Aula 7

Aula passada

- Método de Monte Carlo
- Estimando somatórios
- Calculando erro
- Estimando π
- Erro de π
- Integração de Monte Carlo
- Monte Carlo Ray Tracing

Aula de hoje

- Gerando amostras de v.a. discretas
- Gerando Geométrica
- Método da transformada inversa
- Gerando Binomial
- Gerando permutações

Gerando Amostras

- Aula passada usamos apenas gerador uniforme
- Como gerar amostras de v.a. não uniforme

Usar a uniforme!

- **Premissa:** gerador de v.a. uniforme contínua [0,1] disponível

Premissa é (ligeiramente) falsa!

1) número não é contínuo: representação discreta de números no computador (ex. 32 bits)

2) gerador não é aleatório: computador é determinístico (um algoritmo gera o número aleatório)

Gerador Pseudo-aleatório!

Gerador Pseudo-Aleatório

- Problema 1 é contornado usando maior precisão
  - ex. 64 bits divide intervalo [0,1] em 2^{64} pedacinhos
- Problema 2 é contornado usando algoritmos que misturam os bits com manipulações algébricas
  - geram sequência de números no qual o próximo depende do anterior (ou anteriores)
  - passam em muitos testes estatísticos para aferir aleatoriedade
  - ex. Mersenne-Twister (1997) um dos mais usados algoritmos para geração de números pseudo-aleatório

Premissa é falsa na teoria
mas contornável na prática!

Gerando Outras Distribuições

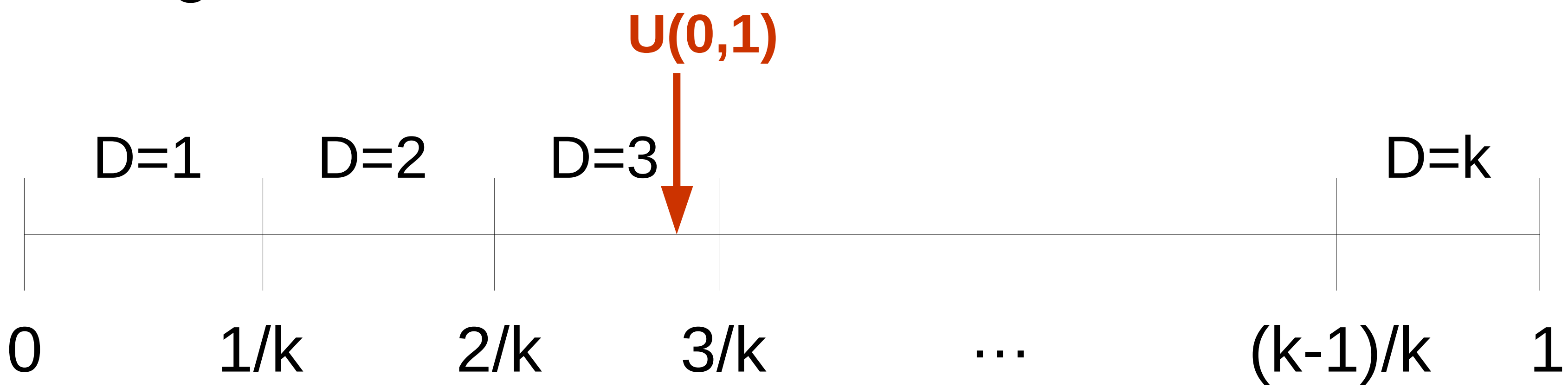
- Assumindo gerador de amostras de uma v.a. uniforme contínua [0,1]
  - U~unif(0,1)

Como gerar v.a. com outras distribuições?

- Ex. como gerar uma amostra de uma v.a. geométrica com parâmetro *p* ?
- Muitas técnicas e algoritmos
  - diferentes complexidade e aplicabilidade
- Melhor abordagem em geral depende da distribuição a ser gerada

Lançando um Dado

- Considere um dado honesto com k faces
- Seja D o valor da face ao lançar o dado
  - P[D = i] = 1/k
- Como gerar valor da face?



