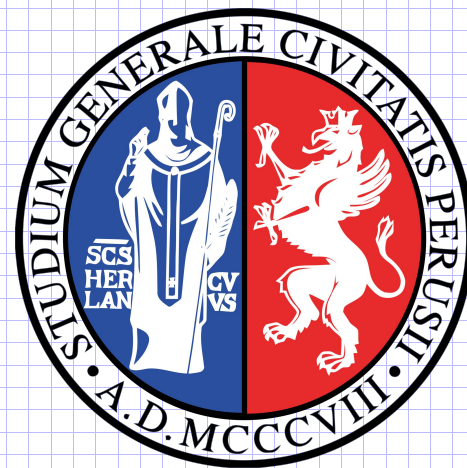


# I numeri di Fibonacci

Manuale per l'uso

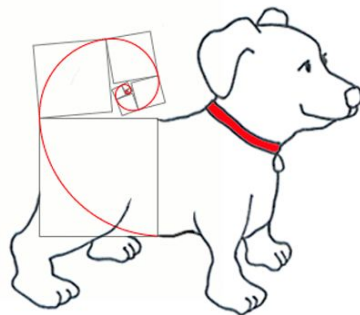
Tommaso Romani  
Nicolo Vescera



# Introduzione

In questa lezione tratteremo i seguenti argomenti:

1. Successioni matematiche
1. Fibonacci e i suoi numeri
2. Contestualizzazione storica dei numeri di Fibonacci
3. Proprietà della successione di Fibonacci
5. Riscontri nella natura
8. Implementazione di Fibonacci



# Successioni

Definizione ed esempi di successioni  
matematiche.

**3, 7, 9, 21, 15, 35, ... ?**

**1, 4, 9, 16, 25, ... ?**

**1, 1, 2, 3, 5, 8, 13, ... ?**

## Definizione di successione

Una successione è una legge che associa ad ogni numero naturale  $n$  un numero reale  $a_n$

*Formalmente può quindi essere definita con la seguente funzione.*

$$f : \mathbb{N} \rightarrow \mathbb{R}$$

**Quesito:**

Provare ad elencare alcune successioni ...

## Esempi di successioni

$1, 2, 3, 4, 5, 6, \dots$

$3, 6, 9, 12, 15, 18, \dots$

$1, \frac{1}{2}, \frac{1}{4}, \frac{1}{8}, \frac{1}{16}, \frac{1}{32}, \dots$

# Successione di Fibonacci

La successione di fibonacci è una sequenza matematica che viene regolata dalla seguente funzione:

$$F_n = \begin{cases} 1 & \text{se } n = 1 \\ 1 & \text{se } n = 2 \\ F_{n-1} + F_{n-2} & \text{se } n \geq 3 \end{cases}$$

