Grupo 4

Estrutura de Dados I

Davi Brandão de Souza
Mauricio Zanetti Neto
Pedro Henrique Alves do Nascimento
Silvio Eduardo Bellinazzi de Andrade

18 de Agosto de 2025

Problema 1

Objetivo

- Verificar se uma dada expressão possui delimitadores balanceados;
- Considerar tanto o fechamento correto (a), assim como a hierarquia (b).

Exemplo:

{[(A+D)/B]*J} → Correto por (a) e (b)

([{A+D}/B]*J) → Correto por (a), incorreto por (b)

((A+D))*J) → Incorreto

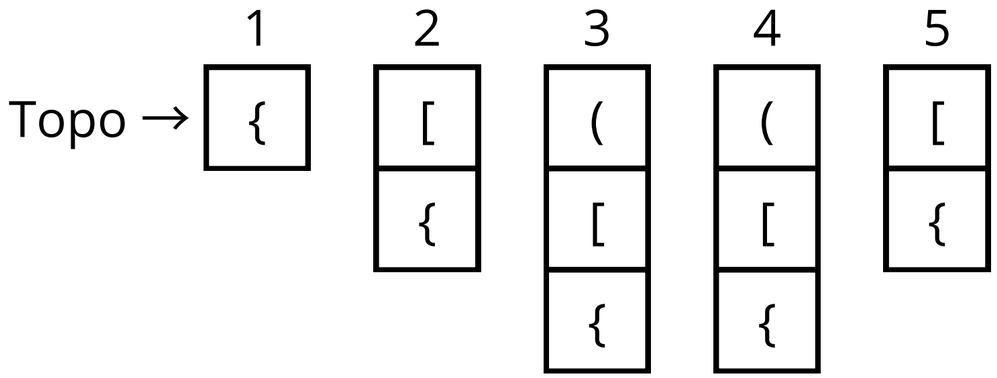
Problema 1

Lógica do algoritmo

Expressão: `{[(A+D)/B]*J}`

1. { → empilha
2. [→ empilha
3. (→ empilha
4. A, +, D → ignora
5.) → verifica se o topo é (→ desempilha

Pilha utilizada



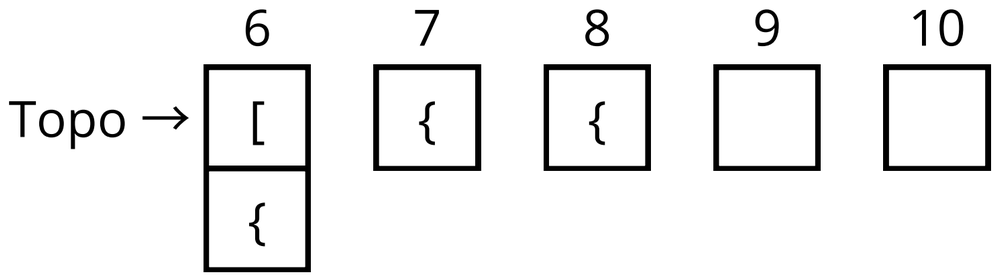
Problema 1

Lógica do algoritmo

Expressão: `{[(A+D)/B]*J}`

6. /, B → ignora
7.] → topo é [→ desempilha
8. *, j → ignora
9. } → topo é { → desempilha
10. Se a pilha está vazia → expressão válida

