquitectura X86-32 bits

Disposición de la variable *i*
en memoria



Código en programa fuente

```
1  int main ()
2  {
3      int i[6];
4      .....
5      i[1]=10;
6      .....
7  }
```

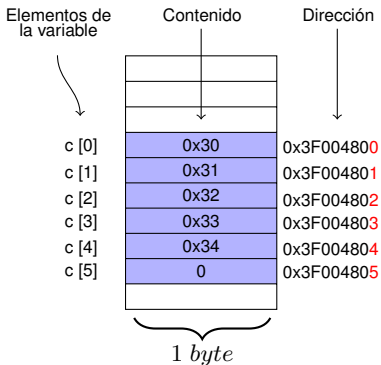
Asigna el valor 10
al elemento 1
del arreglo

Dirección no es lo mismo que orden de elemento

Declaración en código

`unsigned char c[6]`

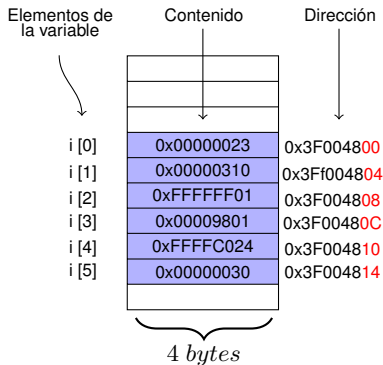
Cardena de caracteres



Declaración en código

`int c[6]`

Arreglo de enteros



Dirección no es lo mismo que orden de elemento

Declaración en código

`unsigned char c[6]`

Cardena de caracteres

Elementos de la variable	Contenido	Dirección
c [0]	0x30	0x3F004800
c [1]	0x31	0x3F004801
c [2]	0x32	0x3F004802
c [3]	0x33	0x3F004803
c [4]	0x34	0x3F004804
c [5]	0	0x3F004805

1 byte

*notar la diferencia
uno termina con '\0' y el otro no*

Declaración en código

`int c[6]`

Arreglo de enteros

Elementos de la variable	Contenido	Dirección
i [0]	0x00000023	0x3F004800
i [1]	0x00000310	0x3F004804
i [2]	0xFFFFFFFF01	0x3F004808
i [3]	0x00009801	0x3F00480C
i [4]	0xFFFFC024	0x3F004810
i [5]	0x00000030	0x3F004814

4 bytes

Dirección no es lo mismo que orden de elemento

Declaración en código

`unsigned char c[6]`

Cardena de caracteres

Elementos de la variable

Contenido

Dirección

c [0]	0x30='0'	0x3F004800
c [1]	0x31='1'	0x3F004801
c [2]	0x32='2'	0x3F004802
c [3]	0x33='3'	0x3F004803
c [4]	0x34='4'	0x3F004804
c [5]	0='\0'	0x3F004805

1 byte

*notar la diferencia
uno termina con '\0' y el otro no*

Declaración en código

`int c[6]`

Arreglo de enteros

Elementos de la variable

Contenido

Dirección

i [0]	0x00000023	0x3F004800
i [1]	0x00000310	0x3F004804
i [2]	0xFFFFFFFF01	0x3F004808
i [3]	0x00009801	0x3F00480C
i [4]	0xFFFFC024	0x3F004810
i [5]	0x00000030	0x3F004814

4 bytes

Un caso especial de arreglo

Arreglo de caracteres

Se trata de un tipo muy usual de dato que llamamos cadena o string (del inglés).

Se inicializa de los siguientes modos:

```
1 /*Una forma de inicializar un arreglo de caracteres con una
   cadena*/
2
3 char cad []= "Hola mundo!";
4
5 /* Otra forma... */
6
7 char cad []={'H','o','l','a',' ','m','u','n','d','o','!','\0'};
```

Un caso especial de arreglo

Arreglo de caracteres

Se trata de un tipo muy usual de dato que llamamos cadena o string (del inglés).