**Quesito:**

Provare a calcolare i primi 10 numeri della successione di Fibonacci

# Successione di Fibonacci

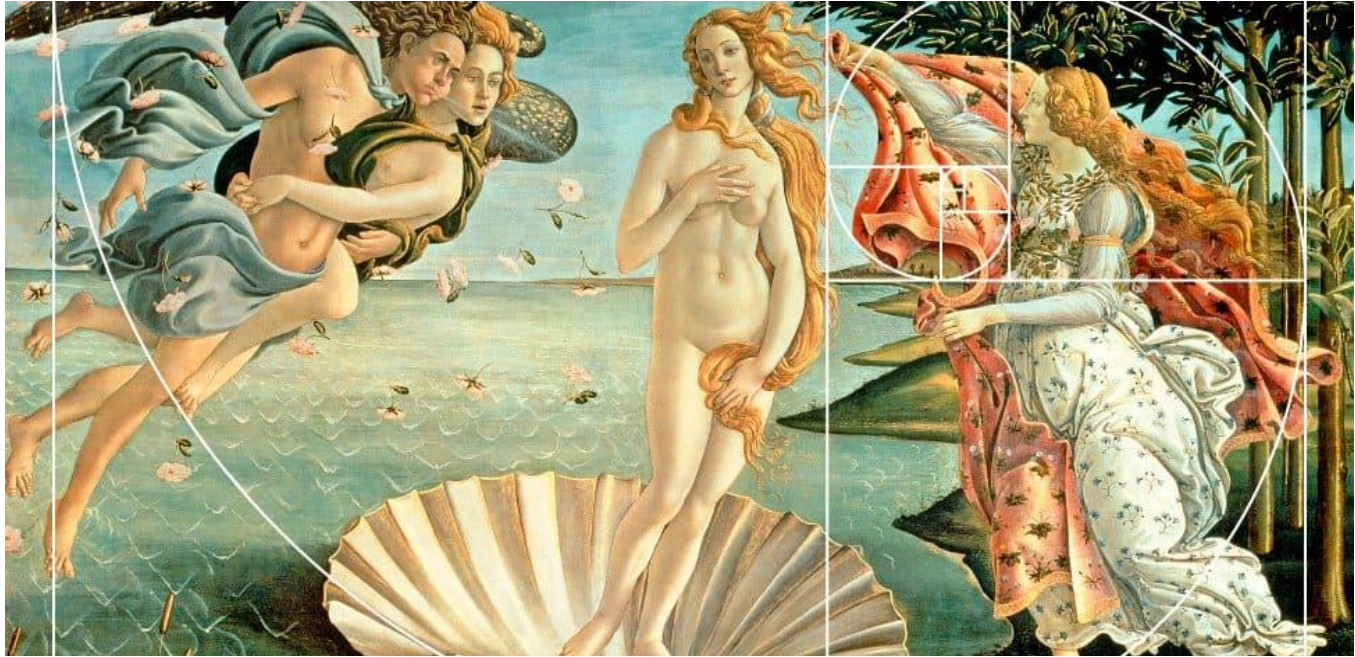
Questa particolare successione può essere osservata nei più disparati campi, dalle ramificazioni di un albero ad una stretta relazione con il triangolo di tartaglia.

1, 1, 2, 3, 5, 8, 13, 21, 34, 55, ...

Come ha fatto Fibonacci a scoprire questa successione?

# Cenni storici

Cenni storici riguardanti la vita di Fibonacci





# La storia della successione

Leonardo Bonacci nacque a Pisa circa nel 1170.

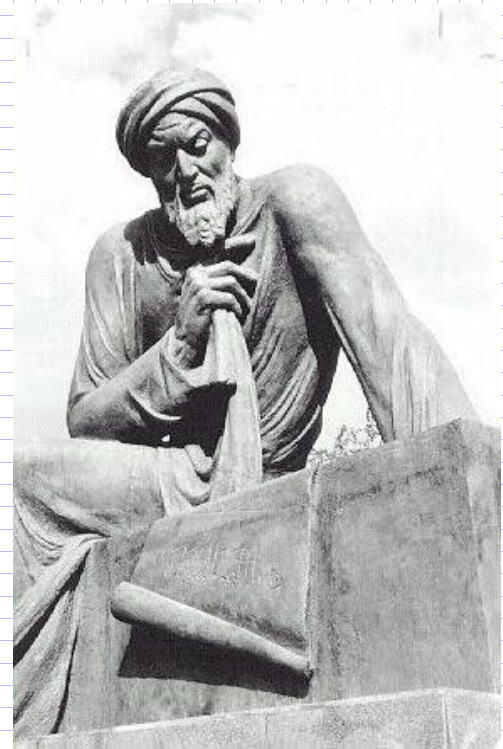
Di famiglia benestante fu spronato dal padre a diventare mercante, mestiere che lo portò a viaggiare molto e entrare in contatto con svariate scuole matematiche.



# La storia della successione

Durante i suoi viaggi in medio oriente entrò in contatto con il pensiero di svariati matematici che alimentarono la sua passione per i numeri, tra cui al-Khwārizmī, padre dell'algebra moderna.

