Processamento da Informação

Google Meet - 01

http://professor.ufabc.edu.br/~jesus.mena/courses/pi-1q-2020/PI-google-meet/

Testando sequências

Sequência de números pares

```
1 2
2 4
3 6
4 8
5 10
6 12
7 14
8 16
9 18
10 20
11 22
12 24
13 26
14 28
15 30
16 32
17 34
18 36
19 38
20 40
```

```python
# Sequencia de 20 numeros pares
for i in range(1,21):
  print (i, 2*i)
```

Sequência de números ímpares

```
1 1
2 3
3 5
4 7
5 9
6 11
7 13
8 15
9 17
10 19
11 21
12 23
13 25
14 27
15 29
16 31
17 33
18 35
19 37
20 39
```

```python
# Sequencia de 20 numeros impares
for i in range(1,21):
  print (i, 2*i-1)
```

Sequência de pares: 2 2 4 4 6 6 8 8 ...

```
1 2
2 2
3 4
4 4
5 6
6 6
7 8
8 8
9 10
10 10
11 12
12 12
13 14
14 14
15 16
16 16
17 18
18 18
19 20
20 20
```

Como podemos gerar essa sequência de números?

Sequência de pares: 2 2 4 4 6 6 8 8 ...

```
1 2
2 2
3 4
4 4
5 6
6 6
7 8
8 8
9 10
10 10
11 12
12 12
13 14
14 14
15 16
16 16
17 18
18 18
19 20
20 20
```

Como podemos gerar essa sequência de números?

```
for i in range(1,21):
  print (i, i+i%2)
```

Sequência de pares: 1 3 3 5 5 7 7 9 9 ...

```
1  1
2  3
3  3
4  5
5  5
6  7
7  7
8  9
9  9
10 11
11 11
12 13
13 13
14 15
15 15
16 17
17 17
18 19
19 19
20 21
```

Como podemos gerar essa sequência de números?

Sequência de pares: 1 3 3 5 5 7 7 9 9 ...

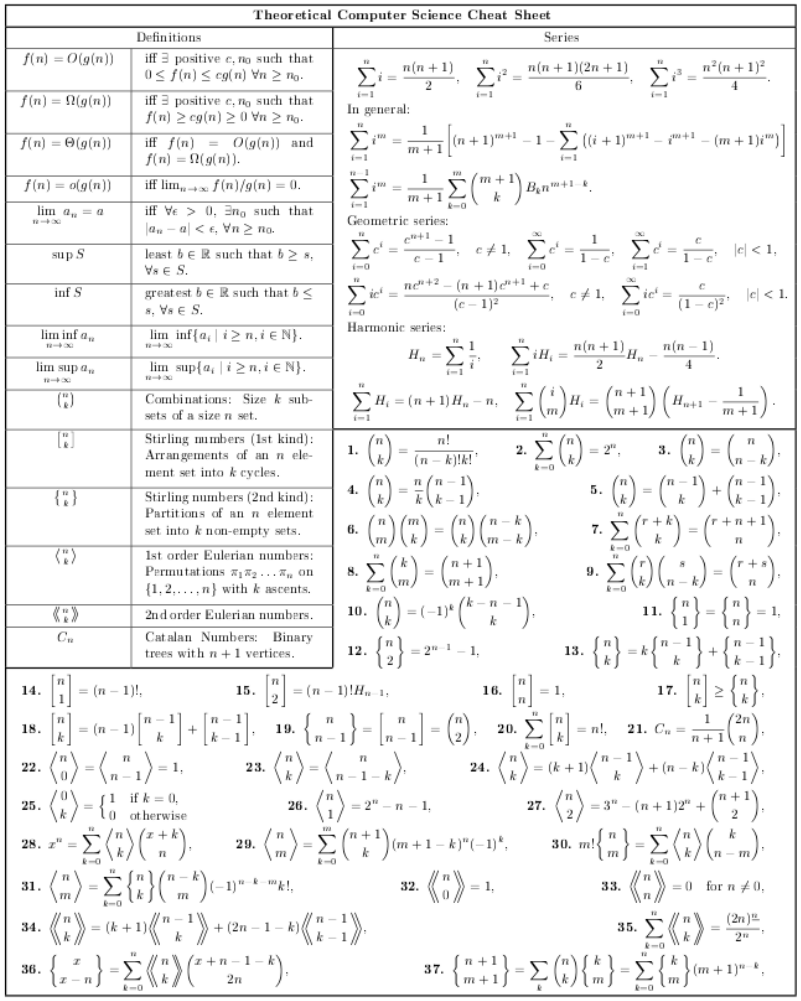
```
1 1
2 3
3 3
4 5
5 5
6 7
7 7
8 9
9 9
10 11
11 11
12 13
13 13
14 15
15 15
16 17
17 17
18 19
19 19
20 21
```

Como podemos gerar essa sequência de números?

```
for i in range(1,21):
  print (i, i+1-i%2)
```

Um material importante

Theorical Computer Science Cheat sheet (folha-de-dicas.pdf)
http://professor.ufabc.edu.br/~jesus.mena/courses/pi-1q-2020/PI-google-meet/