Se inicializa de los siguientes modos:

```
1 /*Una forma de inicializar un arreglo de caracteres con una
   cadena*/
2
3 char cad []= "Hola mundo!";
4
5 /* Otra forma...*/
6
7 char cad []={'H','o','l','a',' ','m','u','n','d','o','!','\0'};
```

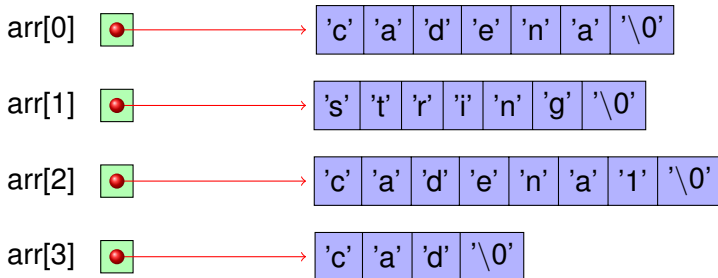

En código....

```
1  /* Utiliza una lista de inicialización para
   2     inicializar el arreglo arr.*/
3  int arr[8] = {23,0,-669,-1,995,1277,90,-9000};
4  /* La cantidad de elementos en la línea anterior
   5     es redundante. Puede hacerse lo mismo de la
   6     siguiente forma*/
7  int arr[] = {23,0,-669,-1,995,1277,90,-9000};
8  /* Si lo vamos a inicializar con el mismo valor
   9     para todos los elementos, la forma adecuada
  10    es la siguiente*/
11
12 int arr[8], i;
13 for (i = 0 ; i < 8 ; i++)
14 {
15     arr[i] = 0;
16 }
```

Arreglos de punteros

```
1 char *arr[4] = { "cadena", "string", "cadena1", "cad" };
```

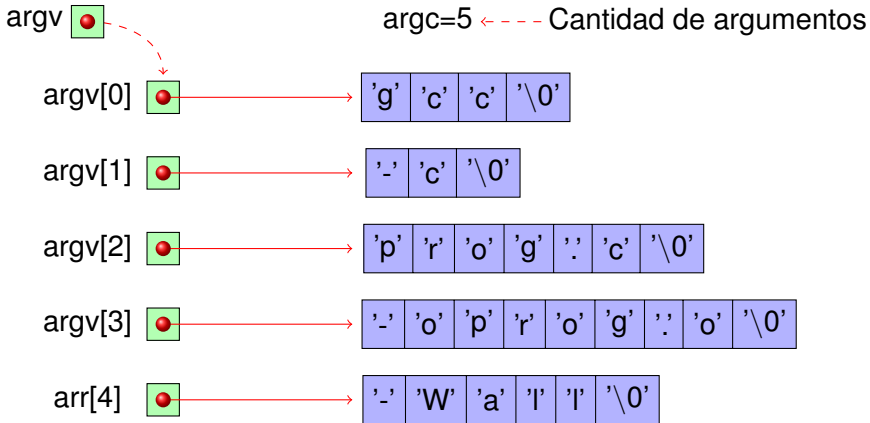
Genera esta disposición en memoria



main, la función multifacética

Ejemplo con función conocida *gcc -c prog.c -oprogram.o -Wall*

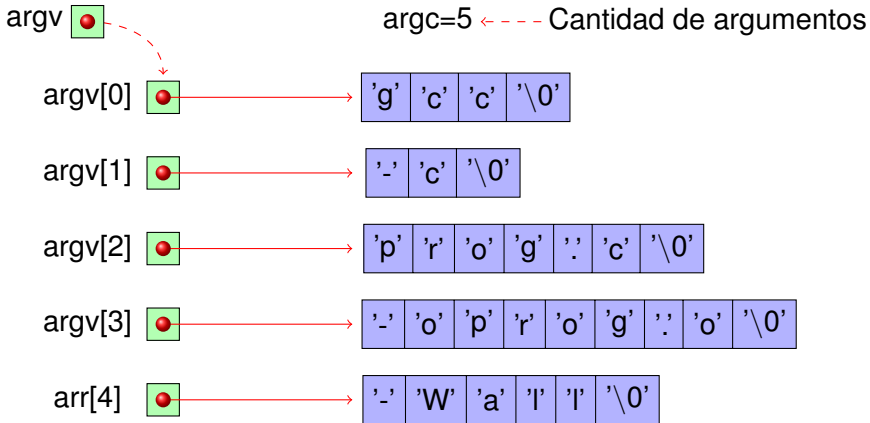
```
int main (int argc, char *argv[])
```



main, la función multifacética

Ejemplo con función conocida **gcc -c prog.c -oprogram.o -Wall**

```
int main (int argc, char **argv)
```



