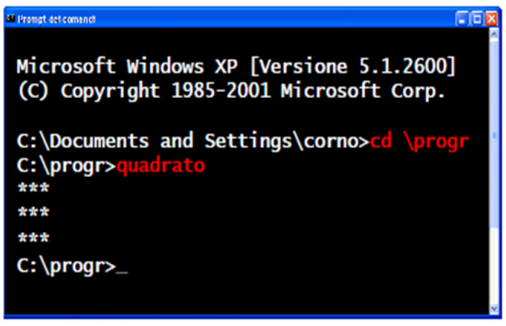
La funzione *main()*

- **La funzione main(), presente in tutti i programmi C, è una funzione come tutte le altre**
- **Unica particolarità: viene chiamata dal Sistema Operativo, appena il programma viene avviato**
  - **Non esiste mai una chiamata a main()**
  - **L'interfaccia della funzione viene definita dalle caratteristiche del sistema operativo**

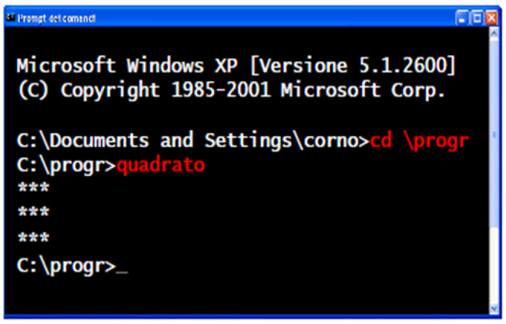
Modalità «consolle»

- **Ricordiamo che il linguaggio C si è evoluto con interfacce «a caratteri»**
- **L'attivazione di un programma avviene digitandone il nome in una finestra di comando**



Modalità «consolle»

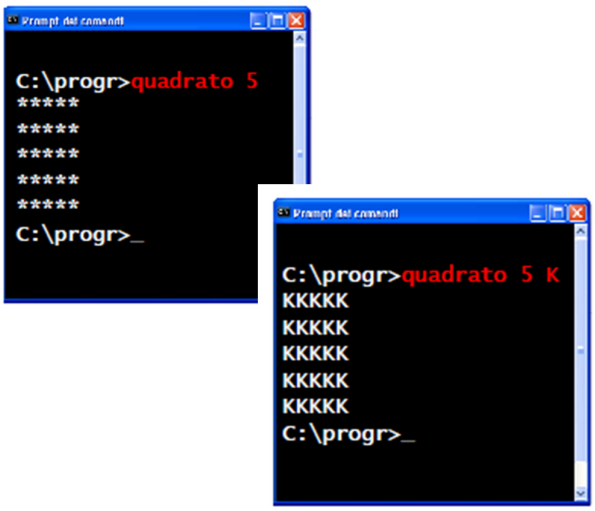
- Ricordiamo che il linguaggio C si è evoluto con interfacce «a caratteri»
- L'attivazione di un programma avviene digitandone il nome in una finestra di comando

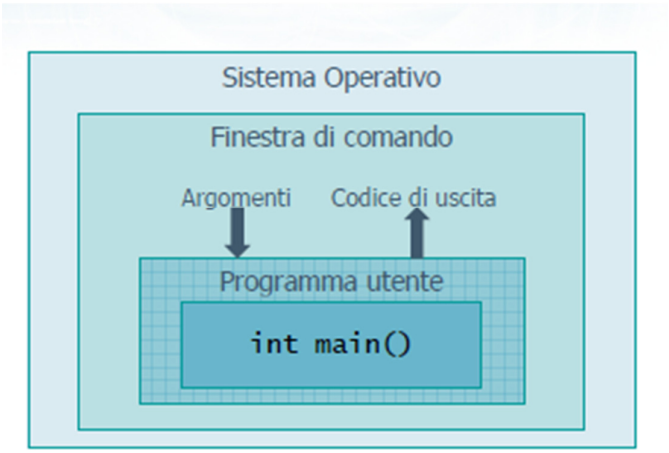


- Al termine dell'esecuzione, il programma restituisce un codice di uscita
  - Un numero intero
  - Indica eventuali condizioni di errore
- Durante l'esecuzione, il programma può accedere a file di stream
  - Default (e se avete incluso «stdio.h»…):
    - L'input è la tastiera
    - L'output è lo schermo

Modalità «consolle»

- La finestra di comando permette di passare al programma una serie di argomenti
  - Nessuna, una o più stringhe di testo
  - Utilizzate dal programma come dati in ingresso