Theoretical Computer Science Cheat Sheet

Definitions

$f(n) = O(g(n))$	iff $\exists$ positive c, n_0 such that $0 \le f(n) \le cg(n)\ \forall n \ge n_0$.
$f(n) = \Omega(g(n))$	iff $\exists$ positive c, n_0 such that $f(n) \ge cg(n) \ge 0\ \forall n \ge n_0$.
$f(n) = \Theta(g(n))$	iff $f(n) = O(g(n))$ and $f(n) = \Omega(g(n))$.
$f(n) = o(g(n))$	iff $\lim_{n\to\infty} f(n)/g(n) = 0$.
$\lim_{n\to\infty} a_n = a$	iff $\forall \epsilon > 0$, $\exists n_0$ such that $\lvert a_n - a\rvert < \epsilon$, $\forall n \ge n_0$.
$\sup S$	least $b \in \mathbb{R}$ such that $b \ge s$, $\forall s \in S$.
$\inf S$	greatest $b \in \mathbb{R}$ such that $b \le s$, $\forall s \in S$.
$\liminf_{n\to\infty} a_n$	$\lim_{n\to\infty} \inf\{a_i \mid i \ge n, i \in \mathbb{N}\}$.
$\limsup_{n\to\infty} a_n$	$\lim_{n\to\infty} \sup\{a_i \mid i \ge n, i \in \mathbb{N}\}$.
$\binom{n}{k}$	Combinations: Size k subsets of a size n set.
$\left[{n \atop k}\right]$	Stirling numbers (1st kind): Arrangements of an n element set into k cycles.
$\left\{{n \atop k}\right\}$	Stirling numbers (2nd kind): Partitions of an n element set into k non-empty sets.
$\left\langle{n \atop k}\right\rangle$	1st order Eulerian numbers: Permutations $\pi_1\pi_2\ldots\pi_n$ on $\{1, 2, \ldots, n\}$ with k ascents.
$\left\langle\!\left\langle{n \atop k}\right\rangle\!\right\rangle$	2nd order Eulerian numbers.
C_n	Catalan Numbers: Binary trees with $n+1$ vertices.

Series

$$\sum_{i=1}^{n} i = \frac{n(n+1)}{2}, \quad \sum_{i=1}^{n} i^2 = \frac{n(n+1)(2n+1)}{6}, \quad \sum_{i=1}^{n} i^3 = \frac{n^2(n+1)^2}{4}.$$

In general:

$$\sum_{i=1}^{n} i^m = \frac{1}{m+1}\left[(n+1)^{m+1} - 1 - \sum_{i=1}^{n}\left((i+1)^{m+1} - i^{m+1} - (m+1)i^m\right)\right]$$

$$\sum_{i=1}^{n-1} i^m = \frac{1}{m+1}\sum_{k=0}^{m}\binom{m+1}{k} B_k n^{m+1-k}.$$

Geometric series:

$$\sum_{i=0}^{n} c^i = \frac{c^{n+1}-1}{c-1}, \quad c \ne 1, \quad \sum_{i=0}^{\infty} c^i = \frac{1}{1-c}, \quad \sum_{i=1}^{\infty} c^i = \frac{c}{1-c}, \quad |c| < 1,$$

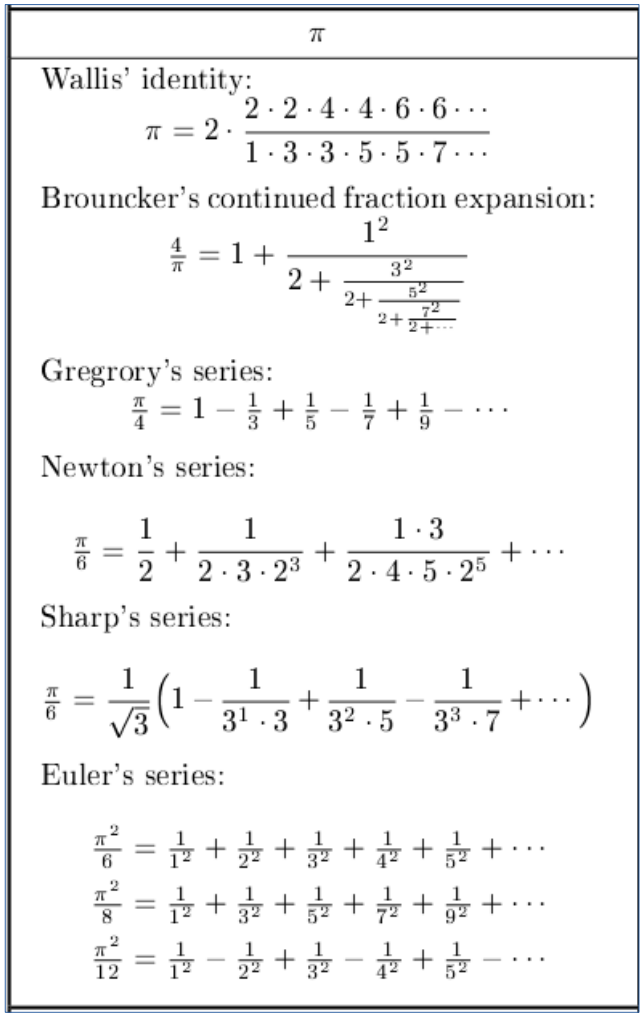
$$\sum_{i=0}^{n} ic^i = \frac{nc^{n+2} - (n+1)c^{n+1} + c}{(c-1)^2}, \quad c \ne 1, \quad \sum_{i=0}^{\infty} ic^i = \frac{c}{(1-c)^2}, \quad |c| < 1.$$

