mlampis.dev

C++

Manuale per le ripetizioni di Informatica

Marco Lampis

1 dicembre 2022

Indice

0 Informazioni generali

Ciao! Sono **Marco**, sono uno studente magistrale in *Software Engineering* al Politecnico di Torino e mi sono laureato in Ingegneria Informatica all'università di Pisa. Nella vita sono un programmatore, uno smanettone e amante di videogiochi! Amo quello che studio, e per questo motivo fornisco ripetizioni di informatica con particolare attenzione a:

- programmazione (Java, C++, C, Python, C#, Javascript, PHP)
- algoritmi e strutture dati
- basi di Dati
- più o meno tutto quello che riguarda l'informatica!

Sia per studenti delle scuole superiori che per l'universita.

0.1 Come svolgo le lezioni

La parola d'ordine alla base delle lezioni è "innovazione"! Potete dire addio alle video-lezioni in cui i docenti condividono lo schermo comportando un inutile spreco di tempo e difficoltà di apprendimento, l'approccio utilizzato è al passo con i tempi con un focus verso l'interazione "docente-studente". Questo viene raggiunto mediante l'utilizzo di piattaforme ad hoc che consentono l'interazione diretta e la programmazione contemporanea tra più persone, senza la necessità di condividere lo schermo! (Quindi è come lavorare sullo stesso computer) Ogni studente viene seguito in un percorso formativo su misura pensato in base alle necessità e agli obbiettivi stabiliti. Per aiutarlo in questo percorso metto a disposizione un sito web creato appositamente per il corso e sempre disponibile che mi sono occupato di realizzare personalmente, con al suo interno una vasta gamma di contenuti utili come: esercizi (con soluzioni), slide, approfondimenti e molto altro!

Le lezioni fanno riferimento sia a esercitazioni che ad approfondimenti teorici, includendo aiuto compiti e revisione. Le lezioni pratiche vedono l'utilizzo dei portali menzionati sopra, mentre per le lezioni teoriche utilizzo un Ipad per prendere appunti per gli studenti che poi rilascio a fine lezione, oltre a slide apposite (ancora in corso di stesura).

Il tutto è accompagnato da una vasta raccolta di materiale che condivido con gli studenti e che aggiorno lezione dopo lezione (e rilascio); tutto il materiale svolto viene dunque pubblicato su in sito

apposito (che non mi è concesso menzionare) ed è sempre consultabile dagli studenti, anche dopo la lezione (gratuitamente).

1 if

Il costrutto `if` consente di verificare se una condizione è verificata ed eseguire un blocco di codice in base al risultato. L'esito è sempre di tipo **booleano**, ovvero può essere solo `true` o `false`.

1.1 Sintassi

La sintassi di `if` è la seguente:

```
if (condizione1){
    // codice eseguito se la condizione é vera
}

else if(condizione2){
    // codice eseguito se la condizione1 non é vera
    // ma é vera condizione2
}

else{
    // codice eseguito se nessuna delle condizioni é vera
}
```

Attenzione: Il blocco di codice che segue `else if` ed `else` è opzionale. Se non viene specificato nessun blocco di codice, il programma non esegue alcuna operazione. Inoltre è importante scrivere le keyword con caratteri minuscoli, in caso contrario si avrà errore

Suggerimento: Non utilizzare `else if` ed `else` se non è necessario, in quanto possono rendere il codice meno leggibile.

1.1.1 Esempio

```
int a = 5;
int b = 6;

```

```
4  if (b > a){
5      cout<<"b é maggiore di a";
6  }
7
8  else if (a == b){
9      cout<<"a e b sono uguali";
10 }
11
12 else{
13     cout<<"a é maggiore di b";
14 }
```

Puoi trovare a questo link alcuni esercizi su **if**: https://esercizi.mlampis.dev/book.php?langs=cpp

2 For

Il ciclo `for` viene utilizzato quando si conosce il numero di iterazioni da compiere. Il ciclo `for` è composto da tre parti:

- **inizializzazione**: viene inizializzato un contatore
- **verifica**: viene verificata la condizione di uscita dal ciclo
- **passo**: viene incrementato il contatore

2.1 Sintassi

La sintassi di `for` è la seguente:

```
for (inizializzazione; verifica; passo){
    // codice eseguito
}
```

> **Nota**: spesso si utilizza il contatore `i` per indicare il numero di iterazioni.

2.1.1 Esempio

Esempio di esercizio che stampa i numeri da 1 a 5 (escluso):

```
for (int i = 0; i < 5; i++) {
  cout << i << "\n";
}
```

2.2 Cicli annidati

E' possibile inserire un `for` all'interno di un'altro `for` (senza limiti di volte), ottenendo un ciclo annidato. In questo caso, il ciclo esterno viene eseguito prima del ciclo interno.