Pilha utilizada

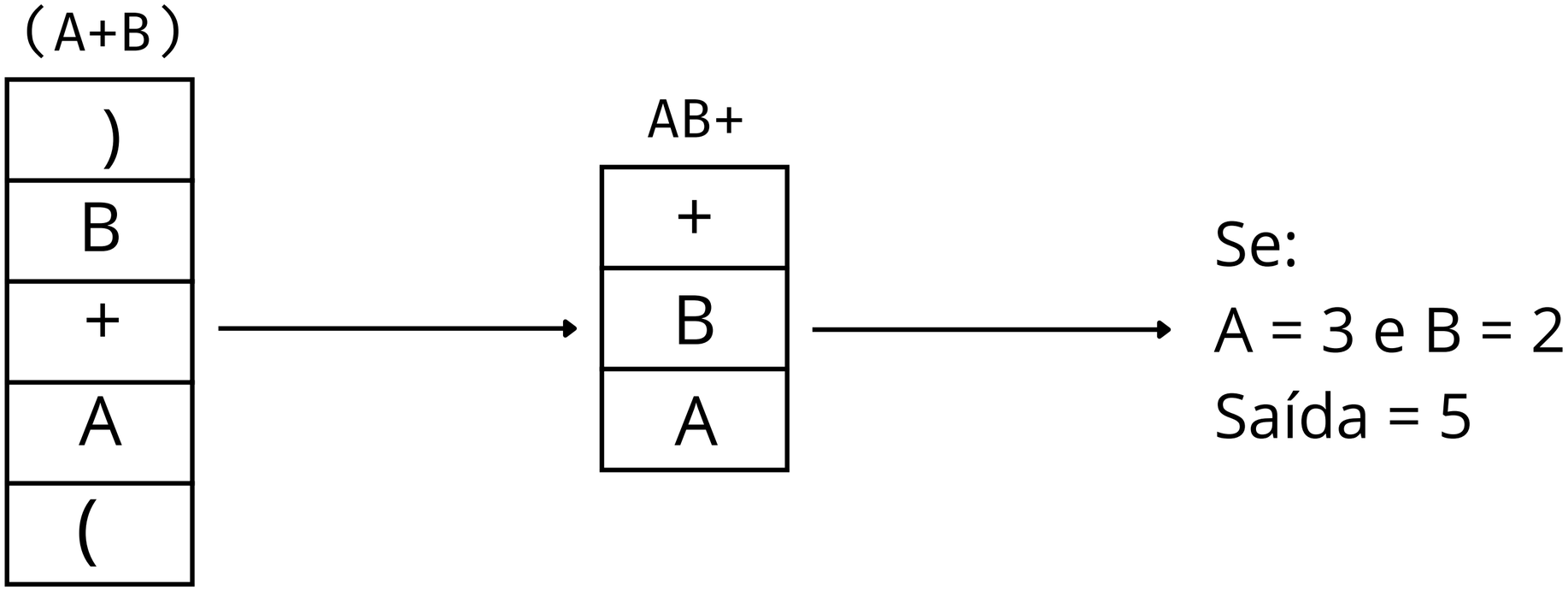


Problema 2

Objetivo

- Permitir a entrada de expressões infixas ou pós-fixas;
- Converter expressões infixas para a forma pós-fixa.
- Resolver a expressão pós-fixa

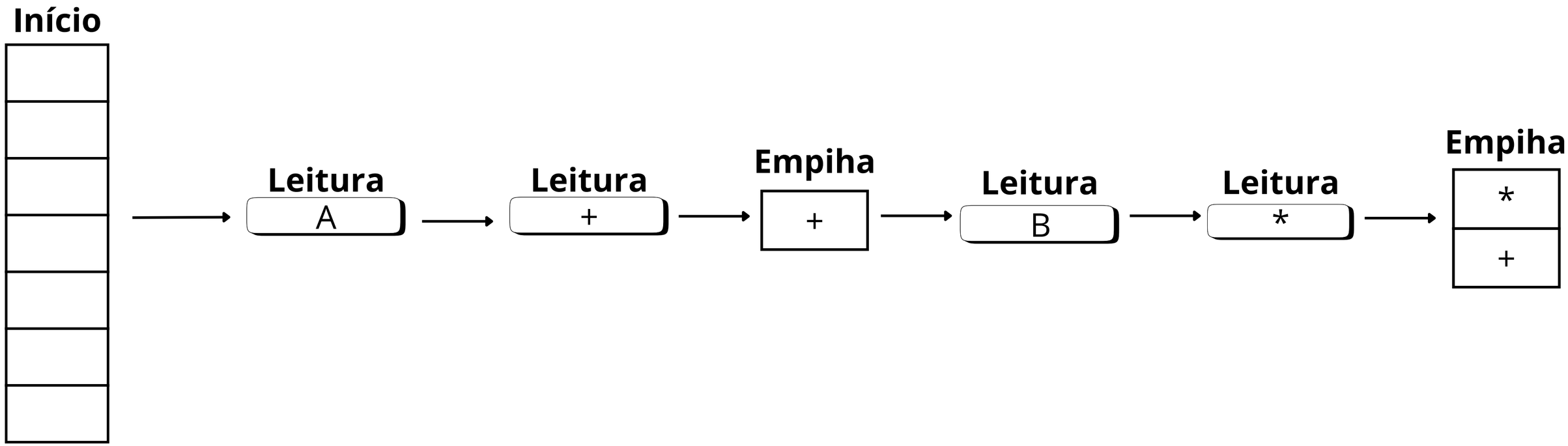
Exemplos:



Problema 2

Exemplo:

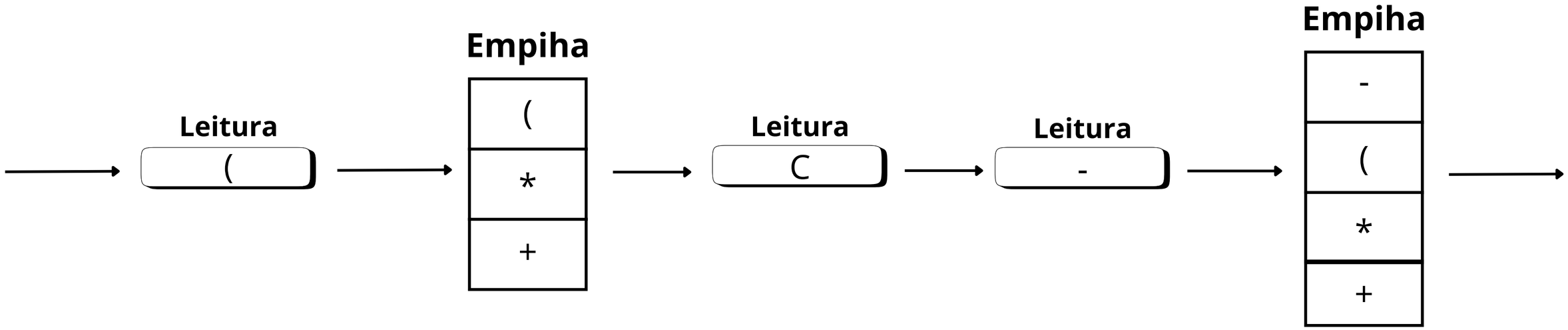
- Convertendo: A + B * (C-D)



Problema 2

Exemplo:

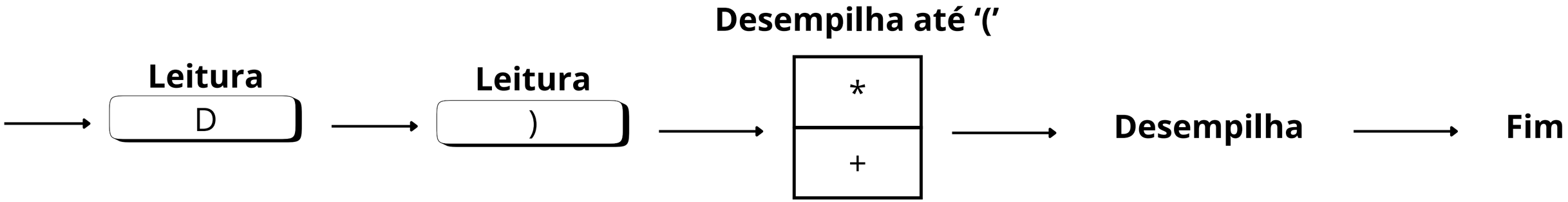
- Convertendo: A + B * (C-D)



Problema 2

Exemplo:

