

# Informatica Teorica



Anno Accademico 2022/2023

Fabio Zanasi

<https://www.unibo.it/sitoweb/fabio.zanasi>

Prima lezione

# 1. Informazioni pratiche

# Struttura del corso

- Le lezioni si tengono dal 21 Febbraio al 5 Aprile.
- Martedì 15-18 e Mercoledì 9-12.
- 30 ore di lezioni frontali, intervallate da 10 ore di esercitazioni.
- Assistente del corso: Melissa Antonelli  
[melissaantonelli2@unibo.it](mailto:melissaantonelli2@unibo.it)  
Melissa sarà il vostro referente per qualsiasi questione relativa alle esercitazioni.

# Domande frequenti

- **Cosa c'è nell'esame?**

Il programma d'esame è dato dalle lezioni svolte in classe, il che include sia le slides che la mia spiegazione a voce.

Alcuni argomenti vengono trattati ma non sono nel programma d'esame: questo verrà sempre segnalato.

Le domande d'esame saranno simili a quelle affrontate durante le esercitazioni, ma in genere molto più semplici e brevi.

- **Le lezioni saranno pubblicate online?**

L'intenzione è di renderle disponibili su Panopto. Non ci sono tuttavia garanzie sulla qualità e la completezza della registrazione. In ogni caso le registrazioni non sostituiscono l'esperienza della lezione in classe, che rimane necessaria per coprire adeguatamente tutto il programma del corso.

# Domande frequenti

- Devo sostenere l'esame per una versione precedente del corso. Che faccio?

Sia il programma che il docente del corso è cambiato da quest'anno. Per sostenere l'esame per una versione precedente, contattare il docente precedentemente in carico, Prof. Andrea Aspertì.

# Ricevimento

Il ricevimento si tiene il Mercoledì dalle 12 alle 13, ma è possibile concordare un orario diverso. Si prega sempre di contattare il docente con buon anticipo se intendete venire a ricevimento.

Il ricevimento é per:

- Discutere di argomenti correlati al corso, non approfonditi in classe ma sui quali magari siete curiosi.
- Discutere di altre questioni legate al vostro percorso di studi (tesi di laurea, esperienza all'estero, magistrale/dottorato...)
- Chiarificare concetti discussi in classe, a condizione che siate venuti a lezione e abbiate già provato a comprenderli in autonomia.

Il ricevimento non è per:

- Risolvere gli esercizi insieme (le esercitazioni hanno proprio questo scopo, e per chiarimenti ulteriori potete contattare Melissa Antonelli).
- Recuperare lezioni a cui non avete partecipato.

# Prerequisiti

Teoria degli insiemi e logica di base

Saper scrivere una dimostrazione matematica in modo  
*chiaro e rigoroso.*

**Opzionale, ma aiuta:**

Dimostrazioni per contraddizione.

Automi a stati, grammatiche, linguaggi formali.

# Bibiografia

Principale:

M. Sipser - Introduction to the Theory of Computation

Opzionali per approfondimenti:

A.J. Kfoury, R. N. Moll, M. A. Arbib - A Programming Approach to Computability

J. Hopcroft, J. Ullman - Introduction to Automata Theory,  
Languages and Computation

B. Barak, S. Arora - Computational Complexity: A Modern  
Approach

Domande?

## 2. Sguardo d'insieme sul corso

# Domande 'urgenti'

- Quali linguaggi di programmazione dovrei conoscere?
- Quale software dovrei imparare a usare?
- Quali competenze professionali dovrei acquisire?
- Cos'è più importante per il mio CV?

**La risposta sarà la stessa tra 2, 5 o 10 anni?**

# Domande 'urgenti'



I linguaggi di programmazione cambieranno  
I modelli di calcolo cambieranno  
**I concetti fondamentali** dell'informatica non cambieranno.

# Fondamenti

**Cos'è un linguaggio di programmazione?**

**Cos'è un computer?**

**Ci sono limiti a ciò che possiamo programmare?**

In che modo possiamo approcciare  
*formalmente* queste domande?

Possiamo sviluppare una *teoria matematica*  
che risponda in modo chiaro ed esauriente?

# Che cos'è un computer?



# Che cos'è un computer?

Che cosa distingue un computer da un non-computer?  
Che cos'è l'**essenza** della computazione?