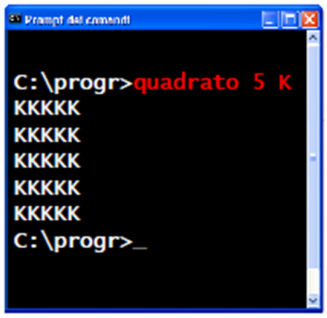




Argomenti del programma

- **Numero variabile di argomenti**
  - **Anche nessuno**
- **Tipo argomenti**
  - **Numeri**
  - **Caratteri o stringhe**

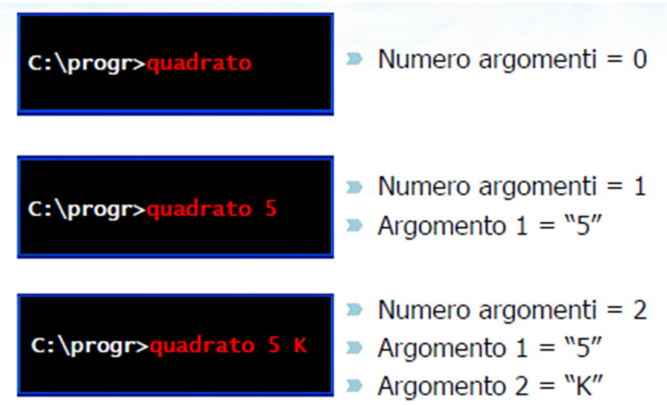
- **Gli argomenti sono anche indicati come**
  - **«argomenti del programma»**
  - **«argomenti del main»**
  - **«parametri del programma»**
  - **«parametri sulla linea di comando»**

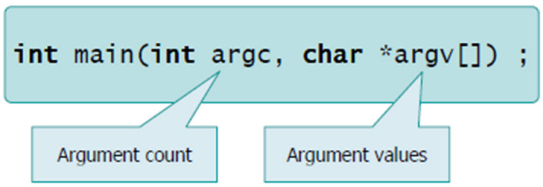


- **Il chiamante (sistema operativo) non ha modo di sapere quanti parametri servono al programma, né di che tipo**
  - **verranno trattati in modo standardizzato**

Argomenti del programma

- **Gli argomenti sulla linea di comando vengono trattati come un vettore di stringhe**
  - **Anche nessuno**
- **Il programma riceve**
  - **Una copia del vettore di stringhe**
  - **Un valore numerico che indica quante stringhe sono presenti**





ARGC, ARGV

- **int argc**
- **Numero di parametri sulla linea di comando**
  - **Incrementato di uno, in quanto il nome del programma viene considerato come un parametro**
    - **Se non vi sono argomenti, vale 1**
    - **Se vi sono k argomenti, vale k+1**

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

Argument count — Argument values

- **char *argv**
- **E' un vettore di stringhe**
  - **Ogni stringa è un parametro del programma**
  - **Vi sono argc diverse stringhe**
    - **La prima stringa (argv[0]) è il nome del programma**
    - **La seconda stringa (argv[1]) è il primo argomento (se esiste)**
    - **…**
    - **L'ultimo argomento è in argv[argc-1]**

ARGC, ARGV

Esempi

```
C:\progr>quadrato
```

- argc==1
- argv[0]=="quadrato"

```
C:\progr>quadrato 5
```

- argc==2
- argv[0]=="quadrato"
- argv[1]=="5"

```
C:\progr>quadrato 5 K
```

- argc==3
- argv[0]=="quadrato"
- argv[1]=="5"
- argv[2]=="K"

ARGC, ARGV

Per capire meglio...

arg.c

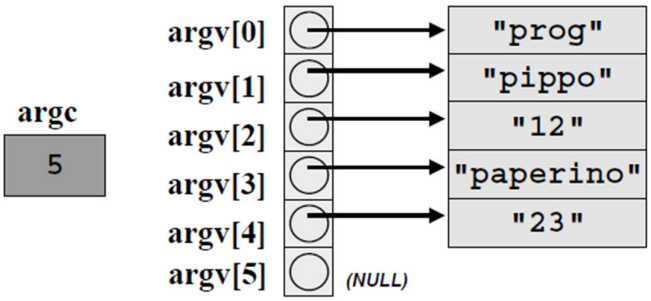
```
#include <stdio.h>

int main(int argc, char *argv[])
{
    int i ;

    printf("argc = %d\n", argc) ;
    for(i=0; i<argc; i++)
    {
        printf("argv[%d] = \"%s\"\n",
            i, argv[i]) ;
    }
}
```

ARGC, ARGV

- **Il vettore argv contiene i dati sotto forma di stringa**
- **Per i dati di tipo numerico (int o double) occorre effettuare la conversione da stringa a numero**
  - **i=atoi(argv[1]);**
  - **r=atof(argv[1]);**

```
C:\progr>progr pippo 12 paperino 23
```