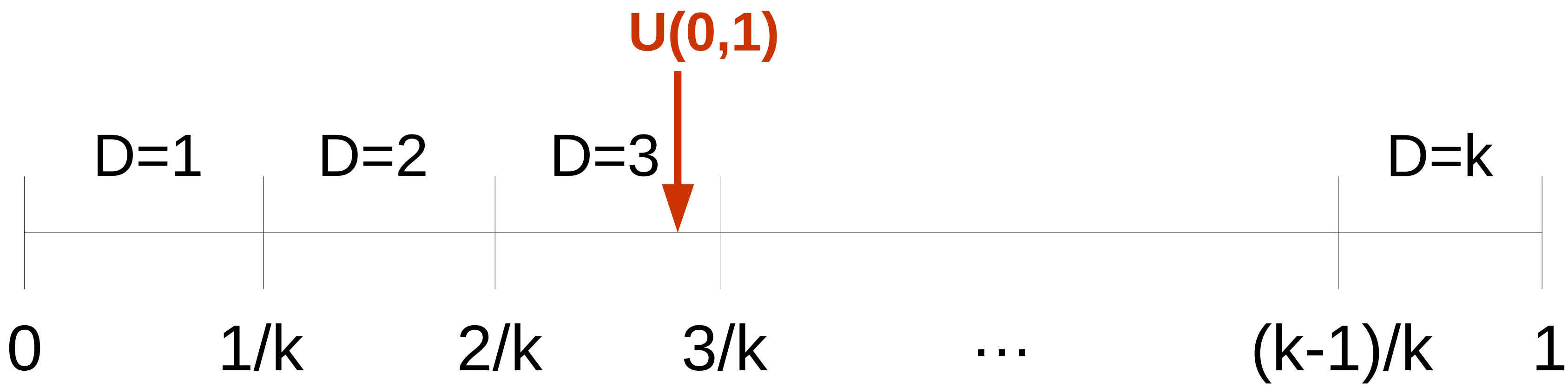
- Gerar U. Determinar *i*, tal que (i-1)/k < U <= i/k

```
U = unif(0,1); i=1;
while(! ((i-1)/k < U <= i/k)) i++;
return i;
```

- Complexidade?

Lançando um Dado

- Podemos tirar proveito que o dado é honesto



- Algoritmo mais eficiente, determina *i* diretamente

  U = unif(0,1);

  i = int(kU) + 1; ← int(r): retorna parte inteira de r

  return i;

- Complexidade?

Lançando outro Dado

- Considere que dado com k faces não é honesto
- Seja D o valor da face ao lançar o dado
  - prob. da face *i* é proporcional a w_i

$$P[D=i]=\frac{w_i}{W} \qquad W=\sum_{i=1}^{k} w_i$$

- Como gerar valor da face? Generalização do algoritmo inicial resolve!
- Gerar U. Determinar i, tal que

$$\sum_{j=1}^{i-1} w_i < WU \leq \sum_{j=1}^{i} w_i$$

```
U = unif(0,1); s = 0; i=1;
while (s <= WU) s += w[i];  i++;
return i;
```

- Complexidade?

Lançando Dado Eficiente

$$P[D=i]=\frac{w_i}{W} \qquad W=\sum_{i=1}^{k} w_i$$

- Considere w_i inteiro. Como gerar v.a. de forma mais eficiente? Em O(1)?
- **Ideia:** pré-processamento
  - 1) alocar um vetor de tamanho W
  - 2) preencher w_i posições com valor *i*
- Para escolher face do dado
  - 1) escolher uniforme inteiro em [1, W]
  - 2) acessar vetor para retornar face
- Complexidade? Memória?
- Truque muito usado!

Alias Method

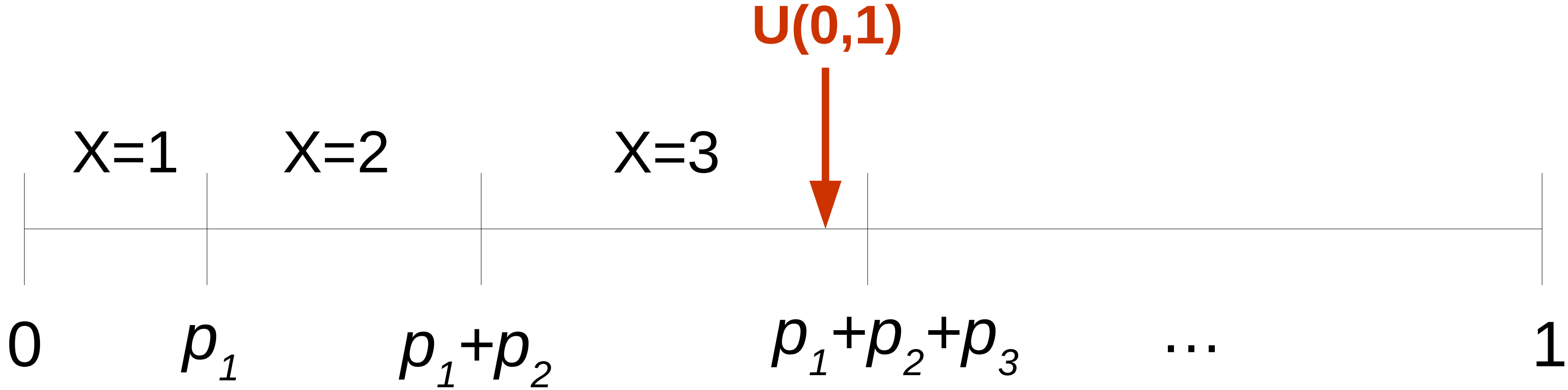
- Método eficiente para gerar D
  - não assume w_i inteiros, mas precisa de W
  - usa memória Θ(k), dois vetores
- Pré-processamento (criação dos vetores)
  - O(k log k) ou O(k), dependendo do método
- Geração de número aleatório usando vetores
  - O(1) – duas escolhas uniformes
- Muito usado para acelerar geração de sequência iid com muitas opções em cada escolha
  - ex. escolher aluno da UFRJ com probabilidade proporcional ao CRA, w_i