Tornato in Italia produsse molte opere, tra cui la più importante è senza dubbio il *Liber Abbacci*, un trattato che introdusse in Europa il sistema decimale indo arabo, con relativi metodi di calcolo.



# La storia della successione

Guadagnò molta notorietà, tanto che gli venne riconosciuto un vitalizio dall'imperatore Federico II per poter continuare i suoi studi.

In questo periodo scoprì la successione a cui da il nome cercando di creare un modello rappresentativo che potesse esprimere la crescita di una popolazione di conigli.



# La storia della successione








## Formulazione originale:

- Si parte con una coppia di conigli
- Ogni coppia diventa fertile dopo un mese di vita
- Una volta fertile ogni coppia genera un volta al mese un solo altra coppia

## Evoluzione del modello:

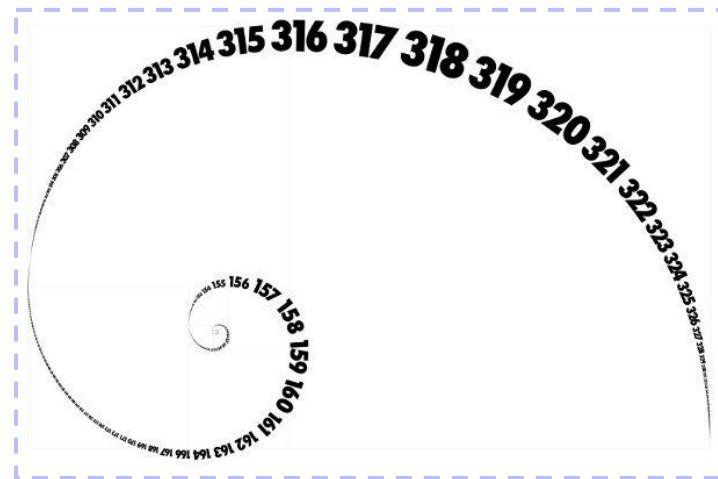
- Dopo 1 mese la prima coppia diventa fertile
- Dopo 2 mesi ci sono 2 coppie, di cui 1 sola fertile
- Dopo 3 mese le coppie sono 3, di cui solo 2 fertili
- Dopo 4 mesi le coppie sono 5, di cui 3 fertili

# La storia della successione

Mesi	Coppie nate	Coppie adulte *	Totale coppie	Evoluzione nascite nei primi sei mesi
Inizio	0	1	1	
1°	1	1	2	
2°	1	2	3	
3°	2	3	5	
4°	3	5	8	
5°	5	8	13	
6°	8	13	21	

# Proprietà

## Proprietà della successione di Fibonacci



# Proprietà della successione di Fibonacci 1

La successione di Fibonacci gode delle seguenti proprietà:

- In tutta la successione compaiono solo 3 quadrati perfetti e un cubo

$$F_1 = 1, F_2 = 1, F_{12} = 144 \mid F_6 = 8$$

- Due numeri consecutivi nella successione di Fibonacci sono coprimi
- Tutti i numeri dopo il quarto che sono primi avranno anche indice primo
- Se  $n$  è divisore di  $m$  allora  $F_n$  divide  $F_m$
- L'MCD tra  $F_n$  e  $F_m$  corrisponde al numero di Fibonacci la cui posizione è l'MCD tra  $n$  e  $m$

# Proprietà della successione di fibonacci 2

- Il quadrato di ogni numero di Fibonacci differisce di uno dal prodotto dei due numeri di fianco ad esso

Dati i seguenti numeri:  $F_5 = 5$ ,  $F_6 = 8$ ,  $F_7 = 13$

Allora avremo che:  $F_6^2 = F_5 \cdot F_7 - 1$

- Se si sommano i primi  $n$  numeri di Fibonacci e poi si aggiunge 1, il risultato sarà uguale al numero di indice  $n+2$

$$\left( \sum_{i=0}^n F_i \right) + 1 = F_{n+2}$$





# La Poesia della Successione di Fibonacci

Riscontri nel mondo della successione di Fibonacci.