- **Se il parametro è invece una stringa, è possibile**
  - **Utilizzare un puntatore alla stringa di interesse**
  - **Copiarlo in una variabile (all'occorrenza, mediante strcpy)**



Esempio 1

- **Scrivere un programma che dato un carattere e una stringa da linea di comando, conti quante volte il carattere compare nella stringa**
  - **c:> contaOccorrenze *c stringaincuicercare***

- **Algoritmo**
  - **Recuperare gli argomenti**
  - **Scandire la stringa carattere per carattere, contando le occorrenze del carattere dato**
  - **Scrivere il numero di occorrenze a video**

Esempio 1

```
#include <stdio.h>

int main (int argc, char* argv[])
{
  int cont=0, i=0;
  char ch=argv[1][0];
  char *s=argv[2];

  while (s[i] !='\0')
      if (s[i++] = =ch)
          cont++;
  printf ("totale occorrenze: %d", cont);
}
```

Esempio 1 (migliore…)

```
#include <stdlib.h>
int main (int argc, char* argv[])
{ int cont=0, i=0;

  if(argc!=3){
      printf("Numero argomenti non corretto\n");
      exit(0);}
  {char ch=argv[1][0];
    char *s=argv[2];
    while (s[i]!='\0')
            if (s[i++]==ch)
                cont++;
    printf ("totale occorrenze: %d\n", cont);  }
}
```

Esempio 2

- **Scrivere un programma che sommi tutti i numeri (supposti interi) passati come argomenti a linea di comando e stampi a video questo risultato**
  - **c:> sommatutto 12 34 5 67**

- **Algoritmo**
  - **Recuperare gli argomenti**
  - **Trasformare ciascun argomento (stringa) in un intero: si utilizza una funzione di libreria int atoi(char *)**
  - **Scrivere il risultato a video**

Esempio 2

```
#include <stdio.h>
#include <stdlib.h>

int main (int argc, char* argv[])
{
   int sum=0, i=0;

  while (i < (argc-1))
      sum += atoi(argv[++i]);

  printf ("somma totale : %d", sum);
}
```

Argomenti da linea di comando

- **Come passare argomenti dalla linea di comando … quando non c'è una linea di comando, come negli ambienti di sviluppo integrati (esempio lcc)?**
  - **C:> nomeprog pippo 12 pluto 25**
- **Esiste un'apposita opzione da menu**
  - **Project→configuration→debugger (command line arguments)**
- **Oppure si esegue il programma da finestra di comando**

Valore di ritorno

- Al termine dell'elaborazione il programma restituisce un numero intero al sistema operativo
  - Tale valore viene spesso ignorato, ma in caso di esecuzione può essere utilizzato a livello di sistema operativo
- Il valore di ritorno è un int – si preferisce utilizzare «interi positivi piccoli». Convenzionalmente:
  - 0, se tutto è OK
  - !=0, se si è verificata una condizione di errore (il valore specificato può essere utilizzato per specificare la causa dell'errore)
- Il modo più semplice per restituire tale valore è di utilizzare l'istruzione return all'interno della funzione main()
  - L'elaborazione è immediatamente interrotta
  - Il valore ritornato viene passato al sistema operativo

Valore di ritorno

- Esiste inoltre la funzione di libreria exit, dichiarata in «stdlib.h», che assolve alla stessa funzione
  - L'elaborazione è immediatamente interrotta
  - Il valore ritornato viene passato al sistema operativo
- Il vantaggio rispetto all'istruzione return è che può essere usata all'interno di qualsiasi funzione, non solo del main()

```
void exit(int value) ;
```

- Il modo più semplice per restituire tale valore è di utilizzare l'istruzione return all'interno della funzione main()
  - L'elaborazione è immediatamente interrotta
  - Il valore ritornato viene passato al sistema operativo

Esercizio 3

- **Si scriva un programma da utilizzarsi come semplice calcolatrice sulla linea di comando**
- **Il programma, denominato «calcola», accetta 3 parametri sulla linea di comando**
  - **Il primo ed il terzo parametro sono degli operandi, espressi come numeri reali**
  - **Il secondo parametro è un operatore, scelto tra +, -, * e /**
- **Il programma stampa il risultato corrispondente all'operazione**

```
c:\progr>calcola 3 + 2
Risultato: 5.0000
c:\progr>calcola 3 * 2
Risultato: 6.0000
c:\progr>calcola 3 $ 2
Errore: operatore non riconosciuto
c:\progr>calcola 3 +
Errore: operando mancante
```