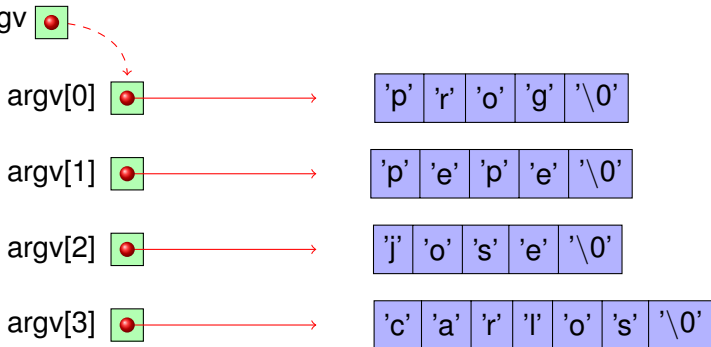
Ejemplo de *array de direcciones*

```
./prog pepe jose carlos
```

Los argumentos del programa son:

- pepe
- jose
- carlos

argv



Arquitectura X86-32 bits

argv 0xFFE07650

0xFFE07650

argv[0]	0xFFF47600
argv[1]	
argv[2]	
argv[3]	

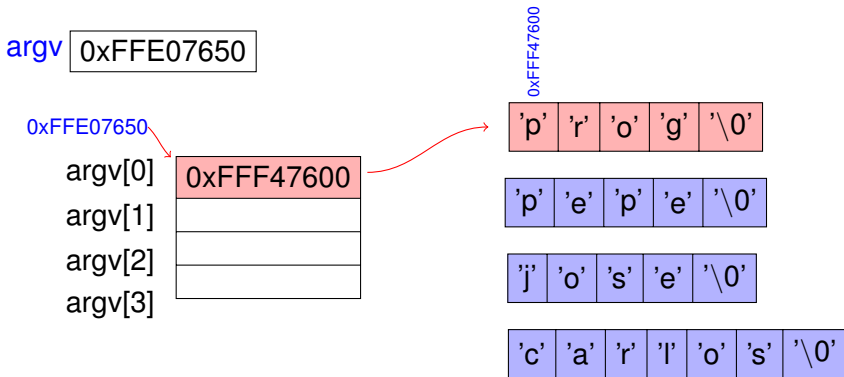
'p' 'r' 'o' 'g' '\0'

'p' 'e' 'p' 'e' '\0'

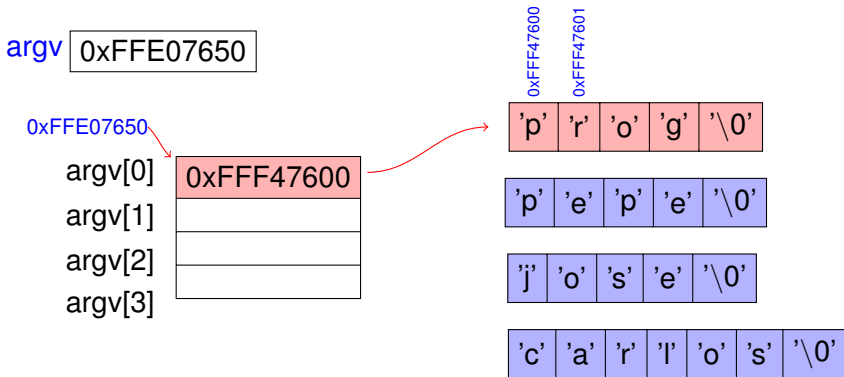
'j' 'o' 's' 'e' '\0'

'c' 'a' 'r' 'l' 'o' 's' '\0'