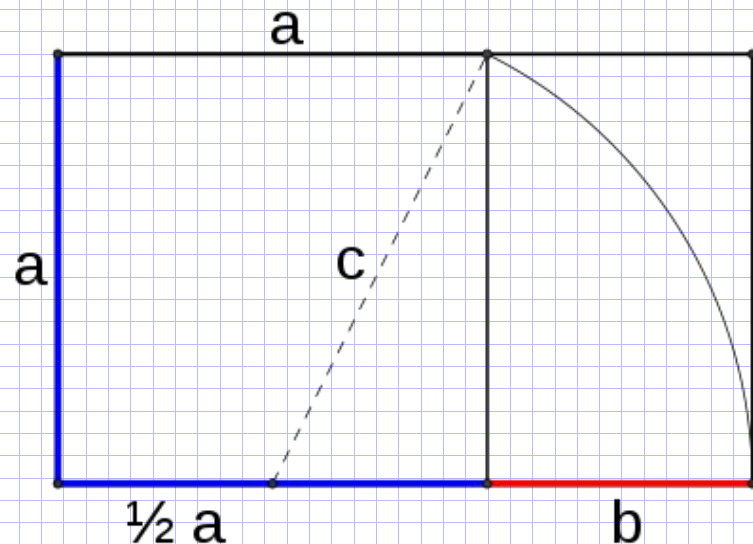
# Sezione aurea

È il rapporto tra due grandezze di cui la maggiore è medio proporzionale tra la minore e la somma della 2.

La costante aurea corrisponde a :

1,6180339887...

Il rapporto tra due numeri di Fibonacci consecutivi approssima viva via sempre di più questa costante

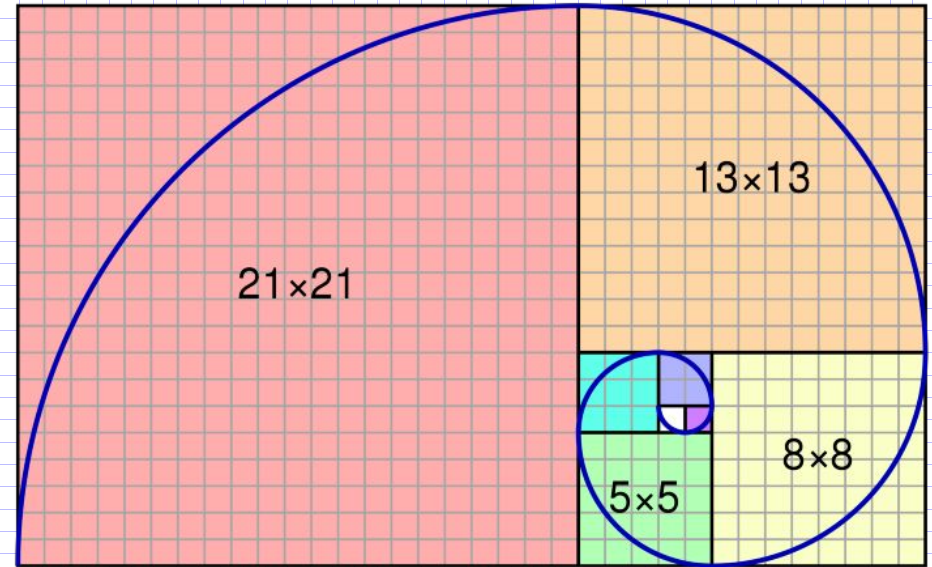


# Sezione aurea

Costante aurea	1,618033
$2 \div 1$	2.000000
$3 \div 2$	1.500000
$5 \div 3$	1.666666
$8 \div 5$	1.600000
$13 \div 8$	1.625000
$21 \div 13$	1.615384
$34 \div 21$	1.619047
$55 \div 34$	1.617647
$89 \div 55$	1.618888
...	...

