



Scacchi & Python

Quando la Regina incontra il Codice

Lezione 1: Il Mondo degli Scacchi è una Matrice

(e non quella del film)

 III Liceo Scientifico Biella - Scienze Applicate

 Python Biella Group



Fun Fact Iniziale

Posizioni possibili negli scacchi: 10^{120}

Atomi nell'universo osservabile: 10^{80}

Conclusione scientifica:

Gli scacchi sono letteralmente più complicati dell'universo

No pressure 😅



Perché studiare gli scacchi a scuola?




- Non per diventare campioni (tranquilli)
- Ma perché:
 - sono un sistema complesso
 - hanno regole precise
 - generano dati
 - sono la palestra storica dell'Intelligenza Artificiale



Se capisci gli scacchi, capisci come "pensa" un computer



Agenda di Oggi

1.  Rappresentazione della scacchiera
 - Coordinate e notazioni
2.  I pezzi e le loro personalità
 - Cast completo del dramma scacchistico
3.  Python-chess: il nostro strumento
 - Setup e primi passi



Parte 1

Rappresentare la Scacchiera

"Come trasformare 64 caselle in un incubo matematico"



La scacchiera come dato

La scacchiera NON è:

- legno
- marmo
- qualcosa che cade se la scuoti

La scacchiera È:

- una **struttura dati**
- con **64 posizioni**
- ognuna con **informazioni precise**



La Notazione Algebrica

Coordinate Umane

- File (colonne): a-h
- Righe: 1-8
- Casella: e4, d5, a1



Esempio:

- ♔ Re bianco parte da e1
- ♟ Pedone si muove in e4

Coordinate Python

	a	b	c	d	e	f	g	h
8
7
6
5
4
3
2
1

Esempio:

-  Re bianco parte da e1
-  Pedone si muove in e4

Perché i Matematici Hanno Rovinato Tutto

Notazione Normale

- Facile da leggere
- Intuitiva, Usata da tutti

e4 = "vai in e4"

Notazione Matematica

Perché:

- i computer amano i numeri
- le lettere sono solo numeri travestiti
- Matrici [riga][colonna]
- Indici da 0



Bitboard vs Array 8x8

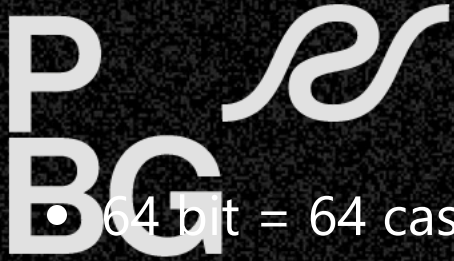
La Battaglia Epica delle Rappresentazioni

Array 8x8: Il Classico

Rappresentazione intuitiva

```
board = [
    ['♖', '♘', '♙', '♔', '♕', '♚', '♞', '♜'],
    ['♙', '♙', '♙', '♙', '♙', '♙', '♙', '♙'],
    [' ', ' ', ' ', ' ', ' ', ' ', ' ', ' '],
    [' ', ' ', ' ', ' ', ' ', ' ', ' ', ' '],
    [' ', ' ', ' ', ' ', ' ', ' ', ' ', ' '],
    [' ', ' ', ' ', ' ', ' ', ' ', ' ', ' '],
    ['♙', '♙', '♙', '♙', '♙', '♙', '♙', '♙'],
    ['♜', '♞', '♚', '♕', '♔', '♙', '♘', '♖']
]
```

- ✓ Pro: Capibile anche da tua nonna
- ✗ Contro: Lento come la burocrazia italiana



Bitboard: Il Nerd

- 64 bit = 64 caselle
- Velocissimo
- Illeggibile per gli umani

⚔️ Efficienza vs Sanità mentale

(spoiler: vince l'efficienza)

[illegible]

- ✓ Pro: Velocissimo, operazioni bit-a-bit
- ✗ Contro: Debugging = incubo esistenziale



Notazione FEN

"Come Descrivere una Partita se Fossi un Robot"

La notazione FEN (Forsyth-Edwards Notation) è un modo compatto per descrivere completamente una posizione scacchistica usando solo una stringa di testo. È come uno "screenshot testuale" della scacchiera.