Harmonic series:

$$H_n = \sum_{i=1}^{n} \frac{1}{i}, \qquad \sum_{i=1}^{n} iH_i = \frac{n(n+1)}{2}H_n - \frac{n(n-1)}{4}.$$

$$\sum_{i=1}^{n} H_i = (n+1)H_n - n, \qquad \sum_{i=1}^{n}\binom{i}{m}H_i = \binom{n+1}{m+1}\left(H_{n+1} - \frac{1}{m+1}\right).$$

1. $\binom{n}{k} = \frac{n!}{(n-k)!k!}$
2. $\sum_{k=0}^{n}\binom{n}{k} = 2^n$
3. $\binom{n}{k} = \binom{n}{n-k}$
4. $\binom{n}{k} = \frac{n}{k}\binom{n-1}{k-1}$
5. $\binom{n}{k} = \binom{n-1}{k} + \binom{n-1}{k-1}$
6. $\binom{n}{m}\binom{m}{k} = \binom{n}{k}\binom{n-k}{m-k}$
7. $\sum_{k=0}^{n}\binom{r+k}{k} = \binom{r+n+1}{n}$
8. $\sum_{k=0}^{n}\binom{k}{m} = \binom{n+1}{m+1}$
9. $\sum_{k=0}^{n}\binom{r}{k}\binom{s}{n-k} = \binom{r+s}{n}$
10. $\binom{n}{k} = (-1)^k\binom{k-n-1}{k}$
11. $\left\{{n \atop 1}\right\} = \left\{{n \atop n}\right\} = 1$
12. $\left\{{n \atop 2}\right\} = 2^{n-1} - 1$
13. $\left\{{n \atop k}\right\} = k\left\{{n-1 \atop k}\right\} + \left\{{n-1 \atop k-1}\right\}$
14. $\left[{n \atop 1}\right] = (n-1)!$
15. $\left[{n \atop 2}\right] = (n-1)!H_{n-1}$
16. $\left[{n \atop n}\right] = 1$
17. $\left[{n \atop k}\right] \ge \left\{{n \atop k}\right\}$
18. $\left[{n \atop k}\right] = (n-1)\left[{n-1 \atop k}\right] + \left[{n-1 \atop k-1}\right]$
19. $\left\{{n \atop n-1}\right\} = \left[{n \atop n-1}\right] = \binom{n}{2}$
20. $\sum_{k=0}^{n}\left[{n \atop k}\right] = n!$
21. $C_n = \frac{1}{n+1}\binom{2n}{n}$
22. $\left\langle{n \atop 0}\right\rangle = \left\langle{n \atop n-1}\right\rangle = 1$
23. $\left\langle{n \atop k}\right\rangle = \left\langle{n \atop n-1-k}\right\rangle$
24. $\left\langle{n \atop k}\right\rangle = (k+1)\left\langle{n-1 \atop k}\right\rangle + (n-k)\left\langle{n-1 \atop k-1}\right\rangle$
25. $\left\langle{0 \atop k}\right\rangle = \begin{cases}1 & \text{if } k = 0,\\ 0 & \text{otherwise}\end{cases}$
26. $\left\langle{n \atop 1}\right\rangle = 2^n - n - 1$
27. $\left\langle{n \atop 2}\right\rangle = 3^n - (n+1)2^n + \binom{n+1}{2}$
28. $x^n = \sum_{k=0}^{n}\left\langle{n \atop k}\right\rangle\binom{x+k}{n}$
29. $\left\langle{n \atop m}\right\rangle = \sum_{k=0}^{m}\binom{n+1}{k}(m+1-k)^n(-1)^k$
30. $m!\left\{{n \atop m}\right\} = \sum_{k=0}^{n}\left\langle{n \atop k}\right\rangle\binom{k}{n-m}$
31. $\left\langle{n \atop m}\right\rangle = \sum_{k=0}^{n}\left\{{n \atop k}\right\}\binom{n-k}{m}(-1)^{n-k-m}k!$
32. $\left\langle\!\left\langle{n \atop 0}\right\rangle\!\right\rangle = 1$
33. $\left\langle\!\left\langle{n \atop n}\right\rangle\!\right\rangle = 0 \quad \text{for } n \ne 0$
34. $\left\langle\!\left\langle{n \atop k}\right\rangle\!\right\rangle = (k+1)\left\langle\!\left\langle{n-1 \atop k}\right\rangle\!\right\rangle + (2n-1-k)\left\langle\!\left\langle{n-1 \atop k-1}\right\rangle\!\right\rangle$
35. $\sum_{k=0}^{n}\left\langle\!\left\langle{n \atop k}\right\rangle\!\right\rangle = \frac{(2n)^n}{2^n}$
36. $\left\{{x \atop x-n}\right\} = \sum_{k=0}^{n}\left\langle\!\left\langle{n \atop k}\right\rangle\!\right\rangle\binom{x+n-1-k}{2n}$
37. $\left\{{n+1 \atop m+1}\right\} = \sum_{k}\binom{n}{k}\left\{{k \atop m}\right\} = \sum_{k=0}^{n}\left\{{k \atop m}\right\}(m+1)^{n-k}$



π

Wallis' identity:

$$\pi = 2 \cdot \frac{2 \cdot 2 \cdot 4 \cdot 4 \cdot 6 \cdot 6 \cdots}{1 \cdot 3 \cdot 3 \cdot 5 \cdot 5 \cdot 7 \cdots}$$

Brouncker's continued fraction expansion:

$$\frac{4}{\pi} = 1 + \cfrac{1^2}{2 + \cfrac{3^2}{2 + \cfrac{5^2}{2 + \cfrac{7^2}{2 + \cdots}}}}$$