Una **teoria** della computabilità richiede un **modello astratto** di cosa sia uno strumento di calcolo.

# Che cos'è un computer?

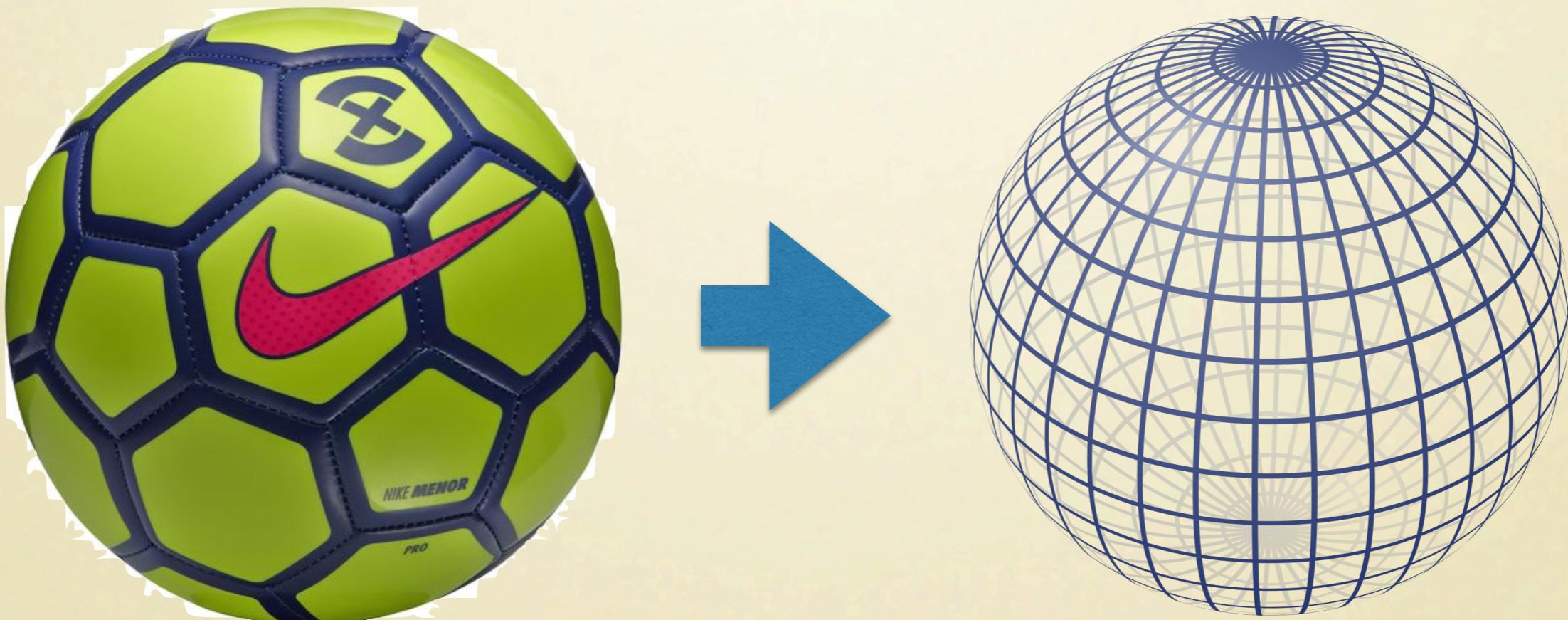
Aristotele: distinzione tra le proprietà **essenziali** e quelle **accidentali** di un oggetto

*Una sedia può essere fatta di legno o di metallo, ma questa proprietà è accidentale, ovvero, essa rimane una sedia indipendentemente dal materiale di cui è fatta.*



# Che cos'è un computer?

In Geometria: al fine di studiare formalmente lo spazio, dobbiamo **astrarre** le proprietà **essenziali** degli oggetti.



# Che cos'è un computer?

Vedremo che questa domanda ha **più di una** risposta corretta.

É una macchina di Turing

É una macchina a registri

É una funzione ricorsiva

É un linguaggio di programmazione  
`sufficientemente espressivo'

**Tesi di Church-Turing:** qualunque problema che un essere umano può calcolare seguendo un algoritmo può essere calcolato indifferentemente usando uno di questi strumenti di calcolo.

