

Tema 8: Herramientas

Argumentos en línea de órdenes

Concepto



- Algunos programas pueden requerir argumentos. Por ejemplo, `gcc` es un programa que requiere argumentos como el tipo de operación, los ficheros fuentes, el fichero ejecutable, ...

`gcc -o leeFichero.exe leeFichero.c`

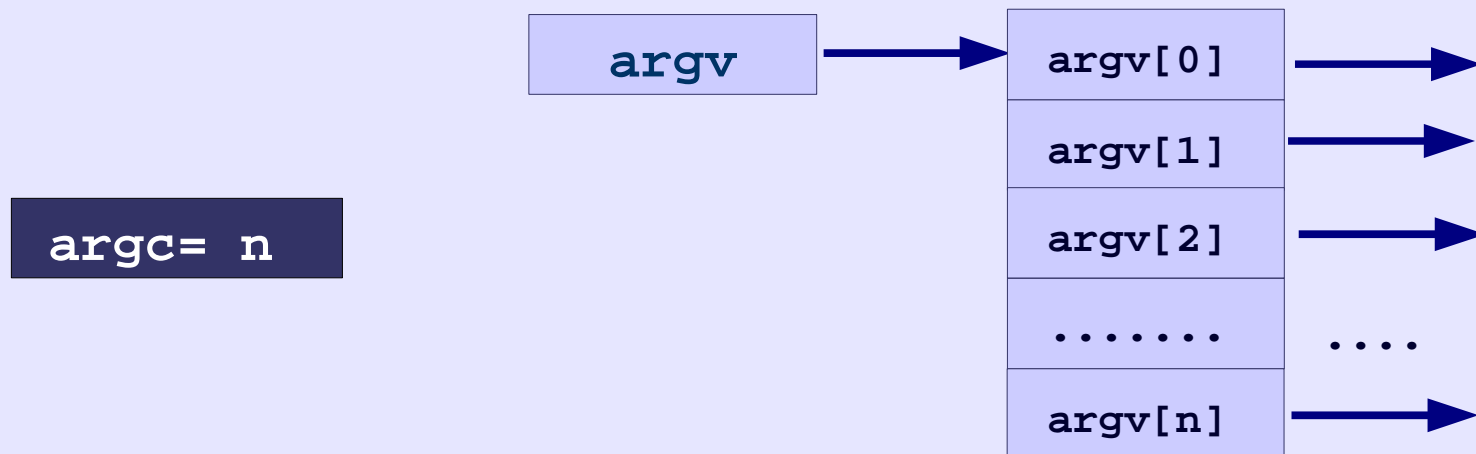
programa *argumentos*

- Podemos construir programas que reciban argumentos cuando son llamados desde la línea de órdenes
 - ◆ Programa que lee un fichero
- `leeFichero.exe alumnos.txt`
- ◆ En vez de pedir dentro del programa el fichero que queremos leer, se lo pasamos al programa cuando lo ejecutamos.

Concepto



- ¿Cómo hacemos que nuestros programas reciban argumentos?
 - ◆ Cuando se llama al *main*, se le invoca con dos argumentos
 - **argc** (*argument count*). Es el número de argumentos con los que se invocó al programa desde la línea de órdenes
 - **argv** (*argument vector*). Es un puntero a un vector de cadenas de caracteres que contiene los argumentos, uno por cadena.