Gregrory's series:

$$\frac{\pi}{4} = 1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \frac{1}{9} - \cdots$$

Newton's series:

$$\frac{\pi}{6} = \frac{1}{2} + \frac{1}{2 \cdot 3 \cdot 2^3} + \frac{1 \cdot 3}{2 \cdot 4 \cdot 5 \cdot 2^5} + \cdots$$

Sharp's series:

$$\frac{\pi}{6} = \frac{1}{\sqrt{3}}\left(1 - \frac{1}{3^1 \cdot 3} + \frac{1}{3^2 \cdot 5} - \frac{1}{3^3 \cdot 7} + \cdots\right)$$

Euler's series:

$$\frac{\pi^2}{6} = \frac{1}{1^2} + \frac{1}{2^2} + \frac{1}{3^2} + \frac{1}{4^2} + \frac{1}{5^2} + \cdots$$

$$\frac{\pi^2}{8} = \frac{1}{1^2} + \frac{1}{3^2} + \frac{1}{5^2} + \frac{1}{7^2} + \frac{1}{9^2} + \cdots$$

$$\frac{\pi^2}{12} = \frac{1}{1^2} - \frac{1}{2^2} + \frac{1}{3^2} - \frac{1}{4^2} + \frac{1}{5^2} - \cdots$$

Pág. 6

Wallis' identity:

$$\pi = 2 \cdot \frac{2 \cdot 2 \cdot 4 \cdot 4 \cdot 6 \cdot 6 \cdots}{1 \cdot 3 \cdot 3 \cdot 5 \cdot 5 \cdot 7 \cdots}$$

Crie uma função que permita calcular o valor de PI usando a equação de Wallis.

Considere ós 1000 primeiros termos.

Assinatura:

```
def pi_wallis() -> float:
```

Wallis' identity:

$$\pi = 2 \cdot \frac{2 \cdot 2 \cdot 4 \cdot 4 \cdot 6 \cdot 6 \cdots}{1 \cdot 3 \cdot 3 \cdot 5 \cdot 5 \cdot 7 \cdots}$$

```python
def pi_wallis() -> float:
    produto = 2

    for i in range(1,1001):
        produto = produto * (i+i%2)/(i+1-i%2)

    return produto
```

Brouncker's continued fraction expansion:

$$\frac{4}{\pi} = 1 + \cfrac{1^2}{2 + \cfrac{3^2}{2 + \cfrac{5^2}{2 + \cfrac{7^2}{2 + \cdots}}}}$$

Crie uma função que permita calcular o valor de PI usando a equação de Wallis.

Considere ós 1000 primeiros termos.

Assinatura: `def pi_brouncker() -> float:`

Brouncker's continued fraction expansion:

$$\frac{4}{\pi} = 1 + \cfrac{1^2}{2 + \cfrac{3^2}{2 + \cfrac{5^2}{2 + \cfrac{7^2}{2 + \cdots}}}}$$

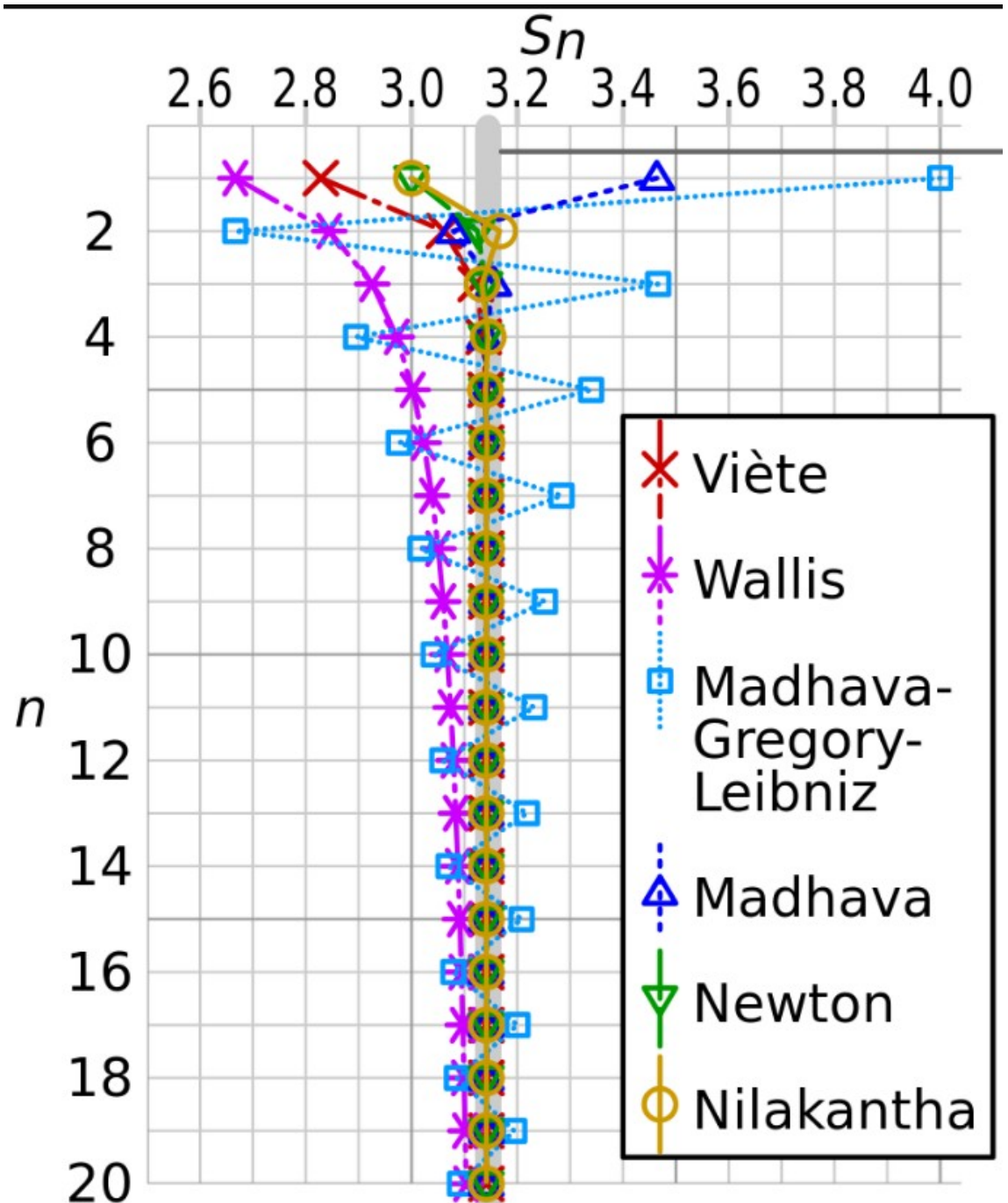
```
def pi_brouncker() -> float:
    den = 1

    for i in range(1000,0,-1):
        den = 2 + (i*2+1)**2 / den

    return 4/(1+1/den)
```