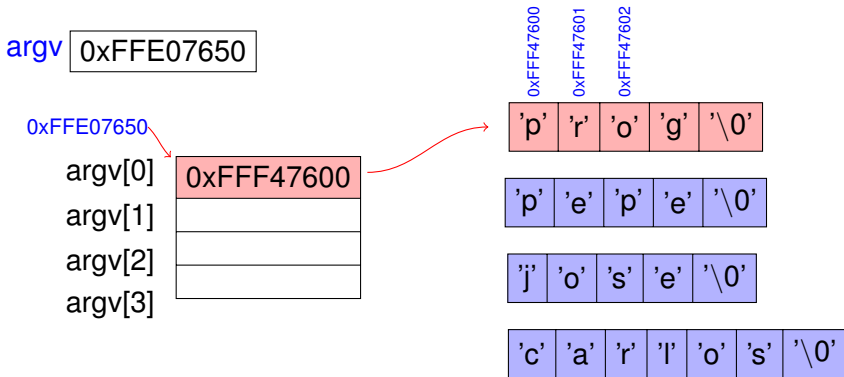
Arquitectura X86-32 bits



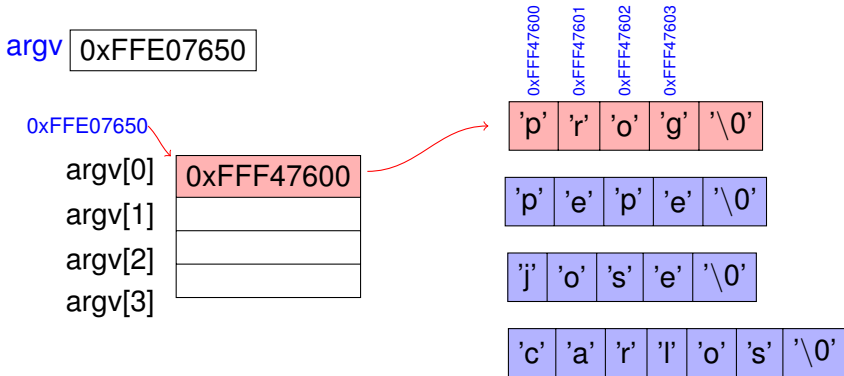
Arquitectura X86-32 bits



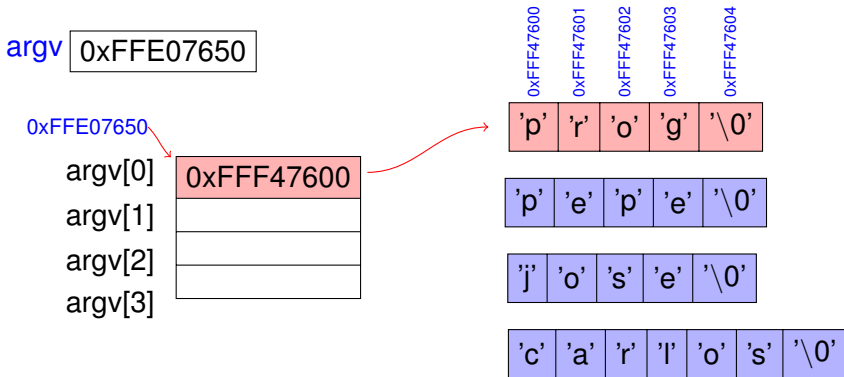
Arquitectura X86-32 bits



Arquitectura X86-32 bits



Arquitectura X86-32 bits



Arquitectura X86-32 bits

argv 0xFFE07650

0xFFE07650

argv[0]	0xFFF47600
argv[1]	0xFFF47605
argv[2]	
argv[3]	

'p' 'r' 'o' 'g' '\0'

'p' 'e' 'p' 'e' '\0'

'j' 'o' 's' 'e' '\0'

'c' 'a' 'r' 'l' 'o' 's' '\0'

Arquitectura X86-32 bits

argv 0xFFE07650

0xFFE07650

argv[0]	0xFFF47600
argv[1]	0xFFF47605
argv[2]	0xFFF4760A
argv[3]	

'p' 'r' 'o' 'g' '\0'

'p' 'e' 'p' 'e' '\0'

'j' 'o' 's' 'e' '\0'

'c' 'a' 'r' 'l' 'o' 's' '\0'

Arquitectura X86-32 bits

argv 0xFFE07650

0xFFE07650

argv[0]	0xFFF47600
argv[1]	0xFFF47605
argv[2]	0xFFF4760A
argv[3]	0xFFF4760F

'p' 'r' 'o' 'g' '\0'

'p' 'e' 'p' 'e' '\0'

'j' 'o' 's' 'e' '\0'

'c' 'a' 'r' 'l' 'o' 's' '\0'