Serve per:

- salvare posizioni
- analizzare partite
- far comunicare i programmi



rnbqkbnr/pppppppp/8/8/8/8/PPPPPPP/RNBQKBNR w KQkg - 0 1

Campo 6: Numero mossa

P
BG



Campo 1: Posizione dei Pezzi

Descrive i pezzi dall'8^a traversa alla 1^a (dall'alto verso il basso), separando ogni riga con / .

- **Lettere maiuscole** = pezzi bianchi
- **Lettere minuscole** = pezzi neri

Lettera	Pezzo
K/k	Re (King)
Q/q	Regina (Queen)
R/r	Torre (Rook)
B/b	Alfiere (Bishop)
N/n	Cavallo (kNight)
P/p	Pedone (Pawn)

Power

Esempio passo-passo:

Posizione iniziale:

8		r	n	b	q	k	b	n	r	→	rnbqkbnr
7		p	p	p	p	p	p	p	p	→	pppppppp
6		→	8
5		→	8
4		→	8
3		→	8
2		P	P	P	P	P	P	P	P	→	PPPPPPPP
1		R	N	B	Q	K	B	N	R	→	RNBQKBNR
		a	b	c	d	e	f	g	h		

FEN del campo 1:

rnbqkbnr/pppppppp/8/8/8/8/PPPPPPPP/RNBQKBNR



Esempio 1: Dopo 1.e4

```

8 | r n b q k b n r
7 | p p p p p p p p
6 | . . . . . . . .
5 | . . . . . . . .
4 | . . . . P . . .
3 | . . . . . . . .
2 | P P P P . P P P
1 | R N B Q K B N R

```

← Il pedone bianco è andato in e4

← Lo spazio vuoto in e2

FEN:

```
rnbqkbnr/pppppppp/8/8/4P3/8/PPPP1PPP/RNBQKBNR
```

↑

↑

e4 = 4 vuote e2 vuota

+ P

+ 3 vuote



Campo 2: Turno di Chi Gioca

Semplicissimo:

- w = tocca al **bianco** (white)
- b = tocca al **nero** (black)



Campo 3: Diritti di Arrocco

Indica quali arroccchi sono ancora **possibili** (non ancora eseguiti e non invalidi).

Simboli:

- κ = Bianco può arroccare **corto** (lato re, O-O)
- q = Bianco può arroccare **lungo** (lato regina, O-O-O)
- k = Nero può arroccare **corto**
- q = Nero può arroccare **lungo**
- - = Nessun arrocco disponibile



- $KQkq$ = Tutti e 4 gli arroccchi possibili (posizione iniziale)
- Kq = Solo bianco corto e nero lungo
- K = Solo bianco corto
- $-$ = Nessun arrocco possibile

Quando si perde il diritto di arroccare?

1. Se muovi il re
2. Se muovi una torre
3. Se una torre viene catturata



Campo 4: En Passant

Indica se è possibile catturare "en passant" e in quale casella.

Cos'è l'en passant?

Quando un pedone avanza di 2 caselle dalla posizione iniziale e finisce accanto a un pedone avversario, questo può catturarlo "al volo" come se avesse mosso di una sola casella.

Notazione:

- Casella target (dove il pedone che cattura finirà)
- - se non c'è en passant disponibile



Esempio:

Posizione prima:

```
4 | . . . . .  
3 | . . . . .  
2 | . . . . P . . . ← Pedone bianco in e2
```

Dopo e2-e4 (pedone bianco avanza di 2):

```
4 | . . . . P . . . ← Pedone bianco arriva in e4  
3 | . . . . . ← Questa è la casella "en passant"
```

Se c'è un pedone nero in d4 o f4, può catturare "passando" per e3.

FEN campo 4: e3 (la casella dove il pedone catturerebbe)

P
BG



Campo 5: Regola delle 50 Mosse

Conta il numero di **semimosse** (half-moves) dall'ultima:

- Cattura di un pezzo
- Mossa di un pedone

Serve per la **regola delle 50 mosse**: se passano 50 mosse complete (100 semimosse) senza catture o mosse di pedoni, si può dichiarare **patta**.

Esempi:

- 0 = Appena catturato un pezzo o mosso un pedone
- 15 = 15 semimosse senza eventi significativi
- 99 = Tra una semimossa alla patta!



Campo 6: Numero della Mossa

Indica a che **mossa** della partita siamo.

