

ALGORITMOS

MÉTODO GULOSO

COPYRIGHT © 2023 DIATINF/CNAT/IFRN
JORGIANO VIDAL



- Método guloso
- Problema do troco
- Agendamento de atividades
- Comentários
- Exemplo: Par de soma s





MÉTODO GULOSO





- Contrói solução iterativamente
 - Melhor opção local





- Contrói solução iterativamente
 - Melhor opção local
- Solução ótima local **deve ser** ótima global
 - **Propriedade gulosa**





- Contrói solução iterativamente
 - Melhor opção local
- Solução ótima local **deve ser** ótima global
 - **Propriedade gulosa**
- **Irreversível**
 - Escolha feita **nunca é desfeita**
 - Diferente de *backtracking*





- Contrói solução iterativamente
 - Melhor opção local
- Solução ótima local **deve ser** ótima global
 - **Propriedade gulosa**
- **Irreversível**
 - Escolha feita **nunca é desfeita**
 - Diferente de *backtracking*
- Simples
 - **Fácil** de implementar





- Contrói solução iterativamente
 - Melhor opção local
- Solução ótima local **deve ser** ótima global
 - **Propriedade gulosa**
- **Irreversível**
 - Escolha feita **nunca é desfeita**
 - Diferente de *backtracking*
- Simples
 - **Fácil** de implementar
- Eficiente
 - Soluções **rápidas**





PROBLEMA DAS MOEDAS





PROBLEMA DAS MOEDAS

- Quantidade mínima de moedas para valor v

- Moedas = $\{5, 2, 1\}$





PROBLEMA DAS MOEDAS

- Quantidade mínima de moedas para valor v

- Moedas = $\{5, 2, 1\}$



- Exemplo: R\$ 0,07

- Duas moedas:

5 e 2 centavos





PROBLEMA DAS MOEDAS

- Quantidade mínima de moedas para valor v

- Moedas = $\{5, 2, 1\}$



- Exemplo: R\$ 0,07

- Duas moedas:

