

Informazioni Generali

Marco Alberti



**Dipartimento
di Matematica
e Informatica**



**Università
degli Studi
di Ferrara**

Programmazione e Laboratorio, A.A. 2024-2025

Ultima modifica: 6 dicembre 2023

Attenzione! Questo materiale didattico è per uso personale dello studente ed è coperto da copyright.
Ne sono vietati la riproduzione e il riutilizzo anche parziale, ai sensi e per gli effetti della legge sul diritto d'autore.

Sommario

1 Argomento del corso: Programmazione

2 Informazioni pratiche

Sommario

1 Argomento del corso: Programmazione

2 Informazioni pratiche

Informazione

I **dati** sono informazione codificata per poter essere elaborata e scambiata.

L'**informazione** è la risposta (parziale) a una domanda; l'informazione riduce l'incertezza di chi la riceve.

L'informazione si misura in **bit** (binary digit) o multipli. Un bit è la quantità di informazione sufficiente a dimezzare l'incertezza (e ad annullarla se le scelte sono solo due).

Convenzionalmente i possibili valori di un bit sono 0 e 1.

Esempi

- *Quale* direzione prendere a un bivio (1 bit)
- *Quale* fra i numeri compresi fra 0 e 255 (8 bit = 1 byte)
- *Quale* fotografia (1 milione di byte)
- *Quale* contabilità annuale di un'azienda

Computer e informazione

I computer sono utili perché, **programmati**, aiutano a risolvere problemi di carenza di informazione.

Problema

Qual è la somma di due numeri?

Il computer non crea informazione che non esiste; la rende immediatamente disponibile all'utente, elaborando quella fornita.

Problema

Che tempo fa adesso a Tokyo?

Programmazione

Il corso ha l'obiettivo di introdurre le basi della programmazione dei calcolatori elettronici (computer) utilizzando il linguaggio di programmazione C.

- **Computer**: macchina in grado di
 - immagazzinare dati
 - elaborare dati
 - comunicare dati con l'esterno per mezzo di dispositivi di input (ingresso) e output (uscita)
- Le modalità di elaborazione e la comunicazione sono specificate da un **programma**, cioè una rappresentazione
 - delle informazioni gestite (**strutture dati**)
 - delle operazioni da eseguire (**algoritmo**)in un **linguaggio di programmazione**.
- La **programmazione** è l'attività di produzione dei programmi.

Un programma (in linguaggio C)

001_informazioni_generali/somma.c

```
1  #include <stdio.h>
2
3  int main() {
4      int a, b, c;
5      printf("Immetti due numeri interi:\n");
6      scanf("%d%d", &a, &b);
7      c = a + b;
8      printf("La somma di %d e %d e` %d\n", a, b, c);
9      return 0;
10 }
```

Quale problema risolve?

Contenuti del corso

- Concetti alla base della risoluzione di problemi per mezzo di calcolatore elettronico (hardware e software, macchina di Von Neumann e macchine astratte, input e output, algoritmo, programma, processo)
- Paradigma procedurale/imperativo di programmazione: espressioni e istruzioni, controllo di flusso, astrazione procedurale
- Tipi di dato (rappresentazione dell'informazione) primitivi, composti e astratti
- Gestione della memoria
- Ricorsione
- Linguaggio di programmazione C

Capacità da acquisire

- Identificazione delle strutture dati (rappresentazione dell'informazione) e degli algoritmi (procedimenti di calcolo) adatti alla risoluzione di problemi per mezzo di un calcolatore
- Implementazione in linguaggio C di algoritmi e strutture dati (in particolare, operazioni su *sequenze* di dati primitivi, composti e astratti)
- Strutturazione modulare di un programma
- Utilizzo dei principali strumenti per la programmazione

Le nozioni saranno presentate in modo da favorire l'acquisizione progressiva di queste capacità, per problemi di complessità crescente.

Sommario

1 Argomento del corso: Programmazione

2 Informazioni pratiche

Attività didattiche

- Lezioni in presenza: esposizione argomenti di teoria con esempi, demo e semplici esercizi. Saranno rese disponibili le registrazioni di anni precedenti
- Laboratorio (in presenza): svolgimento guidato di esercizi più complessi
- Pre-corso: per chi ha difficoltà a seguire le prime lezioni. Già terminato, lezioni disponibili in Classroom
- Tutorato (dott.ssa Elisabetta Gentili, elisabetta.gentili1@unife.it): svolgimento di esercizi, con l'assistenza di un tutor, disponibile anche per chiarimenti

Lezioni ed esercitazioni: di regola

- Martedì, 10.30 – 13.30: lezione
- Mercoledì, 10.30 – 13.30: lezione
- Giovedì, 13.30 – 16.30: laboratorio

Tutorato:

- Martedì, 14.00 – 16.00 (dal 1 ottobre 2024)

Dove: Aula F9, Chiostro di Santa Maria delle Grazie, Via Fossato di Mortara, 15

Ricevimento (via Saragat 1, Blocco A, ufficio 038):