- Parte da 1
- Si incrementa **dopo ogni mossa del nero**

Esempi:

- 1 = Prima mossa (bianco deve ancora giocare)
- 10 = Decima mossa completa (bianco ha giocato 10 volte, nero 9)
- 50 = Cinquantesima mossa

P
BG



Esempi Completi Commentati

1. Posizione Iniziale

```
rnbqkbnr/pppppppp/8/8/8/8/PPPPPPPP/RNBQKBNR w KQkq - 0 1
```

- Pezzi in posizione standard
- Turno del bianco
- Tutti gli arroccchi possibili
- Nessun en passant
- Zero mosse senza eventi
- Mossa numero 1



2. Dopo 1.e4 c5 (Difesa Siciliana)

```
rnbqkbnr/pp1ppppp/8/2p5/4P3/8/PPPP1PPP/RNBQKBNR w KQkq c6 0 2
```

- Il nero ha mosso il pedone c7-c5
- Turno del bianco
- Tutti gli arroccchi ancora disponibili
- c6 = en passant disponibile (se il bianco avesse un pedone in d5 o b5)
- 0 = appena mosso un pedone (reset contatore)
- 2 = siamo alla seconda mossa



3 Scacco Matto del Barbiere (Scholar's Mate)

```
r1bqkb1r/pppp1Qpp/2n2n2/4p3/2B1P3/8/PPPP1PPP/RNB1K1NR b KQkq - 0 4
```

- Regina bianca in f7 fa scacco matto
- Turno del nero (ma è matto!)
- Arrocchi ancora teoricamente possibili
- Nessun en passant
- Nessuna cattura nell'ultima semimossa
- Mossa numero 4



4. Solo i Re (Posizione Impossibile da Matto)

8/8/8/4k3/8/8/8/4K3 w - - 0 1

- Solo re bianco (e1) e re nero (e5)
- Turno del bianco
- - = Nessun arrocco possibile (non ci sono torri!)
- Nessun en passant
- Partita che finirà in patta



Trucchi per Leggere FEN Velocemente

1. **Conta le / :** Ce ne devono essere esattamente 7 (8 righe - 1)
2. **Verifica la somma:** Ogni riga deve "sommare" a 8
 - Esempio: `rnbqkbnr` = 8 pezzi ✓
 - Esempio: `4P3` = $4 + 1 + 3 = 8$ ✓
 - Esempio: `8` = 8 caselle vuote ✓
3. **Maiuscole vs minuscole:** Maiuscole = bianco, minuscole = nero
4. **L'ordine è importante:** Sempre dall'8^a alla 1^a riga



Errori Comuni

✗ SBAGLIATO:

```
rnbqkbnr/pppppppp/8/8/8/8/PPPPPPPP/RNBQKBNR
```

Mancano gli altri 5 campi!

✓ GIUSTO:

```
rnbqkbnr/pppppppp/8/8/8/8/PPPPPPPP/RNBQKBNR w KQkq - 0 1
```




Esercizio Pratico

Prova a decodificare questa FEN:





```
r1bqk2r/pppp1ppp/2n2n2/2b1p3/2B1P3/5N2/PPPP1PPP/RNBQK2R w KQkq - 4 5
```

► Soluzione



Conclusione

La notazione FEN è fondamentale perché:

-  Permette di **salvare e condividere** posizioni
-  È usata da **tutti i motori scacchistici**
-  Consente di **inizializzare** partite da qualsiasi posizione
-  È lo **standard universale** per rappresentare gli scacchi

In python-chess :

```
# Creare una board da FEN
board = chess.Board("rnbqkbnr/pppppppp/8/8/8/8/PPPPPPPP/RNBQKBNR w KQkq - 0 1")

# Ottenere la FEN dalla board
fen = board.fen()
print(fen)
```




FEN: Anatomia di una Stringa

```
rnbqkbnr/pppppppp/8/8/8/8/PPPPPPPP/RNBQKBNR w KQkq - 0 1
```

Parte	Significato	Esempio
<code>rnbqkbnr/pppppppp/...</code>	Posizione pezzi	r=torre nera, P=pedone bianco
<code>w</code>	Turno	w=bianco, b=nero
<code>KQkq</code>	Arrocco disponibile	K=corto bianco, q=lungo nero
<code>-</code>	En passant	casella target o <code>-</code>
<code>0</code>	Mosse senza cattura	contatore
<code>1</code>	Numero mossa	partita