Melhor algoritmo?

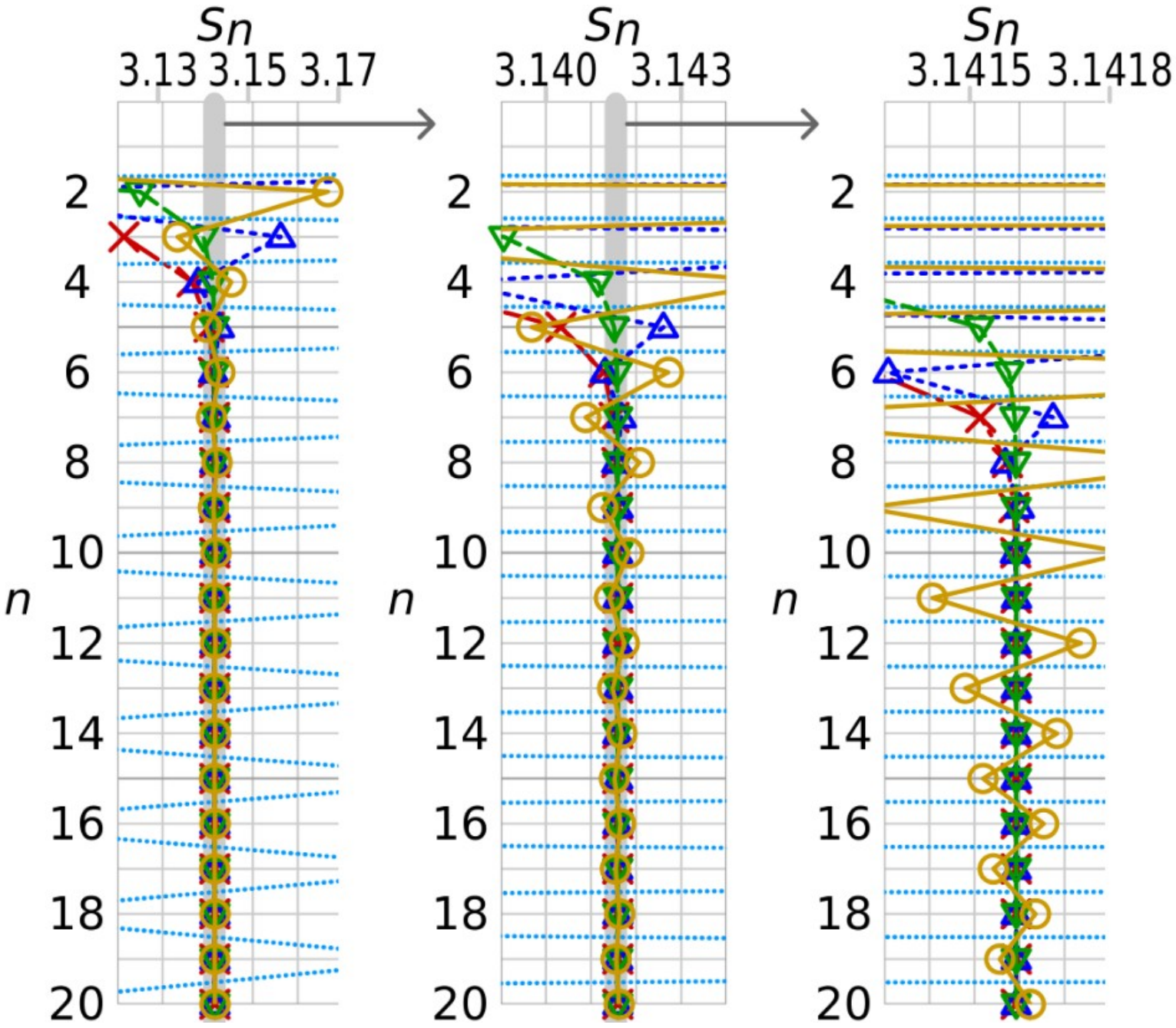
Qual seria o melhor algoritmo?

O que seria melhor?
- O mais rápido?
- O mais preciso?