- 5 e 2** centavos



- Propriedade gulosa

- **Maior moeda possível sempre escolhida**





PROBLEMA DAS MOEDAS



PROBLEMA DAS MOEDAS

- **CUIDADO!!!**



- **CUIDADO!!!**
 - Algoritmo guloso pode depender dos dados

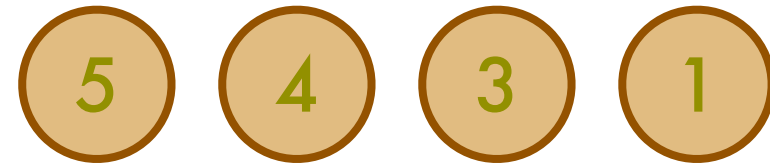


PROBLEMA DAS MOEDAS

- **CUIDADO!!!**

- Algoritmo guloso pode depender dos dados

- moedas = {5, 4, 3, 1}





PROBLEMA DAS MOEDAS

- **CUIDADO!!!**

- Algoritmo guloso pode depender dos dados

- `moedas = {5, 4, 3, 1}`



- `valor = R$ 0,07`

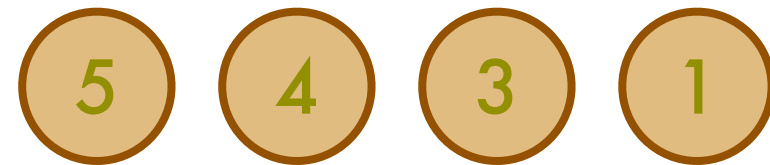


PROBLEMA DAS MOEDAS

- **CUIDADO!!!**

- Algoritmo guloso pode depender dos dados

- `moedas = {5, 4, 3, 1}`



- `valor = R$ 0,07`

- **Algoritmo guloso**



PROBLEMA DAS MOEDAS

- **CUIDADO!!!**

- Algoritmo guloso pode depender dos dados

- moedas = {5, 4, 3, 1}



- valor = R\$ 0,07

- **Algoritmo guloso**

- 3 moedas: 5, 1, 1



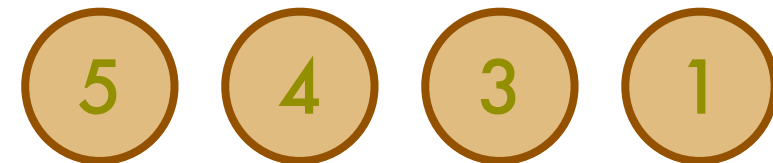


PROBLEMA DAS MOEDAS

- **CUIDADO!!!**

- Algoritmo guloso pode depender dos dados

- moedas = {5, 4, 3, 1}



- valor = R\$ 0,07

- **Algoritmo guloso**

- 3 moedas: 5, 1, 1



- **Resposta correta:**

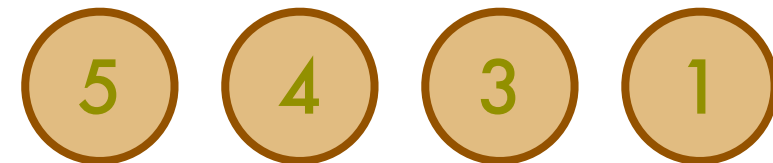


PROBLEMA DAS MOEDAS

- **CUIDADO!!!**

- Algoritmo guloso pode depender dos dados

- moedas = {5, 4, 3, 1}



- valor = R\$ 0,07

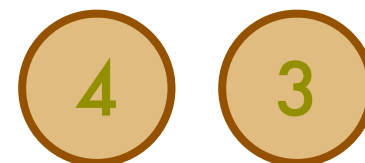
- **Algoritmo guloso**

- 3 moedas: 5, 1, 1



- **Resposta correta:**

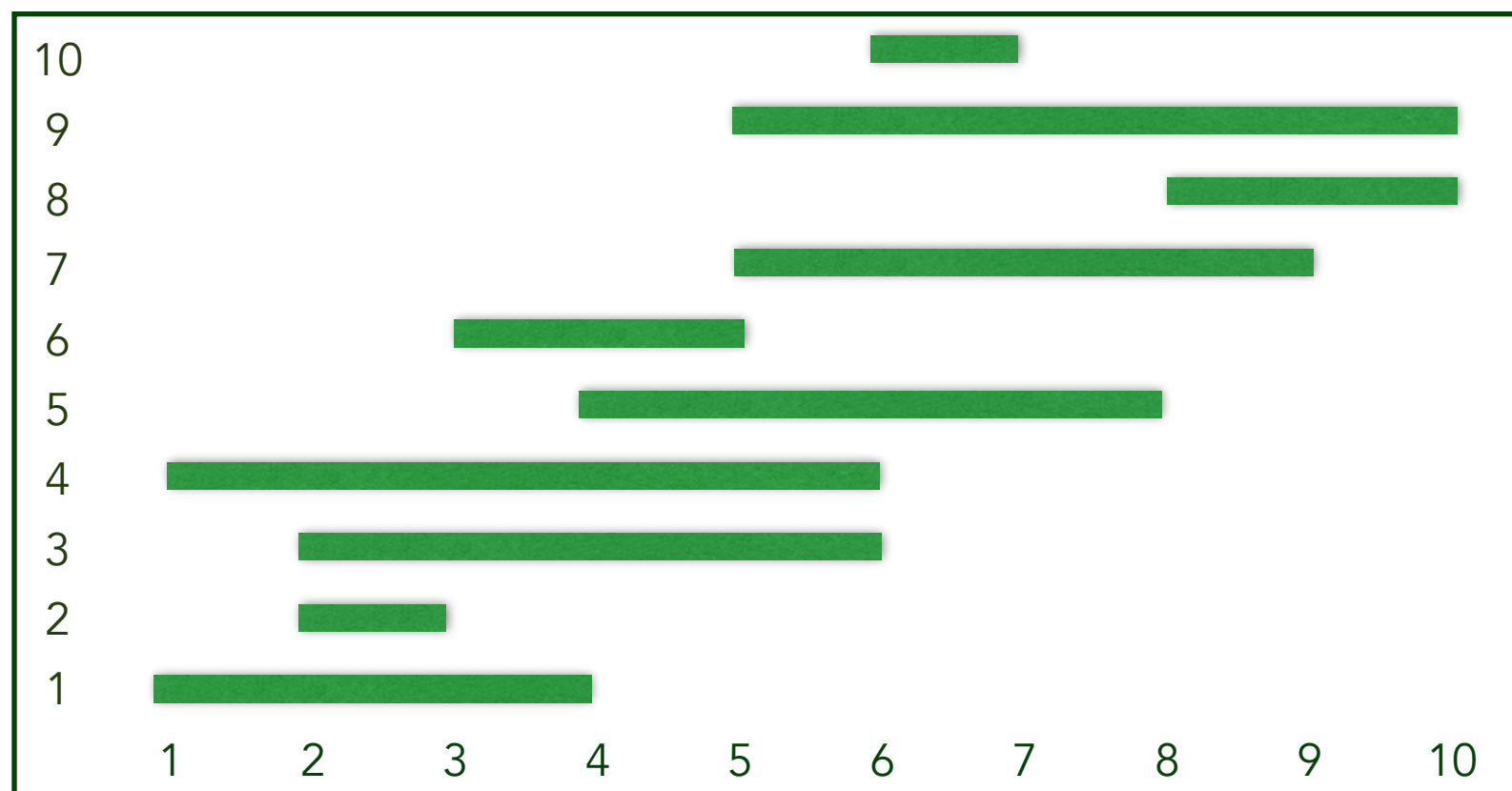
- 2 moedas: 4 e 3





ESCOLHA DE ATIVIDADES

- n atividades
- Hora início e hora fim
- Quais atividades escolher para realizar **maior quantidade** de atividades?