FEN: Esempi Pratici

Posizione iniziale:

```
rnbqkbnr/pppppppp/8/8/8/8/PPPPPPPP/RNBQKBNR w KQkq - 0 1
```

Dopo 1.e4:

```
rnbqkbnr/pppppppp/8/8/4P3/8/PPPP1PPP/RNBQKBNR b KQkq e3 0 1
```

Scholar's Mate (Matto del Barbiere):

```
r1bqkb1r/pppp1Qpp/2n2n2/4p3/2B1P3/8/PPPP1PPP/RNB1K1NR b KQkq - 0 4
```

Nota: i numeri rappresentano caselle vuote consecutive



Parte 2

I Pezzi e le Loro Personalità

Cast completo del dramma scacchistico



"Il Pezzo Più Importante ma Anche il Più Codardo"

Caratteristiche:

- Si muove di 1 casella
- In tutte le direzioni
- Non può andare sotto scacco
- Se muore → game over

Personalità:

- VIP con bodyguard
- Fragile come cristallo
- Deve essere protetto 24/7
- Ha il potere ma non lo usa



La Regina

"OP (Overpowered) sin dal Medioevo"

Caratteristiche:

- Torre + Alfiere combinati
- 8 direzioni
- Distanza illimitata
- Pezzo più potente

Valore: 9 punti

Personalità:

- Boss finale livello 1
- Fa quello che vuole
- Tutti la temono



La Torre

"Linee Rette e Zero Creatività"

Caratteristiche:

- Orizzontale e verticale
- Distanza illimitata
- Solida e affidabile
- Arrocco: suo momento di gloria

Valore: 5 punti

Personalità:

- Il ragioniere del gruppo
- Prevedibile ma efficace
- "Io vado dritto"



L'Alfiere

"Il Pezzo che Odia Metà Scacchiera"

Caratteristiche:

- Solo diagonali
- Un colore a vita
- Raggio illimitato
- Sempre in coppia

Valore: 3 punti

Personalità:

- Razzista cromatico
- "Caselle bianche? No grazie"
- Discrimina il 50% della board



Il Cavallo

"L'unico che Può Saltare - Probabilmente ha fatto Parkour"

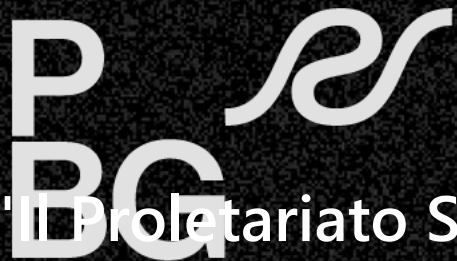
Caratteristiche:

- Movimento a "L"
- Salta sopra i pezzi
- Cambia colore casella
- Imprevedibile

Valore: 3 punti

Personalità:

- Acrobata della scacchiera
- "Le regole? Quali regole?"
- Ninja degli scacchi



Il Pedone

"Il Proletariato Scacchistico con Sogni di Mobilità Sociale"

Caratteristiche:

- Avanza di 1 (o 2 all'inizio)
- Cattura in diagonale
- Non torna indietro
- Promozione → Regina!

Valore: 1 punto (ma...)

Personalità:

- Sacrificabile
- Sogna in grande
- "Un giorno sarò Regina"



Tabella Valori Standard

Pezzo	Simbolo	Valore	Potere Relativo
Pedone		1	★
Cavallo		3	★★★
Alfiere		3	★★★
Torre		5	★★★★★
Regina		9	★★★★★★★★★
Re		∞	💀 (game over)



Parte 3

Python-Chess: Setup

"Finalmente si programma"



Installazione

Comando base:

```
pip install python-chess
```

Se non funziona:

```
pip install python-chess --force-reinstall
```

Se ancora non funziona:

```
# Avete rotto qualcosa  
# Opzioni:  
# 1. Riavviate il computer  
# 2. Piangete  
# 3. Chiedete aiuto  
# 4. Tutte le precedenti
```




Verifica Installazione

```
import chess

# Se non ci sono errori → SUCCESS! 🎉
print("python-chess installato correttamente!")

# Creiamo la prima scacchiera
board = chess.Board()
print(board)
```