- su appuntamento

- Prova teorica (max 11 punti, soglia 6): domande e semplici esercizi su tutti gli argomenti del corso (materiale didattico non consentito)
- Prova pratica (max 22 punti, soglia 12): scrittura di un programma in linguaggio C che implementi la specifica assegnata, valutato per identificazione e corretta implementazione di strutture dati e degli algoritmi appropriati alle specifiche, utilizzo efficiente delle risorse, stile (chiarezza, utilizzo di costrutti appropriati, corretta strutturazione)
- E' necessario sostenere le prove nello stesso appello
- L'esame si supera se
 - entrambe le prove sono sufficienti; il voto è pari alla somma dei punteggi (30 se la somma è 30 o 31, 30 e lode se la somma è 32 o 33), oppure
 - una delle prove è insufficiente ma la somma dei punteggi è maggiore o uguale a 18 e si supera una prova orale/pratica integrativa; il voto è uguale alla somma delle prove pratica e teorica

Prove pratiche parziali

E' possibile suddividere la prova pratica in due prove parziali:

1a 4 novembre 2024;

2a 19 dicembre 2024.

Ogni prova parziale ha un punteggio massimo di 11 punti. La prova pratica è superata si ottengono almeno 12 punti in totale e almeno 5 in ognuna delle prove parziali; il punteggio è la somma dei due punteggi parziali.

Una volta superata la prova pratica, si può sostenere quella teorica in qualsiasi appello dello stesso anno accademico (fino a settembre 2023); il voto d'esame è determinato come nella slide 10. Se non si supera la prova teorica, occorre ripetere anche quella pratica.

Materiale didattico

- Diapositive usate a lezione, disponibili in Classroom
- Esempi di codice su GitHub
- Manuali degli strumenti
- Testi didattici (disponibili in biblioteca, non obbligatori; vanno benissimo edizioni precedenti):
 - P.Deitel, H.Deitel - Il linguaggio C. Fondamenti e tecniche di programmazione (Nona edizione) - Pearson
 - A.Bellini, A.Guidi - Linguaggio C (Quinta edizione) - Mc Graw-Hill
- Testi di riferimento:
 - B.W.Kernighan, D.R.Ritchie - Il linguaggio C. Principi di programmazione e manuale di riferimento (Seconda edizione) - Pearson
 - A.Kelley, I.Pohl - C Didattica e Programmazione (Seconda edizione) - Pearson

Diapositive

- Saranno sempre disponibili il lunedì mattina precedente la lezione, in modo che si possano stampare e annotare a lezione
- Saranno via via aggiornate con la versione annotata dal docente
- I titoli degli esempi di codice inseriti nelle diapositive sono link agli esempi pubblicati (il testo del link indica dove trovarli se la diapositiva è stampata). Esempio alla slide 5.

Come si impara a programmare?

Saper programmare richiede

- Capacità di analisi: dato un programma, capire cosa fa (ed eventualmente perché non fa quel che si vuole), per mezzo di un *modello mentale* della macchina, che si costruisce via via con
 - Lezioni
 - Materiale didattico
 - Testi
 - Esercizi: i computer funzionano in modo non intuitivo e a volte ci sorprendono costringendoci ad aggiornare il nostro modello mentale
- Capacità di sintesi: creare un programma che faccia quel che si vuole, combinando le nozioni e le tecniche apprese. Si acquisisce
 - Programmando

Come preparare l'esame

- Nessun prerequisito
- L'esame vale 12 crediti, corrispondenti a 300 ore di cui
 - 96 di attività didattica frontale (lezioni, esercitazioni)
 - 24 di tutorato
 - 180 di studio individuale: da a ora a gennaio 12 settimane effettive, quindi circa 15 ore a settimana, cioè 3 ore al giorno lavorativo.
- Come impiegare le 3 ore? E' necessario svolgere esercizi (al calcolatore e con carta e penna) per imparare a programmare, ma anche per capire la teoria!
Suggerimento:
 - 1 ora di studio della teoria
 - 2 ore di esercizi
- Cominciando subito: se non si fanno gli esercizi non si capiranno le nozioni di base e sarà difficile seguire gli argomenti avanzati, che saranno trattati più velocemente

Fonti di informazioni

- Scheda ufficiale dell'insegnamento (*syllabus*): <https://unife.coursecatalogue.cineca.it/insegnamenti/2024/45914/2016/9999/10431>
- Il materiale didattico è distribuito attraverso la piattaforma Google Classroom.
Per accedere:
 - ① <https://classroom.google.com>
 - ② entrare con le proprie credenziali @edu.unife.it (le stesse dell'email ricevuta all'immatricolazione).
 - ③ iscriversi al corso con la password [moiuwte](#)
- Esempi di codice, soluzioni di esercizi etc.:
<https://github.com/lbrmrc/Programmazione2024/>

Software per esercitarsi

Strumenti:

- Editor di testo: Visual Studio Code (<https://code.visualstudio.com>)
- Compilatore: GCC
- Debugger: GDB (o LLDB)
- Controllo di versione: Git

Chi ha Linux o Mac può usare gli strumenti nativi per la sua piattaforma. A chi ha Windows è consigliato usare Windows Subsystem for Linux (vedere pre-corso).