Gerando Geométrica

- Seja *X* uma v.a. com distribuição geométrica, *p*

$$p_i = P[X = i] = (1-p)^{i-1} p, i = 1,2,\ldots$$

- Como gerar valores para *X*?
- Abordagem 1: adaptação do anterior (sem limite superior para valor que v.a. pode assumir)



- Complexidade? Caso médio é O(1/*p*)

Gerando Geométrica

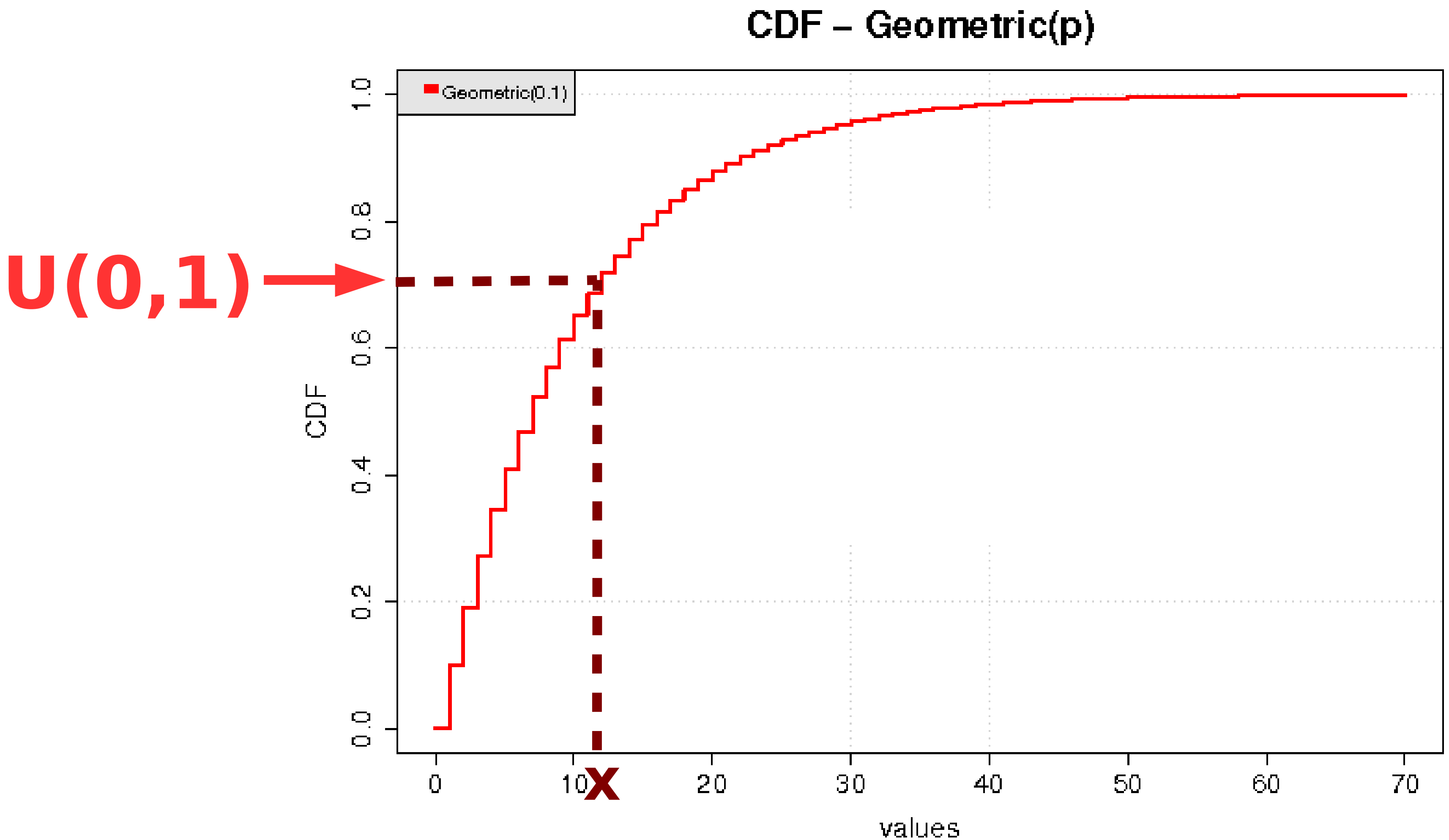
- Abordagem 2: Geométrica é sequência de Bernoulli até ocorrência de positivo
- Gerar uma sequência de Bernoulli(p) até observar evento positivo

```
e = 0; c = 1;
while (! e) if (unif(0,1) <= p) e=1 else c+=1;
return c;
```

