FONDAMENTI DI INFORMATICA

Prof. PIER LUCA MONTESSORO
Ing. DAVIDE PIERATTONI

Facoltà di Ingegneria
Università degli Studi di Udine

Linguaggio C e sistema operativo

Nota di Copyright

`argc` e `argv`

- Sono gli argomenti della funzione **`main`**
- Servono per passare dei parametri al programma dalla linea di comando del sistema operativo
- L'intestazione del programma sarà:

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

- **`argv`** (*argument vector*) è un puntatore a un vettore di stringhe di caratteri contenenti gli argomenti nello stesso ordine con cui sono stati scritti sulla linea di comando
- **`argc`** (*argument count*) è il numero di argomenti scritti sulla riga di comando

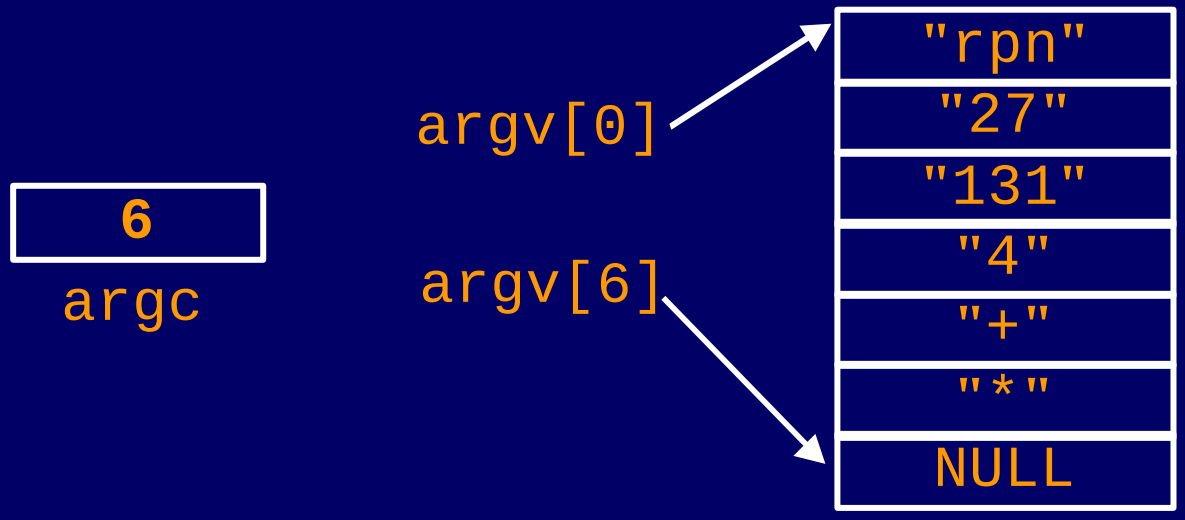
`argc` e `argv`

- Per convenzione, **`argv[0]`** contiene il nome con il quale il programma è stato invocato
- Lo standard impone che **`argv[argc]`** sia un puntatore nullo
- L'ultimo parametro scritto sulla riga di comando è dunque memorizzato in **`argv[argc-1]`**

argc e **argv**: esempio

- Un programma **rpn**, che calcola il valore di un'espressione aritmetica in notazione polacca inversa (*reverse polish notation*), viene lanciato con il seguente comando:

```
C:\PROGRAMMI> rpn 27 131 4 + *
```



argc e argv

```
#include <stdio.h>
#include <stdlib.h>
int main (int argc, char *argv[])
{
   int i;
   for (i = 0; i < argc; i++)
      printf ("argv[%d] = %s\n", i, argv[i]);
   return EXIT_SUCCESS;
}
```

Questo programma stampa nell'ordine i parametri che gli vengono passati dalla riga di comando

Limiti dell'implementazione

- Ogni sistema operativo possiede dei limiti di implementazione per le grandezze manipolabili da un programma
- L'header file **`limits.h`**, presente nella libreria standard, rende noti al programmatore tali limiti, diversi a seconda del sistema operativo
- Lo standard ANSI C impone delle grandezze minime per i valori presenti in **`limits.h`**

Limiti dell'implementazione

Ecco un esempio del contenuto di **limits.h:**

```
CHAR_BIT       8 /* # di bit di un char*/
SCHAR_MAX   +127 /* massimo valore di un signed char */
SCHAR_MIN   -127 /* minimo valore di un signed char */
INT_MAX   +32767 /* se int è memorizzato su 2 byte */
INT_MIN   -32767 /* se int è memorizzato su 2 byte */
FLT_MAX    1E+37 /* massimo valore floating point */
FLT_MIN    1E-37 /* minimo valore floating point */
FLT_DIG       10 /* cifre decimali della precisione */
DBL_EPSILON 1E-9 /* minimo numero x tale che
                    1.0+x != 1.0 */
```

Conclusione di un programma

- Quando un programma termina, il controllo viene restituito al sistema operativo
- Un programma C può comunicare al sistema operativo il risultato della propria esecuzione
- Se si verificano certe condizioni, può essere necessario fermare un programma prima della fine del **`main`**
- La funzione **`void exit(int status)`**, disponibile nella libreria standard **`stdlib.h`**, provoca una terminazione normale del programma, e restituisce al sistema operativo il valore di **`status`**

Codici di errore

- Il file stdlib.h definisce alcuni codici di successo/errore che il programma può restituire al sistema operativo:
  - il valore EXIT_SUCCESS indica una conclusione corretta del programma
  - il valore EXIT_FAILURE segnala invece una conclusione anomala (errore di runtime), la cui interpretazione dipende dal sistema operativo in uso
- La funzione **`void exit(int status)`** impone anche la chiusura dei file aperti (compresi stdin, stdout, stderr) e lo svuotamento dei buffer di lettura/scrittura sui file manipolati dal programma