Output atteso:

```
r n b q k b n r
p p p p p p p p
. . . . . . . .
. . . . . . . .
. . . . . . . .
. . . . . . . .
P P P P P P P P
```




Primo Comando: Creare l'Universo

```
import chess

# GENESI: Sia la scacchiera
board = chess.Board()

# E la scacchiera fu
print(board)

# E vide che era buona
print("Numero di pezzi:", len(board.piece_map()))
print("Turno del:", "Bianco" if board.turn else "Nero")
```

"In principio era il vuoto. Poi venne python-chess."



```
board = chess.Board()

# Chi gioca?
print(f"Turno: {board.turn}") # True = bianco, False = nero

# La partita è finita?
print(f"Game over: {board.is_game_over()}")






# C'è scacco?
print(f"Scacco: {board.is_check()}")

# Posizione in FEN
print(f"FEN: {board.fen()}")
```




Prossima Lezione: Sneak Peek

Cosa impareremo:

-  Generare tutte le mosse legali
-  Applicare mosse alla board
-  Creare un bot che gioca random
-  Capire la differenza tra "pseudo-legal" e "legal"
-  Far combattere bot random uno contro l'altro

Spoiler: il caos sarà bellissimo  



Obiettivi di Oggi

- [x] Capito che gli scacchi > universo
- [x] Imparato le coordinate della scacchiera
- [x] Conosciuto FEN (il linguaggio robot)
- [x] Scoperto le personalità dei pezzi
- [x] Installato python-chess
- [x] Creato la prima scacchiera in Python



Achievement Unlocked:

"Chess Coder Level 1"



Sfida per Casa

1. Installa `python-chess` sul tuo computer
2. Crea una scacchiera e stampala
3. Sperimenta con posizioni FEN custom
4. **Bonus:** Prova a creare una posizione impossibile e vedi cosa succede

```
# Esempio posizione custom  
board = chess.Board("8/8/8/4k3/8/8/8/4K3 w - - 0 1")  
# Solo i due re... è legale?
```




Risorse Utili

- Documentazione: <https://python-chess.readthedocs.io>
- Repository GitHub: <https://github.com/niklasf/python-chess>
- Lichess.org: Per giocare e studiare online
- Chess.com: Alternative con tutorial

Pro tip: La documentazione è vostra amica. Leggere docs > cercare su StackOverflow per 2 ore



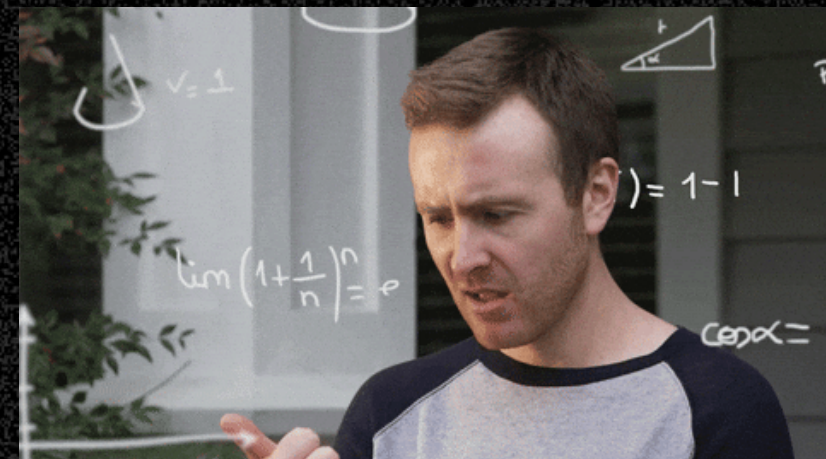
? Q&A

Domande?

Dubbi?

Crisi esistenziali?

"Non esistono domande stupide, solo risposte non ancora cercate su Google"





Fine Lezione 1

Prossima volta:

"Mosse Legali e Mosse Stupide: Una Guida Completa"

 Keep Coding, Keep Playing 



Meme di Chiusura

Programmatori dopo aver installato python-chess:

"I've got the power!" 🎵

Programmatori dopo aver provato a capire bitboard:

"What have I done?" 🤯



Ma noi useremo python-chess che ci nasconde la complessità

Quindi siamo tutti winner 🏆

