Laboratorio 1

Esercizio 0

1. Creare un file di nome **pr0.cpp** con un editore di testo. **Utilizzare l'estensione .cpp è fondamentale** (altrimenti l'editore e il compilatore non sa cosa farne).
2. Copiare nel file **pr0.cpp** le seguenti righe:

```
// Program prints a simple message
#include <iostream>

// function main begins program execution
int main()
{
  std::cout << "Welcome to C++!" << std::endl;   // prints on screen
  return 0;
} // end function main
```

3. Aprire un terminale e posizionarsi nella cartella dove si trova **pr0.cpp**. Compilare (trasformare in un eseguibile **pr0.cpp**) utilizzando il commando **g++ pr0.cpp**. Lanciare l'eseguibile con **./a.out**.

Per ogni esercizio da qua in avanti:

Individuare le variabili necessarie (e il loro tipo!). **Commentare** il codice per facilitare la comprensione dello stesso.

Esercizio 1

Scrivere un programma che legga dalla tastiera due numeri interi (**int**) con **std::cin** e stampi sul schermo (**std::cout**) il loro prodotto.

Esercizio 2

Scrivere un programma che, dati due interi dall'utente tramite **std::cin**, stabilisca se il primo numero è multiplo del secondo. Utilizzare l'operatore **%**, chiamato operatore *modulo*, che restituisce il resto di una divisione. Per valutare se due quantità sono uguale si usa l'operatore **==.** Per esempio, **a==5** è vero (true) se la variabile **a** è uguale a 5. (Un singolo = effettua invece un'assegnazione.) Prima di scriverlo disegnare il diagramma di flusso del programma.

Esercizio 3

Scrivere un programma che legga dalla tastiera un numero intero, *n,* con **std::cin** e stampi sul schermo (**std::cout**) *n* asterischi in una riga utilizzando un ciclo **while.** Per esempio, se l'utente immette 6 allora il programma deve stampare

******

Attenzione: ogni carattere * deve essere stampata singolarmente con il **std::cout << "*";**.

Esercizio 4

A. Scrivere un programma che, utilizzando cicli annidati (un **while** dentro un altro **while**), stampi sullo schermo il seguente "quadrato" di stelle (come prima, ogni carattere * deve essere stampata singolarmente con il **std::cout << "*";**):

```
*****
*****
*****
*****
*****
```

Il numero di caratteri * nella prima riga deve essere letto dal **std::cin** all'inizio dell'esecuzione del programma.
B. Realizzare una seconda versione che stampi

```
*
**
***
****
*****
```

C. E una terza che stampi

```
*****
 ****
  ***
   **
    *
```

Esercizio 5

Scrivere un programma che calcoli il fattoriale di un numero intero, **int,** dato dall'utente, con un ciclo **while**. Testare fino a quale numero il valore stampato è corretto. Ripetere il test utilizzando variabili **short, int, long** e **long long**. Tutti e quattro tipi servono per rappresentare interi ma la loro portata è diversa perché occupano numeri diversi di byte.

Esercizio 6

Scrivere un programma che legga dal **std::cin** 10 numeri reali (il tipo da usare per numeri reali è il **double**) e stabilisca qual era il numero massimo e il numero minimo nella seria. Il programma deve essere realizzato senza utilizzare vettori (array) e non deve utilizzare più di quattro variabili. Prima di scriverlo disegnare il diagramma di flusso del programma.

Esercizio 7

Una approssimazione di π di "grado n" può essere calcolata tramite la somma

$$\pi_n = \sum_{j=0}^{n} (-1)^j \frac{4}{2j+1}$$

1. Sviluppare un programma che prenda un numero intero, n, e calcoli l'approssimazione di "grado n".
2. Sviluppare una seconda versione che prenda un double, ε, e calcoli una approssimazione di π di "grado n" tale che $|\pi_n - \pi_{n-1}| < \varepsilon$ (quindi il valore assoluto dell'ultimo termine della sommatoria deve essere minore di ε).
3. Testare i programmi. Con quale precisione si riesce a calcolare π? Provare a utilizzare diversi tipi di numeri reali (float, double, long double).

(Le prime 50 cifre di π sono 3.14159 26535 89793 23846 26433 83279 50288 41971 69399 3751.)