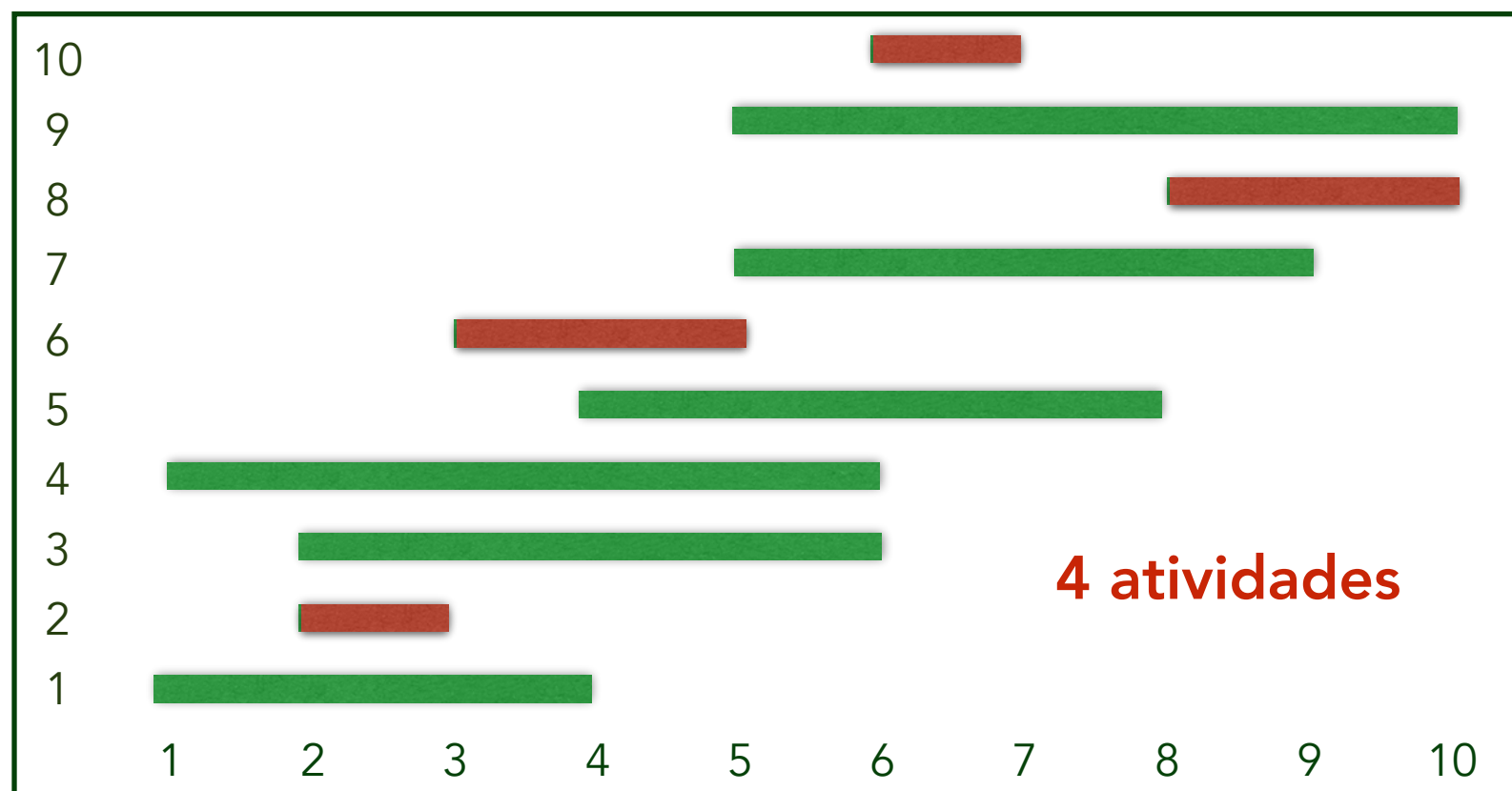


Atividade	Início	Fim
1	1	4
2	2	3
3	2	6
4	1	6
5	4	8
6	3	5
7	5	9
8	8	10
9	5	10
10	6	7

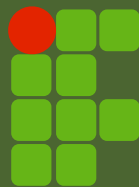


ESCOLHA DE ATIVIDADES

- n atividades
- Hora início e hora fim
- Quais atividades escolher para realizar **maior quantidade** de atividades?