2.2.1 Esempio

```cpp
// ciclo esterno
for (int i = 1; i <= 2; ++i) {
  cout << "Outer: " << i << "\n"; // Eseguito 2 volte

  // Inner loop
  for (int j = 1; j <= 3; ++j) {
    cout << " Inner: " << j << "\n"; // Eseguito 6 volte (2 * 3)
  }
}
```

2.3 Esercitazione

Dato una numero mostrato a schermo, mostrarne la tabellina.

esempio di output:

```
Quale tabellina vorrestri mostrare? 5
5 * 1   =  5
5 * 2   =  10
5 * 3   =  15
5 * 4   =  20
5 * 5   =  25
5 * 6   =  30
5 * 7   =  35
5 * 8   =  40
5 * 9   =  45
5 * 10  =  50
```

```cpp
#include <iostream>
using namespace std;

int main(){

    // creiamo la variabile per salvare la tabellina
    int numero;

    cout<<"Quale tabellina vorrestri mostrare? ";

    // facciamo inserire all'utente il numero
    cin>>numero;

    // vogliamo fare una azione ripetuta
    // partena -> 1
    // arrivo  -> 10 (incluso)
    for(int cont=1; cont <= 10; cont = cont+1){

```

```
19          // stiamo mostrando qualcosa a schermo:
20          // numero " * " cont " = " prodotto
21          int prodotto= numero*cont;
22
23          // 5 * 1   =  5
24          cout<<numero<<" * "<<cont<< " = "<<prodotto<<endl;
25      }
26
27      return 0;
28  }
```

3 Strutture (struct)

Le strutture, solitamente chiamate `struct`, consentono il raggruppamento di più informazioni all'interno di un solo contenitore.

> Puoi pensare a una struct come a una scatola con scompartimenti, in cui ogni scompartimento può contenere a sua volta altri oggetti.

3.1 Sintassi

Per realizzare una struttura è sufficiente utilizzare la keyword `struct` seguita dalle `{}`; al loro interno andremo a mettere le variabili che faranno parte della nostra struttura.

```
struct {             // dichiarazione della struttura
    int  contenuto1; // membro (variabile int)
    char contenuto2; // membro (variabile char)
    // ...
} nomeStruttura;     // nome con cui verrà richiamata
```

3.2 Accedere ai membri

Per accedere a una variabile membro della struttura si utilizza il nome della variabile seguita dal carattere `.`.

```
// dichiarazione della struttura
struct {
    int  contenuto1; // membro (variabile int)
    char contenuto2; // membro (variabile char)
    // ...
} nomeStruttura;     // nome con cui verrà richiamata

// istanzio una struttura
nomeStruttura st1;

// ne assegno i valori
```

```
st1.contenuto1 = 1;
st1.contenuto2 = 'A';
```

3.3 Esercitazione

Costruire un record **Prodotto** con:

- nome
- marca
- prezzo unitario
- data scadenza
- quantità in magazzino

Fare l'input e l'output di un prodotto, stampare poi il valore totale del prodotto accantonato in magazzino.

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

// definiamo il valore massimo di caratteri in una stringa
#define size 20

// creazione della struttura per l'inserimento della data
struct data{
    int giorno;
    int mese;
    int anno;
};

// creazione della struttura per il prodotto
struct prodotto{
  char nome[size];
  char marca[size];
  float prezzo_unitario;
  struct data scadenza;
  int quantita;
};


int main(){

  // creaimo una variabile per salvare
  // le informazioni del prodotto
  struct prodotto pippo;

  // chiediamo di inserire il nome
```

```
31    printf("Inserire il nome del prodotto: ");
32    scanf("%s", pippo.nome);
33
34    // chiediamo di inserire la marca
35    printf("Inserire la marca del prodotto: ");
36    scanf("%s", pippo.marca);
37
38    // chiediamo di inserire il prezzo unitario
39    printf("Inserire il prezzo unitario: ");
40    scanf("%f", &pippo.prezzo_unitario);
41
42    // chiediamo la scadenza del prodotto
43    printf("Inserire il giorno di scadenza: ");
44    scanf("%d", &pippo.scadenza.giorno);
45
46    printf("Inserire il mese di scadenza: ");
47    scanf("%d", &pippo.scadenza.mese);
48
49    printf("Inserire l'anno di scadenza: ");
50    scanf("%d", &pippo.scadenza.anno);
51
52    // inseriamo la quantità dei prodotti
53    printf("Inserire la quantità di prodotto: ");
54    scanf("%d", &pippo.quantita);
55
56
57    printf("Il prodotto %s ha le seguenti caratteristiche: \n", pippo.
          nome);
58    printf("- nome: %s\n", pippo.nome);
59    printf("- marca: %s\n", pippo.marca);
60    printf("- prezzo unitario: %f\n",pippo.prezzo_unitario);
61    printf("- scadenza: %d-%d-%d\n", pippo.scadenza.anno, pippo.scadenza.
          mese, pippo.scadenza.giorno);
62    printf("- quantita': %d\n", pippo.quantita);
63
64    float valore_totale;
65    valore_totale = pippo.prezzo_unitario* pippo.quantita;
66
67    printf("Il valore totale e' di %.2f euro", valore_totale);
68
69    return 0;
70 }
```