Convergência



Convergência



Um pouco sobre expressões regulares

Procurando padrões textuais

- As expressões regulares (*Regular expression* – **RE ou ER**) podem ser utilizadas para **extrair trechos** a partir de texto.
- As ERs são comumente utilizadas para:
  - Normalizaçao de texto (e.g., padronizar o texto convenientemente)
  - Divisão em *tokens* (e.g., divisãfo em palavras usando os espaços?)
  - Radicalização (e.g., lemmatization, stemming)
  - Segmentação de frases (e.g., divisão em frases usando a pontuação)
- **O que são ERs?** Cadeias de texto especias, *em uma linguagem formal*, para busca/extração de trechos de texto.
- **Sinônimos:** Regex, Regexp.

Procurando padrões textuais

É o maior roedor do mundo, pesando até 91 kg e medindo até 1,2 m de comprimento e 60 cm de altura.

Procurando padrões textuais

Como podemos procurar por **qualquer** um dos seguintes nomes?

- capivara
- **C**apivara

É o maior roedor do mundo, pesando até 91 kg e medindo até 1,2 m de comprimento e 60 cm de altura.

Procurando padrões textuais

Como podemos procurar por **qualquer** um dos seguintes nomes?

- capivara
- **C**apivara

É o maior roedor do mundo, pesando até 91 kg e medindo até 1,2 m de comprimento e 60 cm de altura.

```
[cC]apivara
```

https://regex101.com/

https://pt.wikipedia.org/wiki/Capivara

1) Disjunções

- A cadeia de caracteres entre colchetes/parênteses retos (i.e, **'['** e **']'**) especifica uma **disjunção** de caracteres para a **busca**.

Padrão	Casamento
`[cC]apivara`	Capivara capivara
`[1234567890]`	Qualquer digito (mas apenas 1 digito)

1) Disjunções

- Faixas de caracteres

Padrão	Casamento	Exemplo
`[0-9]`	Apenas 1 digito	AED**1** : Algoritmos e Est...
`[a-z]`	Um caractere em minúscula	AED1 : A**l**goritmos e Est...
`[A-Z]`	Um caractere em maiúscula	**A**ED1 : Algoritmos e Est...
`[5-8]`	Apenas um digito: 5, 6, 7 ou 8.	AED1 : Algoritmos e Est...

2) Negação em Disjunções: ^

- O símbolo do acento circunflexo é utilizado para negar uma disjunção.
- Obs: A negação apenas válida como primeiro caractere.

Padrão	Casamento	Exemplo
`[^A-Z]`	Não um caractere em maiúscula	C**i**ência da computação
`[^Ss]`	Nem 'S' nem 's'	**A**ED1 : Algoritmos e Est...

2) Negação em Disjunções: ^

Padrão	Casamento	Exemplo
[^a^]		^Ciência da computação...
a^b		O valor final é a^b com a...

2) Negação em Disjunções: ^

Padrão	Casamento	Exemplo
[^a^]	Nem ‘a‘ nem ‘^‘	^**C**iência da computação...
a^b	Padrão a^b	O valor final é **a^b** com a...

3) Mais sobre Disjunções: | (barra vertical)

- Capivara também é conhecido como Carpincho.

Padrão	Casamento
`[cC]apivara|[cC]arpincho`	
`eu|voce`	
`[a|b|c]`	

3) Mais sobre Disjunções: | (barra vertical)

- Capivara também é conhecido como Carpincho.

Padrão	Casamento
`[cC]apivara|[cC]arpincho`	capivara Capivara carpincho Carpincho
`eu|voce`	eu voce
`[a|b|c]`	=[abc]

4) Outras opções: ? * + .

Padrão	Casamento	Exemplo
pa?ra	Caractere predecessor opcional	para pra
aa*h!	Caractere predecessor: 0 ou mais vezes	ah! aah! aaah! aaaaah!
o+h!	Caractere predecessor: 1 ou mais vezes	Oh! Ooh! Oooh!
ca.a		casa caza caça

4) Outras opções: ? * + .

Padrão	Casamento
`[ab]*`	
`[0-9][0-9]*`	

4) Outras opções: ? * + .

Padrão	Casamento
`[ab]*`	aaaaaaa abababaa aaaabbbbb baaaaaaaa bbbbbbbbbbbb
`[0-9][0-9]*`	Um número de pelo menos um digito. Por que não considerarmos apenas [0-9]* ?

5) Caracteres âncoras: ^ $

- Início de linha: ^ Final de linha: $

Padrão	Casamento
`^[A-Z]`	**S**anto André
`a$`	Cas**a**
`\.$`	Casa**.**
`^A UFABC\.$`	Uma linha contendo exatamente `A UFABC.`

O caractere '\' serve para indicar que o caractere sucessor não é especial.