Atividade	Início	Fim
1	1	4
2	2	3
3	2	6
4	1	6
5	4	8
6	3	5
7	5	9
8	8	10
9	5	10
10	6	7



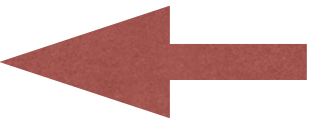
ESCOLHA DE ATIVIDADES



- Qual atividade escolher a cada iteração?
 - Início mais cedo?
 - Fim mais cedo?
 - Menor intervalo?
 - Menos conflito?

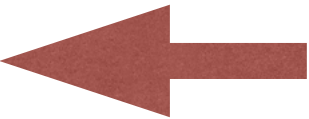


ESCOLHA DE ATIVIDADES

- Qual atividade escolher a cada iteração?
 - Início mais cedo?
 - Fim mais cedo? 
 - Menor intervalo?
 - Menos conflito?



ESCOLHA DE ATIVIDADES

- Qual atividade escolher a cada iteração?
 - Início mais cedo?
 - Fim mais cedo? 
 - Menor intervalo?
 - Menos conflito?
- Necessário ordenar:
 - **$O(n \lg_2 n)$**



ESCOLHA DE ATIVIDADES

- Qual atividade escolher a cada iteração?

- Início mais cedo?

- Fim mais cedo? 

- Menor intervalo?

- Menos conflito?

- Necessário ordenar:

- **$O(n \lg_2 n)$**

```
Seleciona_Atividades(a, s, f, n)
// Ordenar por  $f_i$ 
MergeSort(a, s, f, n)
A  $\leftarrow \{a_1\}$ 
i=1
Para m  $\leftarrow$  2 até n faça
    Se  $s_m \geq f_i$  então
        A  $\leftarrow$  A  $\cup \{a_m\}$ 
        i  $\leftarrow$  m
    Fim Se
Fim Para
retorne A
```



- Simples
- Fácil de implementar
- Eficiente
- **IMPORTANTE**
 - Análise dos dados deve ser feita
 - Alguma **ordem**, normalmente, é necessária
 - Pode ser necessário **ordenar**





- Par de soma s
 - Dado um vetor A de N números inteiros, existe um par de números cuja soma seja S ?



- Par de soma s
 - Dado um vetor A de N números inteiros, existe um par de números cuja soma seja S ?
- Força bruta:
 - Testa todos os pares: $C_2^n \rightarrow O(n^2)$



- Par de soma s
 - Dado um vetor A de N números inteiros, existe um par de números cuja soma seja S ?
- Força bruta:
 - Testa todos os pares: $C_2^n \rightarrow O(n^2)$
- Método guloso
 - Ordena



- Par de soma s
 - Dado um vetor A de N números inteiros, existe um par de números cuja soma seja S ?
- Força bruta:
 - Testa todos os pares: $C_2^n \rightarrow O(n^2)$
- Método guloso
 - Ordena
 - Compara primeiro e último



- Par de soma s
 - Dado um vetor A de N números inteiros, existe um par de números cuja soma seja S ?
- Força bruta:
 - Testa todos os pares: $C_2^n \rightarrow O(n^2)$
- Método guloso
 - Ordena
 - Compara primeiro e último
 - Se maior, descarta último



- Par de soma s
 - Dado um vetor A de N números inteiros, existe um par de números cuja soma seja S ?
- Força bruta:
 - Testa todos os pares: $C_2^n \rightarrow O(n^2)$
- Método guloso
 - Ordena
 - Compara primeiro e último
 - Se maior, descarta último
 - Se menor, descarta primeiro



- Par de soma s
 - Dado um vetor A de N números inteiros, existe um par de números cuja soma seja S ?
- Força bruta:
 - Testa todos os pares: $C_2^n \rightarrow O(n^2)$
- Método guloso
 - Ordena
 - Compara primeiro e último
 - Se maior, descarta último
 - Se menor, descarta primeiro
 - Repete operação até que não haja mais números