# Fondamenti

**Ci sono limiti a ciò che possiamo programmare?**

# C'è un limite a ciò che un computer può calcolare?

Può un computer battere il campione del mondo di scacchi?

**1996**



Può un computer battere il campione del mondo di Go?

**2017**



Può un computer guidare un'automobile in sicurezza?

**???**



# C'è un limite a ciò che un computer può calcolare?

Alcuni problemi non saranno *mai* risolvibili da un computer.

Una teoria della computabilità può dare una *dimostrazione matematica* di questo fatto.

# Problemi irrisolvibili

Il programma P andrà in crash?

La computazione di P fermerà su un dato input?

Il programma P rispetta la specifica data dalla funzione f?

I programmi P e P' sono equivalenti?

# Altri problemi irrisolvibili

## In logica

La formula del prim'ordine  $\phi$  è soddisfacibile?

## In algebra

Quando due parole rappresentano lo stesso elemento di un gruppo? (*Word problem*)

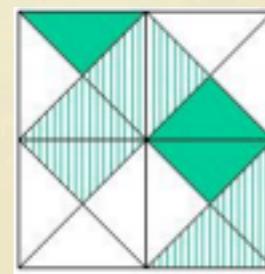
# Altri problemi irrisolvibili

## In algebra

Un'equazione diofantina data ha soluzioni intere?

## In interior design (!)

Possiamo usare un dato insieme di piastrelle per coprire una qualsiasi area quadrata?



# Problemi irrisolvibili

Alcuni problemi fondamentali dell'informatica - ma anche, più in generale, della matematica - sono irrisolvibili.

Il potere dell'astrazione: l'irrisolvibilità di buona parte di questi problemi è stata scoperta **prima** che il primo computer fosse costruito.

Uno degli aspetti centrali del corso sarà imparare a dare una **dimostrazione matematica** che un certo problema è irrisolvibile.

# Problemi irrisolvibili

Più in generale:

- La **maggior parte** dei quesiti che possiamo porre a un computer non è calcolabile.  
(Possiamo dimostrarlo usando l'argomento diagonale di Cantor.)
- Tutti i problemi universali non-triviali sono irrisolvibili  
(Teorema di Rice).

# Siamo spacciati?



Therac-25  
(1985-1987)



Challenger  
(1986)

# Siamo spacciati?

Tutt'altro. Branchi importanti dell'informatica teorica si sono sviluppate **come risposta** a queste limitazioni:

- Semantica algebrica
- Model checking
- Proof assistants (Coq, Agda, Matita...)

*Success stories:*

- **CompCert**: un compilatore per il C dimostrabilmente corretto.
- **Infer**: un tool per il debug di Facebook basato su un formalismo logico.

# Dove siamo

**Cos'è un linguaggio di programmazione?**

**Cos'è un computer?**

**Ci sono limiti a ciò che possiamo programmare?**

# Complessità Computazionale

## E le risorse di calcolo?

Un problema può essere calcolabile, ma richiedere risorse di calcolo (tempo/spazio) irragionevoli per la sua esecuzione.

Di questo si occupa la parte della teoria della computabilità detta della **complessità computazionale**.

Torneremo su questo aspetto nella seconda parte del corso.

# Scaletta del corso (provvisoria)

Settimana 1:  
la tesi di Church-Turing, la macchina di Turing e  
altri modelli di calcolo.

Settimana 2:  
la macchina universale, problemi indecidibili.  
Esercitazione 1.

Settimana 3:  
riduzioni tra problemi, teorema di Rice, diagonalizzazione.

Settimana 4:  
indecidibilità della logica del prim'ordine e incompletezza.  
Esercitazione 2.

# Scaletta del corso (provvisoria)

Settimana 5:  
classi di complessità. La classe P. Riduzioni polinomiali.

Settimana 6:  
La classe NP. Teorema di Cook-Levin.  
Esercitazione 3.

Settimana 7:  
Gerarchia di complessità. Randomness.  
Esercitazione 4.

# E ora?

Introduciamo il nostro primo  
modello di calcolo astratto:  
**La macchina di Turing**

Domande?