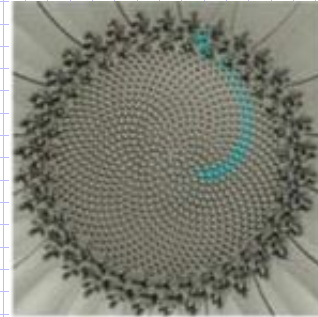
# Spirale aurea

Dalla sequenza di Fibonacci è possibile disegnare il rettangolo aureo, con cui poi si può tracciare la spirale di fibonacci, che è un'ottima approssimazione della spirale aurea

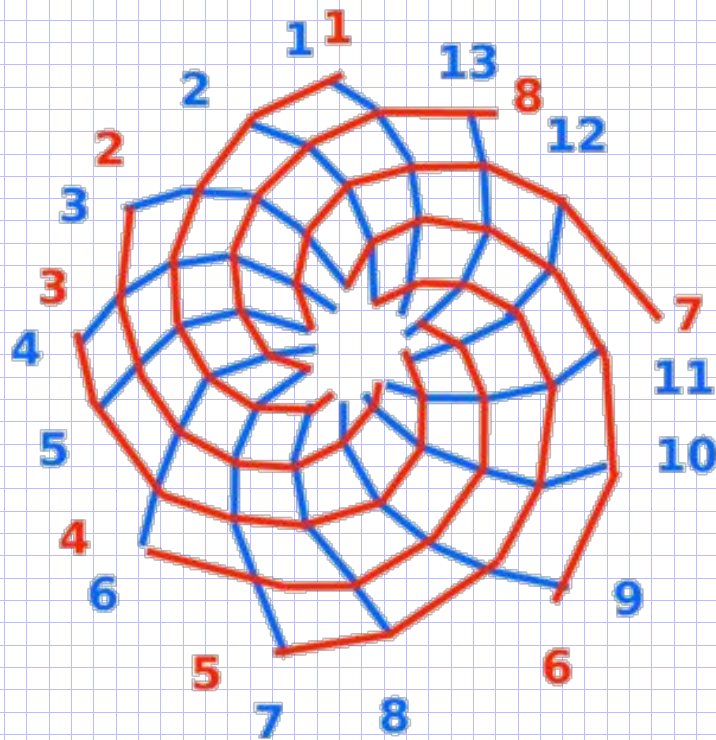
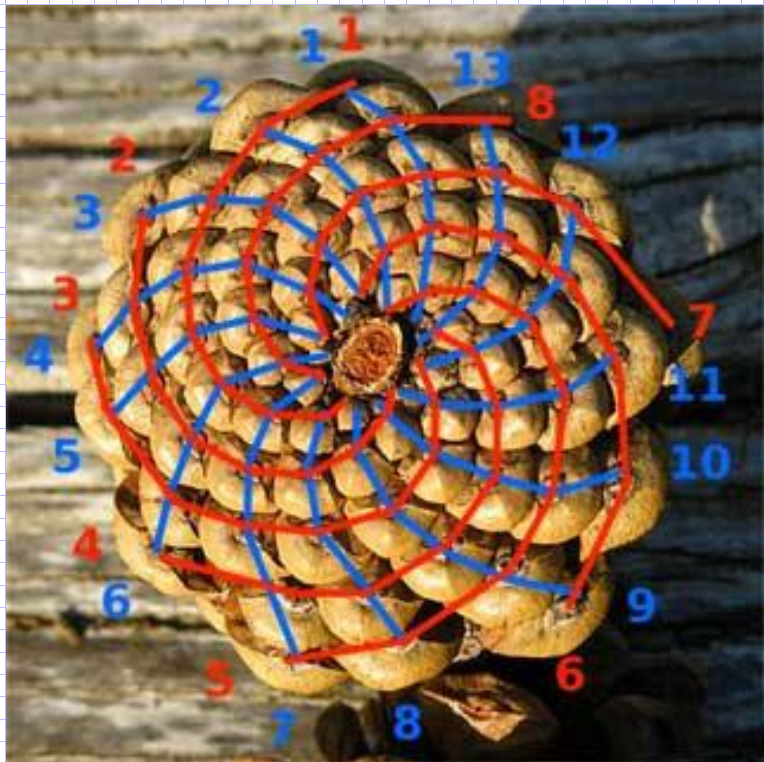