$S = 25$
10, 8, 6, 12, 9, 15, 2, 21



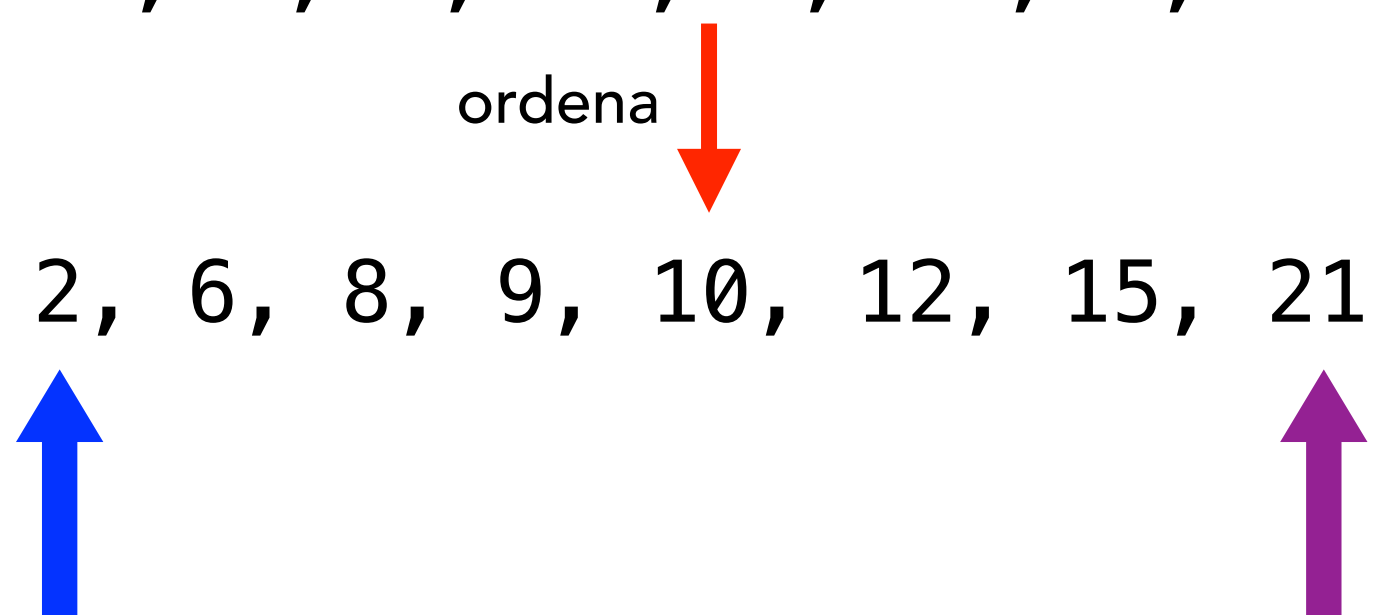
$S = 25$
10, 8, 6, 12, 9, 15, 2, 21
ordena