- Vantagem: não calcula valor p_i da Geométrica
- Desvantagem: gera muitas uniformes
- Complexidade?

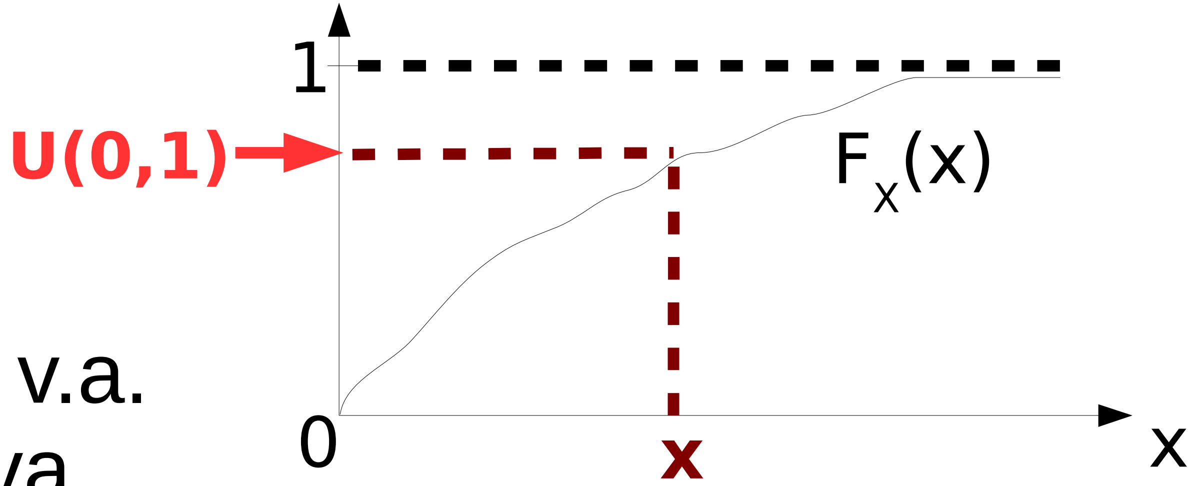
Gerando Geométrica

- Abordagem 3: Usar a inversa da função de distribuição cumulativa, $F_X(x)$



- Retornar $F_X^{-1}(\text{unif}(0,1))$. Complexidade?

Método da Transformada Inversa



- Seja U~unif(0,1), e X v.a. com função cumulativa $F_X(x)$ = P[X <= x]
- Temos que