# Fibonacci e i girasoli

La spirale aurea appare nella disposizione dei semi di girasole all'interno del capolino

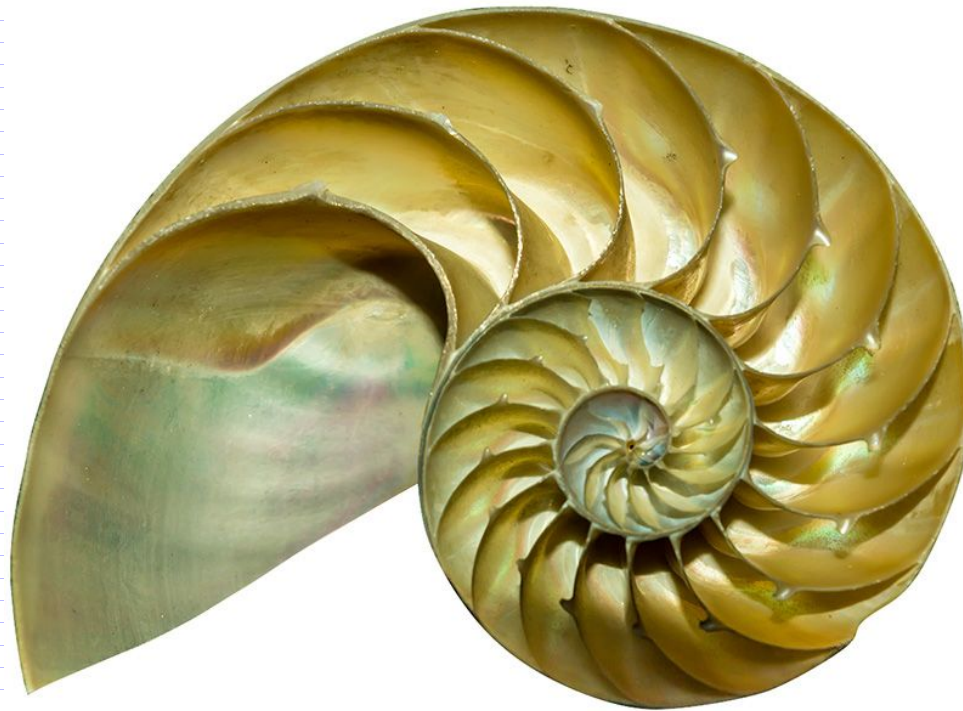


# Fibonacci e le pigne



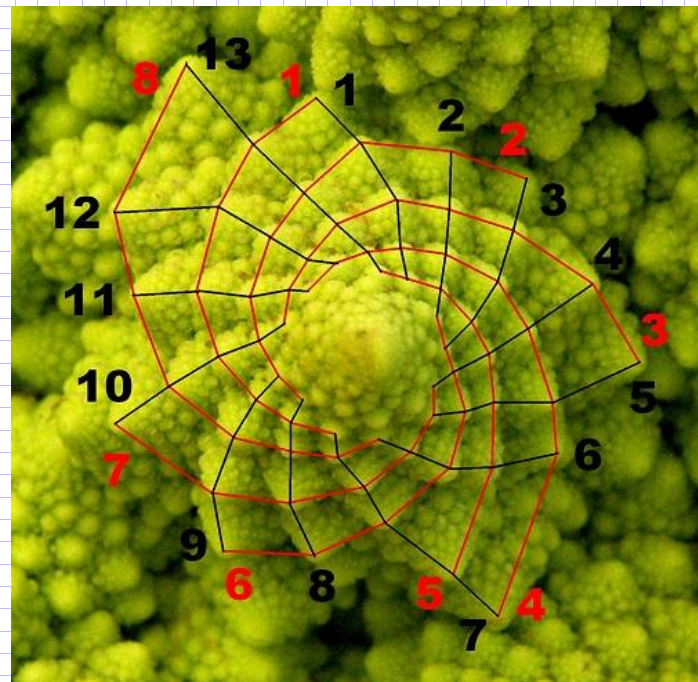


# Fibonacci e le conchiglie





# Fibonacci e i cavoli



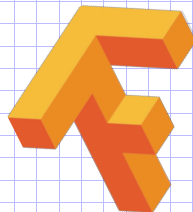
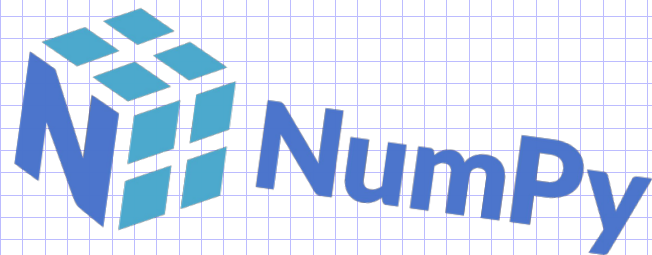
# Python ❤️ Fibonacci!

Implementazione pratica dell'algoritmo per calcolare la successione di Fibonacci.



# Perchè Python ?

- Linguaggio interpretato
- Alto Livello
- Semplice
- Enorme Community
- Divertente



Python >>>>>> others

## Ecco perché !



```
1 pip install py-fibonacci
```



```
1 from fibonacci import fibonacci  
2  
3 fib = fibonacci(10)  
4 print(fib)
```

# Una possibile soluzione ?

# Ricorsione !

Quesito:

Prova a trovare una tua possibile versione di pseudocodice !

```
1 function integer fibonacci(integer idx) {  
2   // caso base  
3   switch(idx) {  
4     case 0: return 0;  
5     case 1: return 1;  
6     case 2: return 1;  
7   }  
8  
9   // ricorsione  
10  integer result = fibonacci(idx - 1) + fibonacci(idx - 2);  
11  return result;  
12 }
```

# E la versione iterativa ?

Quesito:

Prova a trovare una versione  
iterativa !

```
1 function integer fibonacci_iter(integer idx) {  
2     // inizializzo variabili per iterazione  
3     integer n = 0;  
4     integer n1 = 0;  
5     integer n2 = 1;  