$S = 25$
10, 8, 6, 12, 9, 15, 2, 21
ordena
↓
2, 6, 8, 9, 10, 12, 15, 21



$S = 25$
10, 8, 6, 12, 9, 15, 2, 21
ordena
2, 6, 8, 9, 10, 12, 15, 21





$S = 25$
10, 8, 6, 12, 9, 15, 2, 21

ordena



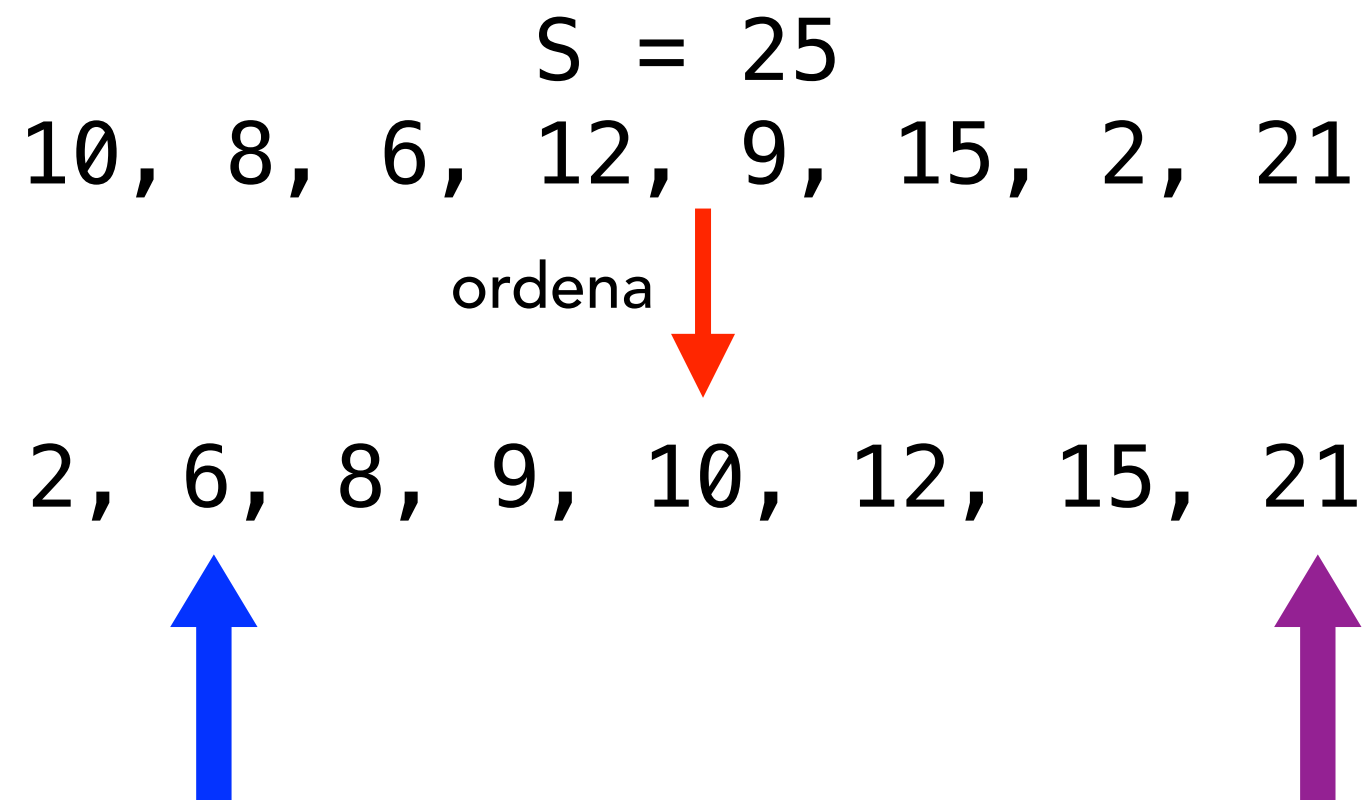
2, 6, 8, 9, 10, 12, 15, 21



$$2+21 < 25$$



$S = 25$
10, 8, 6, 12, 9, 15, 2, 21
ordena
2, 6, 8, 9, 10, 12, 15, 21



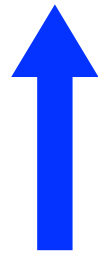


$S = 25$
10, 8, 6, 12, 9, 15, 2, 21

ordena



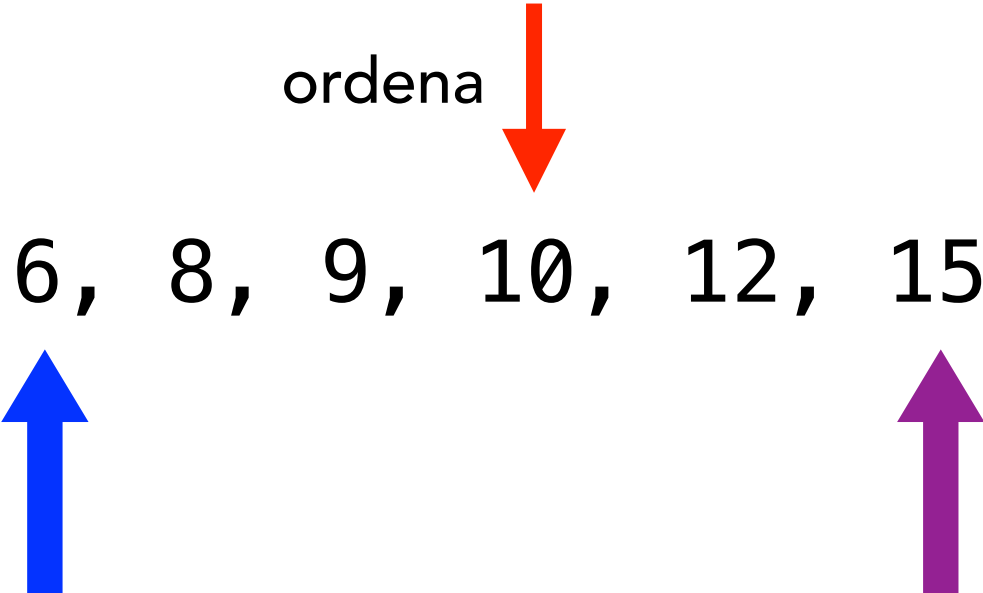
2, 6, 8, 9, 10, 12, 15, 21



$$6 + 21 > 25$$



