# Laboratorio di algoritmi e strutture dati
# Primi programmi - esercizi da svolgere a casa

Docente: Violetta Lonati

Giovedì 4 ottobre 2018

Nota: ricordate le opzioni principali del comando `gcc` (per eventuali dubbi, consultate il manuale on-line `man gcc`):

- `-o` per salvare l'*output* in un file specificato,
- `-c` per arrestare la compilazione prima della fase di *link*,
- `-s` per arrestare la compilazione prima della fase di *assemble*,
- `-E` per arrestare la compilazione dopo la fase di *preprocessing*,
- `-W`, con parametro `all`, per la segnalazione di avvertimenti (warning),
- `-l`, con parametro `m`, per linkare le librerie matematiche.

Utilizzate un editor (ad esempio `gedit`) per creare i file sorgenti secondo quanto indicato dagli esercizi seguenti.

## 1 Conversione temperatura

Scrivete un programma che legga una temperatura in gradi Farenheit e stampi l'equivalente in gradi centigradi.

## 2 Equazione di secondo grado

Scrivete un programma che legga tre coefficienti $a, b, c$ e calcoli le soluzioni (complesse) dell'equazione $ax^2 + bx + c$. Può essere utile includere il file di intestazione `math.h` della libreria standard, contenente la funzione `sqrt` per il calcolo della radice quadrata.

## 3 Alfabeto italiano

Scrivete un programma che legga un carattere e stabilisca se si tratta di una vocale dell'alfabeto italiano, di una consonante dell'alfabeto italiano, o un carattere che non fa parte dell'alfabeto italiano. Per la lettura di un carattere, potete usare l'istruzione `scanf("%c", &c);` dove `c` è una variabile di tipo `char`. Può essere utile inoltre includere il file di intestazione `ctype.h` della libreria standard, contenente alcune funzioni per l'analisi e l'elaborazione di caratteri, quali ad esempio `tolower`.

## 4 Classificazione triangoli

Scrivete un programma che, dati tre numeri, stabilisce se questi possono definire le lunghezze dei lati di un triangolo e, in caso affermativo, classifica il triangolo come equilatero, isoscele o scaleno. Ricordate questa proprietà fondamentale: in un triangolo, ogni lato è più corto della somma degli altri due.

## 5 Cerchio

Scrivete un programma che legga (in una variabile di tipo `float`) il raggio di un cerchio e ne stampi l'area.

Tenete presente che il file `math.h` della libreria standard contiene la definizione della costante `M_PI`, pari al valore di $\pi$ (il rapporto tra la circonferenza ed il diametro). Quando compilate, usate l'opzione `-lm`.

## 6 Quadrati perfetti

Scrivete un programma che stampi la sequenza crescente dei primi 10 quadrati perfetti (ossia numeri $x$ tali che $x = y^2$ per qualche numero naturale $y$). Usate una macro di valore 10 per rendere il programma più facile da modificare!

## 7 Massimo tra due numeri

Scrivete un programma che legga due numeri interi e stabilisca qual è il massimo, usando l'operatore condizionale `expr1 ? expr2 : expr3`.

## 8 Ordine alfabetico

Scrivete un programma che legga due lettere maiuscole e stampi la loro distanza nell'ordine alfabetico. Ad esempio, su ingresso **A C**, il programma deve stampare 3, su ingresso **F B**, il programma deve stampare 5. Ricordate che i caratteri (`char`) in C sono rappresentati come (piccoli) interi e osservate che, secondo la codifica ASCII, non c'è soluzione di continuità e non ci sono altri caratteri tra la A e la Z (cosa di cui potete convincervi con il comando `man ascii`).

## 9 Operatore sizeof e limits.h

L'operatore unario `sizeof` applicato ad una variabile ha per valore la dimensione della variabile a cui è applicato espressa in byte. Così, ad esempio (sull'architettura delle macchine presenti in laboratorio), se la variabile `x` è di tipo `int`, allora l'espressione `sizeof(x)` ha valore 4.

Il file di intestazione `limits.h` contiene (tra le altre cose) le definizioni (ottenute grazie alla direttiva `#define`) dei valori limite, per l'architettura corrente, dei tipi interi e carattere. Ad esempio, il seguente programma stampa l'intervallo di valori possibili per le variabili di tipo intero con segno.

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

int main( void ) {
    printf( "%d..%d\n", INT_MIN, INT_MAX );

    return 0;
}
```

Le definizioni dei limiti per alcuni degli altri tipi fondamentali si chiamano: `SCHAR_MIN`, `SCHAR_MAX` e `UCHAR_MAX` per il tipo carattere con e senza segno, `SHRT_MIN`, `SHRT_MAX` e `USHRT_MAX` per il tipo intero breve, con e senza segno.

Scrivete un programma che dichiari una variabile per ciascuno dei tipi fondamentali e delle sue rispettive varianti `long` e `short` (qualora ammissibili), e ne stampi la dimensione in byte ottenuta tramite l'operatore `sizeof` e l'intervallo di valori possibili per tali tipi.

## 10 Divisori e numeri primi

1. Scrivete un programma che stampi la sequenza decrescente dei numeri divisori di *n*, dove *n* è un numero inserito in input. Modificate il programma in modo che stampi anche il numero di divisori di *n*.

2. Scrivete un altro programma che, usando un ciclo `for`, stabilisca se un numero *n* è primo (ovvero ha per divisori solo 1 e se stesso) oppure no. Cercate di ridurre il numero di istruzioni da eseguire! Scrivete una nuova versione del programma precedente usando un ciclo `while`.

3. Scrivete un programma che scomponga in fattori primi un numero dato in input.

## 11 Intervallo di tempo

Scrivete un programma che calcoli l'intervallo di tempo compreso tra due orari. Assumete che sia gli orari che l'intervallo di tempo siano rappresentati nel formato a 24 ore `hh:mm:ss`. E' possibile usare short int invece che int?

## 12 Conversione orario

Scrivete un programma che, dato un orario in formato a 24 ore `hh:mm`, fornisca il corrispondente orario nel formato AM/PM e viceversa. Ricordate che secondo il formato AM/PM, le 24 ore del giorno sono suddivise in due periodi chiamati AM (ante meridiem) e PM (post meridiem): ogni periodo consiste di 12 ore numerate con 12 (usato come 0), 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11. L'orario `12:00 AM` indica la mezzanotte, l'orario `12:00 PM` indica il mezzogiorno.