6     integer count = 0;  
7  
8     while (count < idx) {  
9         // calcolo numero successivo  
10        n = n1 + n2;  
11  
12        // preparo le variabili per la nuova iterazione  
13        n1 = n2;  
14        n2 = n;  
15  
16        count ++;  
17    }  
18  
19    return n1;  
20 }
```

# Implementazione in Python - Ricorsione

```
1 def fibonacci(idx):  
2     # caso base  
3     if idx == 0:  
4         return 0  
5     elif idx == 1:  
6         return 1  
7     elif idx == 2:  
8         return 1  
9  
10    # ricorsione  
11    result = fibonacci(idx - 1) + fibonacci(idx - 2)  
12  
13    return result
```

# Implementazione in Python - Iterazione

```
1 def fibonacci_iter(idx):
2     n1, n2 = 0, 1
3     count = 0
4
5     while count < idx:
6         # calcolo successiva posizione
7         n = n1 + n2
8
9         # preparo nuova iterazione
10        n1 = n2
11        n2 = n
12
13        count += 1
14
15    return n1
```



# Implementazione in Python - Iterazione

```
1 def fibonacci_iter(idx):
2     n1, n2 = 0, 1
3     count = 0
4
5     while count < idx:
6         print(n1)
7
8         # calcolo successiva posizione
9         n = n1 + n2
10
11        # preparo nuova iterazione
12        n1 = n2
13        n2 = n
14
15        count += 1
16
17    print(n1)
18
19    return n1
```



# Fibonacci in depth

Confronto tra implementazione  
ricorsiva ed iterativa con alcune  
ottimizzazioni.