Exemplos

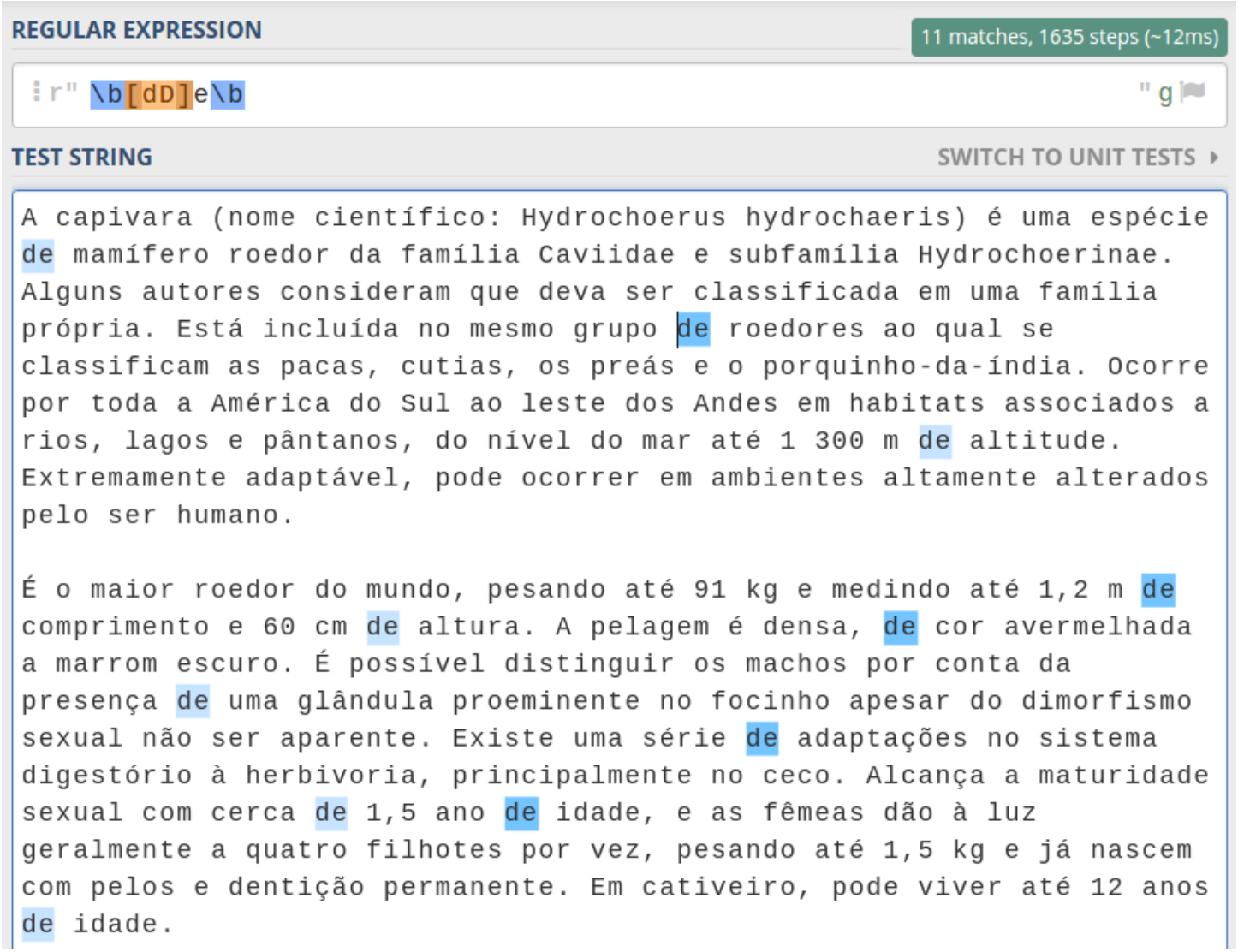
- Procurar pelas instâncias da palavra: **de**

```
[dD]e
```

Regular Expression Syntax

Expression	Description	Examples and Expansions
	Single character expressions	
`.`	any single character	`spi.e` matches "spice", "spike", etc.
`\char`	for a nonalphanumeric *char*, matches *char* literally	`\*` matches "*"
`\n`	newline character	
`\r`	carriage return character	
`\t`	tab character	
`[...]`	any single character listed in the brackets	`[abc]` matches "a", "b", or "c"
`[...-...]`	any single character in the range	`[0-9]` matches "0" or "1" ... or "9"
`[^...]`	any single character not listed	`[^sS]` matches one character that is neither "s" nor "S"
`[^...-...]`	any single character not in the range	`[^A-Z]` matches one character that is not an uppercase letter
	Anchors/Expressions which match positions	
`^`	beginning of line	
`$`	end of line	
`\b`	word boundary	`nt\b` matches "nt" in "paint" but not in "pants"
`\B`	word non-boundary	`all\B` matches "all" in "ally" but not in "wall"
	Counters/Expressions which quantify previous expressions	
`*`	zero or more of previous r.e.	`a*` matches "", "a", "aa","aaa",...
`+`	one or more of previous r.e.	`a+` matches "a", "aa", "aaa",...
`?`	exactly one or zero of previous r.e.	`colou?r` matches "color" or "colour"
`{n}`	*n* of previous r.e.	`a{4}` matches "aaaa"
`{n,m}`	from *n* to *m* of previous r.e.	
`{n,}`	at least *n* of previous r.e.	
`.*`	any string of characters	
`(...)`	grouping for precedence and memory for backreference	
`...|...`	matches either of neighbor r.e.s	`(dog)|(cat)` matches "dog" or "cat"
	Shortcuts	
`\d`	any digit	`[0-9]`
`\D`	any non-digit	`[^0-9]`
`\w`	any alphanumeric/underscore	`[a-zA-Z0-9_]`
`\W`	any non-alphanumeric	`[^a-zA-Z0-9_]`
`\s`	whitespace (space, tab)	`[␣\r\t\n\f]`
`\S`	non-whitespace	`[^␣\r\t\n\f]`

https://regex101.com/



Um exemplo prático: Eliza

Sobre Eliza (chatterbot)

- Foi um dos primeiros sistemas de PLN / MIT:1964-1966

Weizenbaum, Joseph. (1966). ***ELIZA – A computer program for the study of natural language communication between man and machine.*** *Communications of the ACM, 9(1), 36–45.*

- Programado para manter conversa bem '*limitada*' (superficial) com um usuário, **imitando** respostas.
- Basicamente são **procurados padrões textuais** na conversa.
- A resposta de Eliza seria uma **substituição** desses padrões, com a finalidade de criar uma **ilusão de entendimento** da conversa.

