Che cosa serve per cominciare?

- La linea di comando
- Il notepad (o editor equivalente)
- ***Saper scrivere "qualcosa" a video*** da programma – risultato dell'elaborazione

Ma come si fa?!

Input / Output

- L'immissione dei dati di un programma e l'uscita dei suoi risultati avvengono attraverso operazioni di ***lettura e scrittura***
- C ***non ha istruzioni predefinite*** a livello di linguaggio per l'input/output
- In ogni versione ANSI C, esiste una ***Libreria Standard*** (**stdio**) che mette a disposizione alcune funzioni (dette *funzioni di libreria*) per effettuare l'input e l'output

Input / Output

- Le dichiarazioni delle funzioni messe a disposizione da tale libreria devono essere incluse nel programma: **`#include <stdio.h>`**
  - **`#include`** è una direttiva per il **preprocessore** C
  - nella fase che ***precede la compilazione*** del programma ogni direttiva "#..." viene eseguita, provocando delle ***modifiche testuali*** sul programma sorgente. Nel caso di **`#include <nomefile>`** viene sostituita l'istruzione stessa con il contenuto del file specificato
- **Dispositivi standard di input e di output:**
  - per ogni macchina, sono periferiche predefinite (generalmente tastiera per l'input e video per l'output)

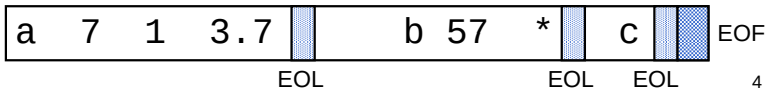
Input / Output

A default, la macchina runtime del C vede le informazioni lette/scritte da/verso i dispositivi standard di I/O come file *sequenziali*, cioè **sequenze di caratteri** (o stream). Vedremo più avanti la possibilità di fare anche I/O in cosiddetto formato binario...

- *stream* di input/output ***possono*** contenere dei caratteri di controllo:
  - End Of File (EOF)
  - End Of Line (EOL)

Sono disponibili funzioni di libreria per:

- ***Input/Output a caratteri***
- ***Input/Output a stringhe di caratteri***
- ***Input/Output con formato***



Input / Output con Formato

- Nell'I/O con formato occorre specificare il ***formato*** (*tipo*) dei dati che si vogliono leggere oppure stampare
- Il ***formato*** stabilisce:
  - **come interpretare** la sequenza dei caratteri immessi dal dispositivo di ingresso (nel caso della lettura)
  - con quale sequenza di caratteri **rappresentare** in uscita i valori da stampare (nel caso di scrittura)

Lettura con Formato: `scanf`

È una **particolare forma di assegnamento**: scanf() assegna i valori letti alle variabili specificate come argomenti (nell'ordine di lettura)

```
scanf(<stringa-formato>, <sequenza-
                variabili>);
```

Ad esempio:

```
int X;
float Y;
scanf("%d%f", &X, &Y);
```


Lettura con Formato: `scanf`

`scanf()` legge una serie di valori ***in base alle specifiche contenute in <stringa-formato>*** e memorizza i valori letti nelle variabili elencate

- restituisce il ***numero di valori letti*** e memorizzati, oppure EOF
- gli ***identificatori*** delle variabili a cui assegnare i valori sono ***sempre*** preceduti dal ***simbolo &*** (ne parleremo diffusamente…)
- la <stringa_formato> può contenere dei caratteri qualsiasi (scartati durante la lettura), che si prevede vengano immessi dall'esterno, insieme ai dati da leggere

```
scanf("%d:%d:%d", &A, &B, &C);
```

richiede che i tre dati da leggere vengano immessi separati dal carattere ":"

Scrittura con Formato: `printf`

- **`printf()`** viene utilizzata per scrivere su standard output il valore di una variabile o, più in generale, il risultato di una espressione
- Anche in scrittura è necessario specificare (mediante una ***stringa di formato***) il formato dei dati che si vogliono stampare

```
printf(<stringa-formato>,<sequenza-elementi>)
```

- **`printf`** restituisce il ***numero di caratteri scritti***
- La stringa di formato della **`printf()`** può contenere sequenze costanti di caratteri da visualizzare

Formati Comuni

- ***Formati più comuni***

```
int                 %d
float               %f
carattere singolo   %c
stringa di caratteri %s
```

- ***Caratteri di controllo***

```
newline             \n
tab                 \t
backspace           \b
form feed           \f
carriage return     \r
```

- Per la stampa del carattere '**%**' si usa: **%%**

Esempio

```
int main()
{
printf("Prime sei potenze di 2:  %d, %d, %d,
  %d, %d, %d", 1, 2, 4, 8, 16, 32);
}
```


getchar() e putchar()

- **`char getchar();`**
  - ***Legge un carattere da standard input e lo restituisce***
  - Prima di cominciare a leggere, attende la pressione del tasto invio
  - La lettura termina quando viene restituito il carattere di fine linea (10)
- **`void putchar(char c);`**
  - ***Scrive un carattere su standard output***

Un sano esercizio

- Tentare di costruire le funzioni di lettura/scrittura (**`scanf`**, **`printf`**) partendo dalle funzioni base di lettura caratteri:
  - **`char getchar();`**
  - **`void putchar(char c);`**
- Quali problemi (strutture dati ancora non esaminate o funzionalità necessarie) incontrate al momento?