Concepto

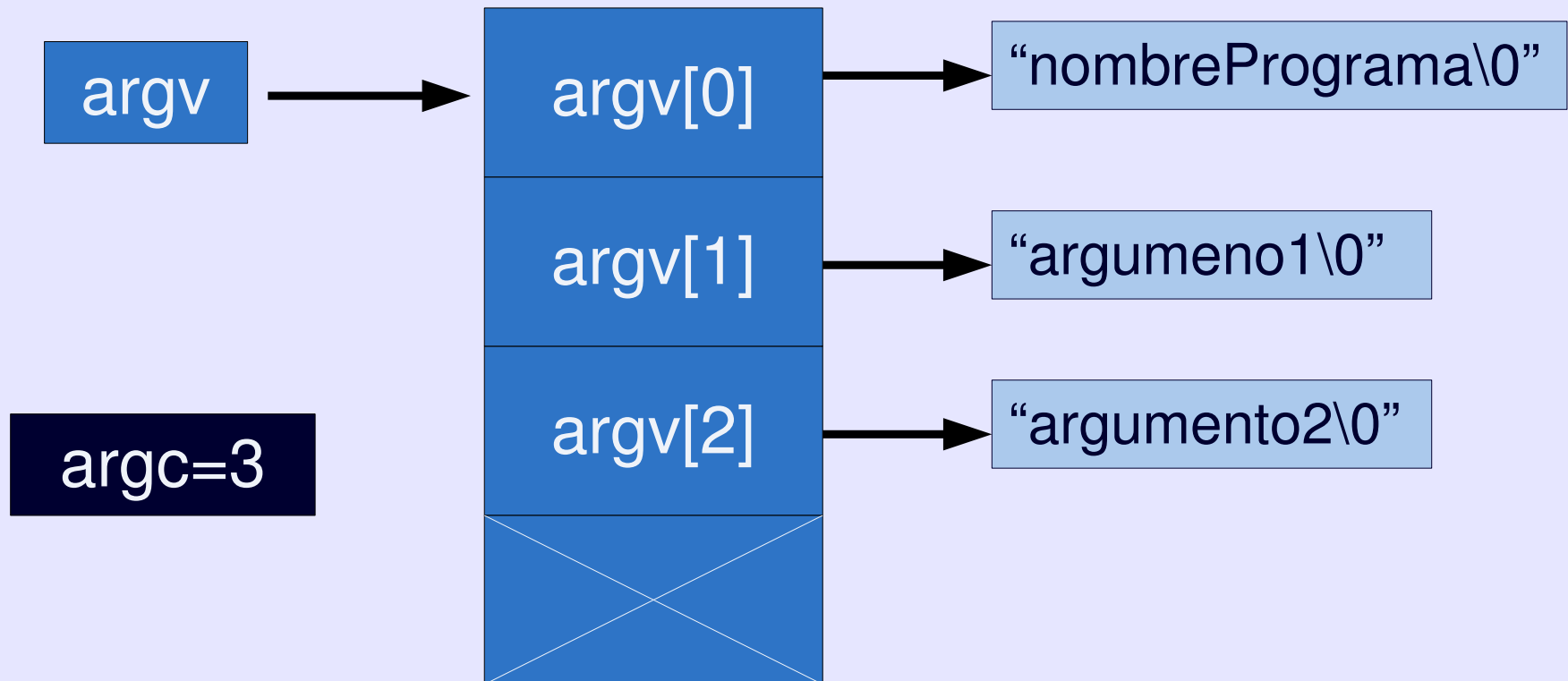


- ***argv[0]*** es el nombre con el que se invocó al programa, por lo que ***argc*** es por lo menos 1.
- Si ***argc*** es 1, entonces no hay argumentos en la línea después del nombre del programa.
 - ◆ No se ha llamado al programa con argumentos
- El primer argumento optativo es ***argv[1]*** y el último ***argv[argc-1]***.
- El estándar requiere que ***argv[argc]*** sea un puntero a NULL.

Concepto



bash\$ nombrePrograma argumento1 argumento2



Ejemplo



args.c

- *Imprime por pantalla el nombre del programa y los argumentos*

```
int main(int argc, char ** argv) {  
    //int main(int argc, char * argv[])  
    int i;  
    printf("Número de argumentos (incluido el  
           nombre del programa)= %d \n", argc);  
  
    printf("Argumentos, incluido el programa: \n");  
    for (i=0; i<argc; i++){  
        printf("Argumento[%d] = %s\n", i, argv[i]);  
    }  
  
    return(0);  
}
```