$S = 25$
10, 8, 6, 12, 9, 15, 2, 21
ordena
2, 6, 8, 9, 10, 12, 15, 21





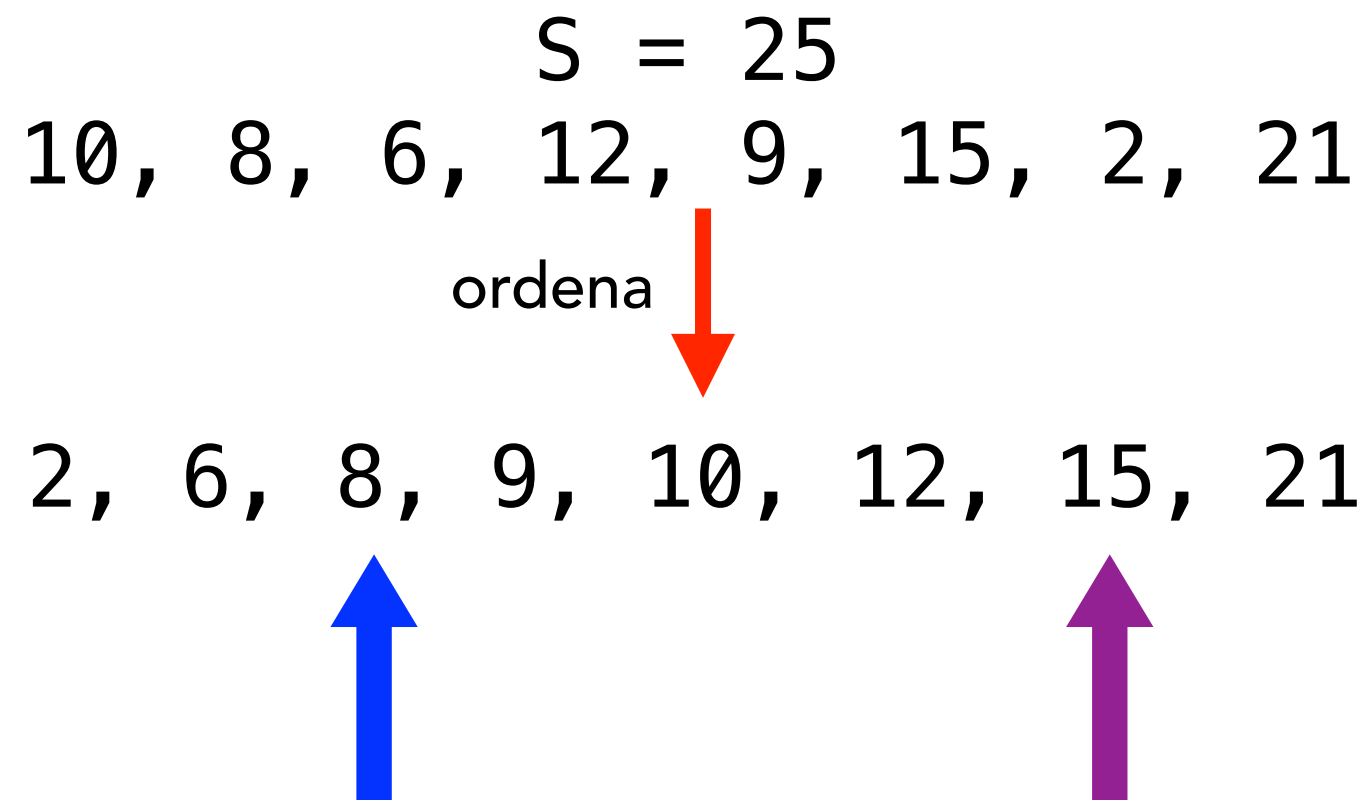
$S = 25$
10, 8, 6, 12, 9, 15, 2, 21

ordena


2, 6, 8, 9, 10, 12, 15, 21



$$6 + 15 < 25$$



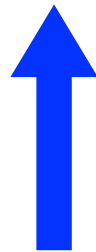


$S = 25$
10, 8, 6, 12, 9, 15, 2, 21

ordena



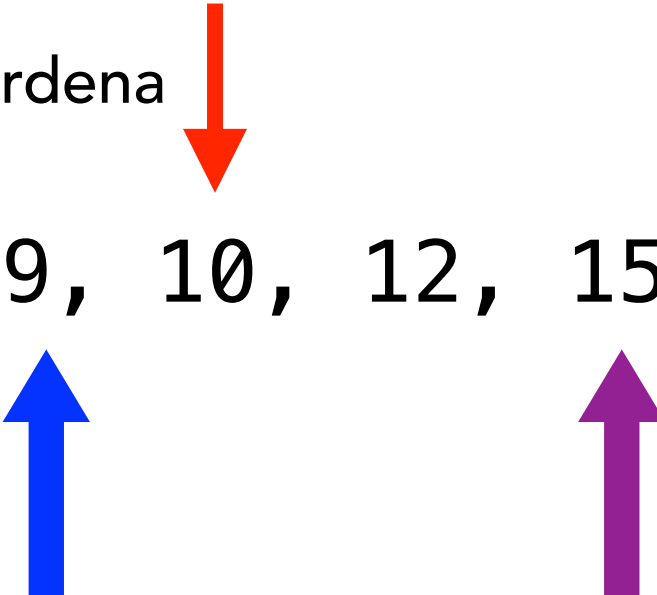
2, 6, 8, 9, 10, 12, 15, 21



$$8 + 15 < 25$$



$S = 25$
10, 8, 6, 12, 9, 15, 2, 21
ordena
2, 6, 8, 9, 10, 12, 15, 21





$S = 25$
10, 8, 6, 12, 9, 15, 2, 21

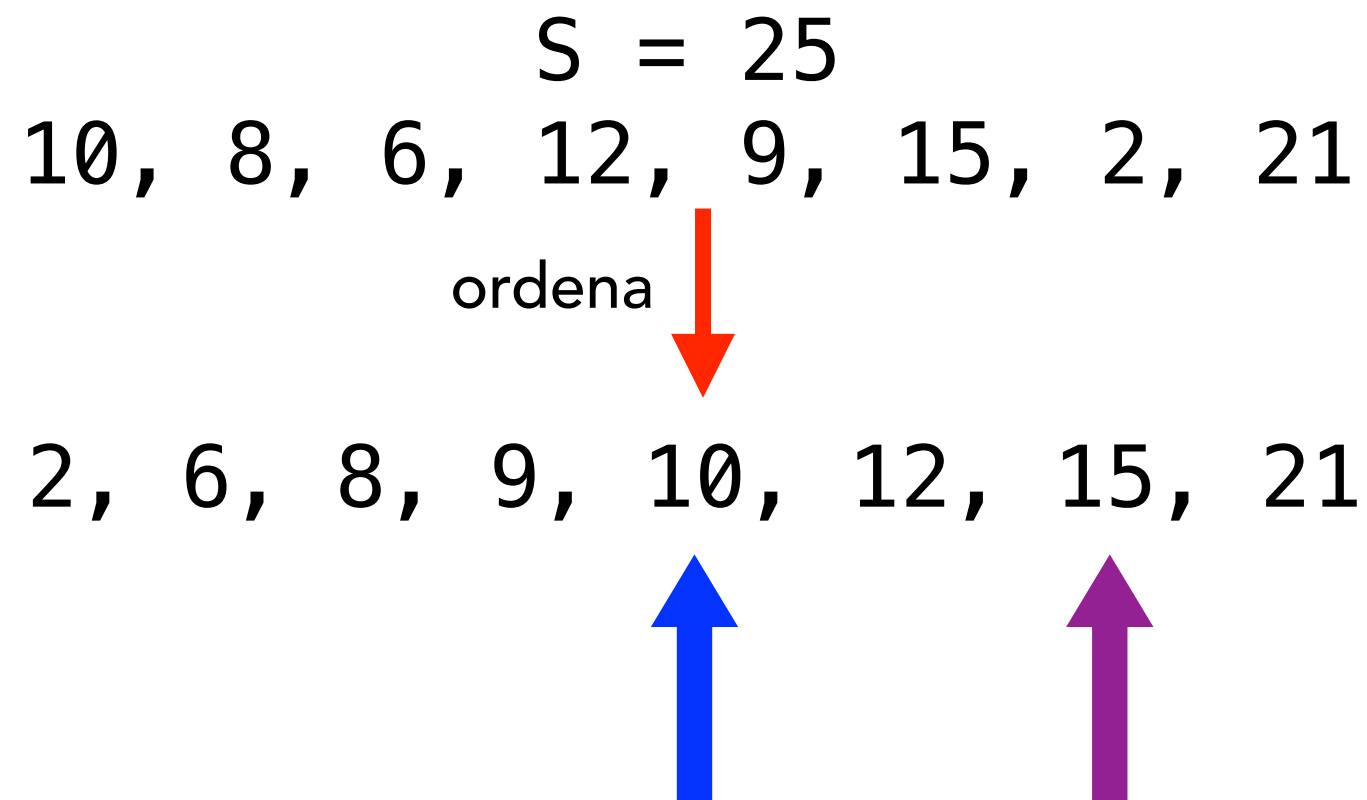
ordena



2, 6, 8, 9, 10, 12, 15, 21



$$9 + 15 < 25$$





$S = 25$
10, 8, 6, 12, 9, 15, 2, 21

ordena



2, 6, 8, 9, 10, 12, 15, 21



$$10 + 15 = 25$$



$S = 25$
10, 8, 6, 12, 9, 15, 2, 21

ordena



2, 6, 8, 9, 10, 12, 15, 21



$$10 + 15 = 25$$

$O(n)$



$S = 25$
10, 8, 6, 12, 9, 15, 2, 21

ordena

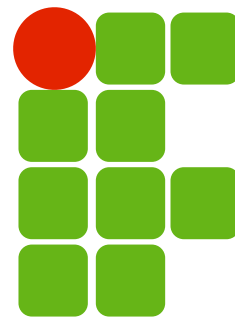

2, 6, 8, 9, 10, 12, 15, 21



$$10 + 15 = 25$$

$O(n)$

Implementação fica como exercício



ALGORITMOS

MÉTODO GULOSO

COPYRIGHT © 2023 DIATINF/CNAT/IFRN
JORGIANO VIDAL