# Ricorsione vs Iterazione

Quesito:

Quale versione risulterà più veloce ??

Ricorsiva:

Calcolo	Tempo
fibonacci(10)	< 0s
fibonacci(50)	TROPPO
fibonacci(100)	TROPPO

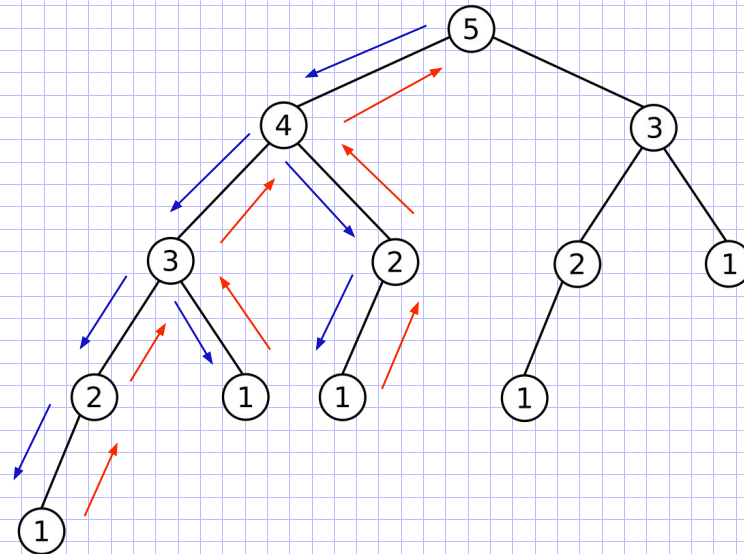
Iterativa:

Calcolo	Tempo
fibonacci(10)	< 0s
fibonacci(50)	< 0s
fibonacci(100)	< 0s

# Problemi della ricorsione

Quesito:

Per quale motivo la ricorsione richiede così tanto tempo ?



# Python + Memoization

La Memoizzazione (Memoization) è una tecnica di programmazione che consiste nel salvare in memoria risultati di una funzione per poi riutilizzarli in futuro senza doverli ricalcolare.


Python mette a disposizione un modulo chiamato *functools* utile a questo scopo. Molte sono le funzioni al suo interno, ma noi ci concentreremo sulla funzione *cache*.

# **from functools import cache**

La funzione cache è una speciale funzione detta wrapper (a.k.a. Decorator) che va ad estendere il comportamento di un'altra funzione (presa come input).

Questa funzione crea un dizionario (lookup dictionary) al cui interno salverà il risultato di ogni chiamata della funzione in ingresso.

# Ricorsione ottimizzata



```
1 from functools import cache
2
3 @cache
4 def fibonacci(idx):
5     # caso base
6     if idx == 0:
7         return 0
8     elif idx == 1:
9         return 1
10    elif idx == 2:
11        return 1
12
13    # ricorsione
14    result = fibonacci(idx - 1) + fibonacci(idx - 2)
15
16    return result
```

# Ricorsione ottimizzata

Ricorsiva:

Calcolo	Tempo
fibonacci(10)	< 0s
fibonacci(50)	TROPPO
fibonacci(100)	TROPPO

Ricorsiva + @cache:

Calcolo	Tempo
fibonacci(10)	< 0s
fibonacci(50)	< 0s
fibonacci(100)	< 0s

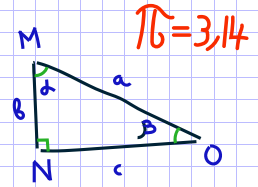
Quesito:

Provare a ricalcolare l'albero di ricorsione della slide 36 con la tecnica della Memoization.



# Fine !

## Our Team



**Tommaso  
Romani**

350646



**Nicolò  
Vescera**

349144