Ejemplo



args_v2.c

- *Imprime por pantalla el nombre del programa y los argumentos*

```
int main(int argc, char ** argv) {  
    int i;  
    printf("Número de argumentos (sin incluir el  
           nombre del programa)= %d \n", argc-1);  
  
    printf("Argumentos,sin incluir el programa:\n");  
    for (i=1;i<argc;i++){  
        printf("Argumento[%d] = %s\n",i,argv[i]);  
    }  
  
    return(0);  
}
```



Consejos de utilización



- Si nuestro programa requiere argumentos debemos
 - ◆ Comprobar que el número de argumentos (***argc***) introducidos por el usuario es correcto.
 - ◆ Comprobar que los valores de los argumentos introducidos por el usuario (***argv***) son correctos.
 - ◆ En caso afirmativo ejecutar la operación que corresponda.
 - ◆ En caso negativo, romper la ejecución del programa y dar al usuario un mensaje indicándole la sintaxis correcta y los valores correctos de los parámetros para llamar al programa.
- Si los argumentos que se pasan son números hay que convertirlos de cadenas a números antes de utilizarlos.
 - ◆ Funciones ***atoi*** y ***atof***



Ejemplos



leeFichero.c

■ *Lectura de un fichero de texto*

```
int main(int argc, char ** argv){
    FILE * f;
    int n;

    //Comprobar que el número de parámetros sea correcto
    if (argc !=2){
        printf("Sintaxis incorrecta: %s <fichero>\n",
            argv[0]);
        exit(-1);
    }

    f = fopen(argv[1],"r");
    if (f==NULL){
        printf("Error al abrir el fichero %s\n",argv[1]);
        exit(-1);
    }
}
```

....



Ejemplos



leeFichero.c

```
....  
    fscanf(f, "%d", &n);  
    while(!feof(f)) {  
        printf("%d\n", n);  
        fscanf(f, "%d", &n);  
    }  
  
    fclose (f);  
  
    return(0);  
}
```



Ejemplos



sumaArgs.c

- *Suma los n números que se le pasan como argumento*

```
int main(int argc, char ** argv){
    int i,nEle,aux;
    int suma;

    //Comprobar que el número de parámetros sea
    //correcto
    if (argc == 1){
        printf("No hay números que sumar\n");
        exit(-1);
    }

    nEle = argc;
    suma = 0;

    . . .
```



Ejemplos



sumaArgs.c

```
....  
    for (i=1; i<nEle; i++){  
        //Convertimos el argumento a número  
        aux = atoi(argv[i]);  
        suma=suma+aux;  
    }  
  
    printf("La suma de los %d elementos es %d\n",  
           nEle, suma);  
  
    return 0;  
  
}
```



Ejemplos



sumaMultiplica.c

- *Suma o multiplica dos números pasados como argumento*

```
int main(int argc, char ** argv){
    float num1, num2, res;
    int opcion;

    //Comprobar que el número de parámetros sea
    //correcto
    if (argc !=4){
        printf("Sintaxis incorrecta: \n");
        printf("%s operacion num1, num2\n",argv[0]);
        printf(" Suma -> operación = 0\n");
        printf(" Producto -> operación = 1\n");
        exit(-1);
    }
    opcion = atoi(argv[1]);
    . . . . .
```

Ejemplos



sumaMultiplica.c

```
//Comprobamos que se haya introducido una opción correcta
```

```
if ((opcion !=0) && (opcion !=1)){  
    printf("Opcion incorrecta: 0(suma) \n  
           1 (producto)\n");  
    exit(-1);  
}
```

```
//Convertimos los números  
num1 = atof(argv[2]);  
num2 = atof(argv[3]);
```

```
if (opcion ==0)  
    res = num1+num2;  
else  
    res = num1*num2;
```



Ejemplos



sumaMultiplica.c

```
printf("El resultado de %s %f y %f es %f\n",  
      opcion==0?"sumar":"multiplicar", num1,  
      num2, res);
```

```
return(0);
```

```
}
```