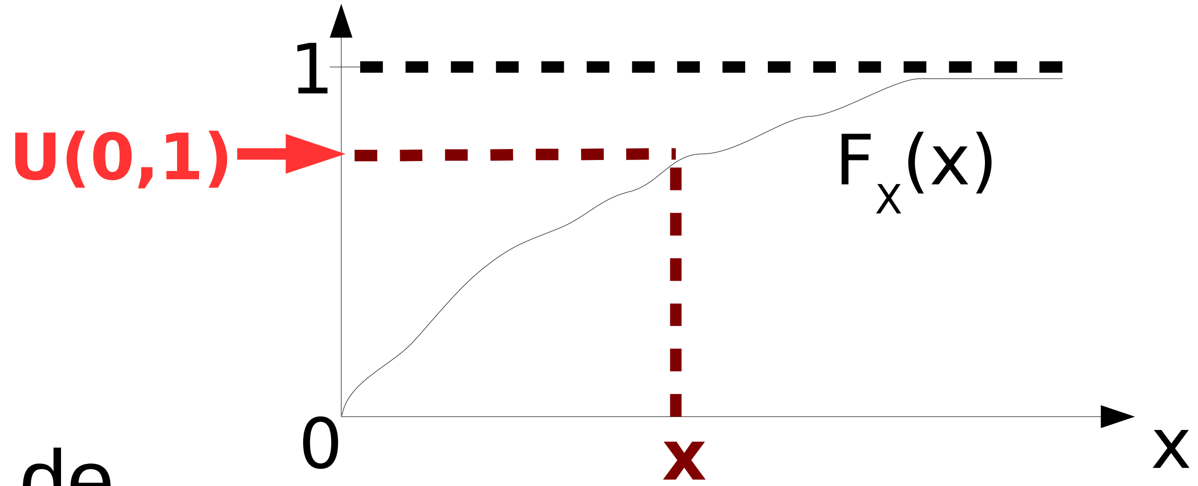
$$X = F_X^{-1}(U)$$

- onde F_X^{-1} é a inversa de F_X (valor de x para qual $F_X(x)$ = u)
- Podemos usar a inversa para gerar amostras!

Método da Transformada Inversa

- Prova da equivalência

$$X = F_X^{-1}(U)$$



- Mostrar que X acima de fato tem distribuição $F_X(x)$

$$P[X \leq x] = P[F_X^{-1}(U) \leq x]$$ ← por definição de X

$$= P[F_X(F_X^{-1}(U)) \leq F_X(x)]$$ ← Equivalência de eventos e monotonicidade de F_X

$$= P[U \leq F_X(x)]$$

$$= F_X(x)$$ ← Prob. da uniforme ser menor que β é igual a β (β em [0,1])

- Mas temos que obter a função inversa!

Inversa da Geométrica

- Seja *X* uma v.a. com distribuição geométrica, *p*

$$F_X(i) = P[X \leq i] = 1 - (1-p)^i, i = 1,2,\ldots$$

$$F_X^{-1}(u) = ?$$

$$1 - (1-p)^i = u$$

$$\rightarrow (1-p)^i = 1 - u$$

$$\rightarrow i = \frac{\log(1-u)}{\log(1-p)}$$

Se u é unif(0,1) então 1-u também é unif(0,1)

$$F_X^{-1}(u) = integer\left(\frac{\log u}{\log(1-p)}\right) + 1$$

i é inteiro, maior que zero

- Gerar u~unif(0,1) e aplicar fórmula acima
- Complexidade: O(1)

Gerando Binomial

- Seja X~Binom(N,p)

$$F_X(i)=P[X\leq i]=\sum_{k=0}^{i}\binom{N}{k}p^k(1-p)^{N-k}$$

- Não temos como inverter analiticamente F_X
- Alternativas?
- Gerar sequência iid de N Bernoulli(p), contar quantos eventos foram positivos
- **Problema:** p muito baixo, N grande, teremos muitos zeros
  - ex. $p=10^{-3}$, $N=10^5$
- **Ideia:** Usar geométrica para acelerar geração de 1

Gerando Binomial

- Sequência de N v.a. Bernoullis(p) iid
  00000010001010100000000100000010001000000000100
- Distribuição da posição do primeiro valor 1? ← Geométrica(p)
- Distribuição da posição do segundo valor 1? ← Geométrica(p), a partir do primeiro valor 1!
- Número de 1 na sequência de tamanho N é igual ao número de geométricas que somadas ocorreram antes de N

$$X = k \mid min_k \sum_{i=1}^{\infty} G_i \geq N$$ ← onde G_i~Geom(p)

- Algoritmo: gerar sequência iid Geom(p) até soma ser >= N
- Complexidade? Quantas geométricas serão geradas, na média?
- N/(1/p) = Np

Gerando Permutações

- Aula passada: gerar uma permutação da cartas do baralho, com probabilidade uniforme
- **Ideia 1**
  - Gerar *i* uniforme entre 1 e 52! (todas permutações)
  - Retornar a *i*-ésima permutação
- Problema? ← Como saber qual é a *i*-ésima permutação?
- **Ideia 2**
  - Escolher um elemento por vez de forma uniforme (sem repetição)
  - Fazer n-1 escolhas sucessivas
- Problema? ← Como fazer escolhas de forma eficiente?

Gerando Permutações

- **Ideia:** usar um vetor para alocar elementos e as escolhas realizadas (conhecido por *Knuth Shuffle*)
- Algoritmo eficiente para gerar uma permutação de N elementos de forma uniforme

```
vetor permuta[1,…,N];
permuta[i] = i para i=1,…,N;
para i=0 até N-1
    j = unif(1, N-i);        ← O(1)
    tmp = permuta[j];
    permuta[j] = permuta[N-i];
    permuta[N-i] = tmp;
```

- Funciona?
- Complexidade?