Sobre Eliza (chatterbot)

- No sistema são procuradas frases como por exemplo:

"Você está **X**"

- Sendo substituidas

"O que faz pensar que eu estou **X?**"

- Simulação de um terapeuta:

→ As respostas são um espelho do pensamento do paciente.

→ Na tentativa de os mesmos pacientes descobram os motivos pelos quais eles gostariam de mudar.

Sobre Eliza (chatterbot)

- **Pessoa**: Os pais são todos iguais!!!
- **Eliza**: DE QUE FORMA?

- **Pessoa**: Eles estão sempre nos incomodando sobre algo
- **Eliza**: VOCÊ PODE PENSAR EM UM EXEMPLO ESPECÍFICO?

- **Pessoa**: Olha só, meus pais me fizeram vir aqui
- **Eliza**: SEUS PAIS FIZERAM VIR AQUI

- **Pessoa**: Eles falam que estou deprimido há muito tempo
- **Eliza**: LAMENTO OUVIR QUE VOCÊ ESTÁ DEPRIMIDO

Proposta 'simples' em Python

```python
import re

patterns = [
    (r"\b(to|estou|eu estou)\b", "EU ESTOU"),
    (r"\b(sou|eu sou)\b", "EU SOU"),
    (r"\b(olha so,?) ", ""),
    (r".*EU ESTOU (deprimido|triste|cansado).*", r"LAMENTO OUVIR QUE VOCE ESTA \1"),
    (r".*EU SOU (infeliz).*", r"POR QUE VOCE ESTA \1?"),
    (r".* todos .*", "DE QUE FORMA?"),
    (r".* sempre .*", "VOCE PODE PENSAR EM UM EXEMPLO ESPECIFICO?"),
    ]

while True:
    comment = input()
    response = comment.lower()
    for pat, sub in patterns:
        response = re.sub(pat, sub, response)
    print (response.upper())
```

Chat bot comparative table

https://docs.google.com/spreadsheets/d/1RgG-dRS42EHlG7QdJOTg2ZO587KutTTPeUfyxVKoIn8/edit#gid=0

Bot Name	Platform	Features	Programming languages / Apps / Integration	Technical details	License	Languages	Project Link	Channels	Clients/Fields	More information
IBM Watson Conversation Service	-	Built on a neural network (one billion Wikipedia words). Has three main components: Intents, Entities, Dialog	Node SDK Java SDK Python SDK iOS SDK Unity SDK	-	Free Standard Premium https://www.ibm.com/watson/developercloud/conversation.html#pricing-block	English, Japanese	https://www.ibm.com/watson/developercloud/conversation.html	speech image text	Healthcare, Finance, Legal, Retail, Fantasy Football	https://www.youtube.com/watch?v=1rTI1WEbg5U
AgentBot	Aivo's own natural language processing technology.	Understands natural language. Memory to maintain coherence during long conversations. Gathers customer information to deliver customized solutions. Continuous evolution. Clarifies intent.	Use our REST API to integrate with your CRM and other platforms.	Integrates with any CRM, internal system, human chat and third party application.	http://agentbot.net/en/request-a-demo/	English, Spanish, Portuguese	http://agentbot.net/en/	voice or messenger channel	Telecommunications and Cable Operators E-Commerce and Online Services Banks and Financial Services Government	https://www.youtube.com/watch?v=KEHMP6TkbSU
Twyla	A proprietary AI platform.	Learns from agent/customer live chats. Blends machine learning and rule-based methods. Answers questions, deflects tickets.	Analyses data either via the API of your helpdesk or chat solution, or a secure file upload.	Is integrated with most major cloud Helpdesk and Live Chat solutions, like Zendesk, Salesforce and Liveperson. So no new processes or software.	Twyla is Software as a Service (AI-as-a-Service) and so comes at a monthly subscription cost, with no up-front setup or installation fees.	English	https://www.twylahelps.com/	Web Facebook Telegram Through messenger apps and live chat	Automation and self-service customer support.	https://www.youtube.com/watch?v=An4UmvqAx0Q
Pypestream	Pypestream's Smart Messaging Platform.	Pypestream uses a patented framework of 'Pypes' and 'Streams'. Natural Language Processing and keyword parsing.	The Smart Messaging Framework Pypeconnect SDK Pypemanager The Pypestream mobile app API plug-ins and integratios	An open and flexible API platform allows for custom integrations and development of 3rd party connectors, plugins and extensions.	Contact Pypestream to obtain current pricing.	English	https://www.pypestream.com/	Pypestream mobile app Brand apps/SMS Web chat Messenger IoT website Pype	By April 2016, the company had 500 businesses signed up and using the messaging platform, including Washington Gas and Billboard.	https://www.youtube.com/watch?v=nfPxUXx7l2w

- Na atualidade existem diferentes sistemas de chat desenvolvidos com técnicas sofisticadas para atendimento, por exemplo, para clientes de um negócio.
- Certamente Eliza foi um sistema pioneiro (baseado em busca de padrões)