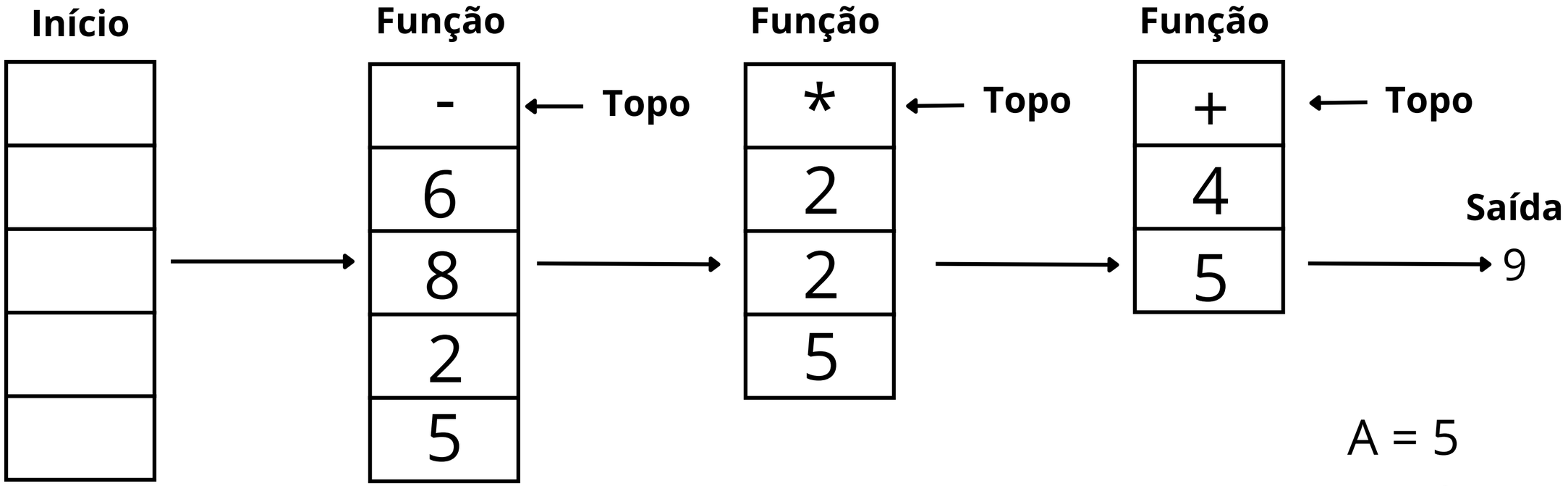
- Convertendo: A + B * (C-D)



Problema 2

Exemplos:

- Realizando Operação: ABCD-*+



A = 5

B = 2

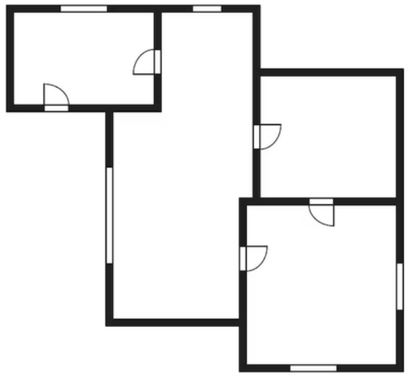
C = 8

D = 6

Problema 3

Problema

- Temos uma planta de uma casa representada por uma matriz contendo paredes e pisos.
- Um cômodo é formado por áreas de piso conectadas entre si.
- É necessário determinar quantos cômodos existem na planta.



```
##############
#...#..######
#####..#....#
####...#....#
####..#######
####..#.....#
#######.....#
#######.....#
##############
```

Objetivo

Desenvolver um algoritmo que percorra a planta, identifique cada cômodo e conte sua quantidade.

Procedimento

Utilizar busca em profundidade (DFS) com pilha para explorar as áreas de piso conectadas, marcando as já percorridas e evitando contagens duplicadas.

Problema 3

Lógica do algoritmo

- Cria uma matriz de processados.
- Cria uma pilha vazia.
- Inicia a contagem de cômodos em 0.
- Inicia percorrendo as posições válidas do mapa.
- Se encontrar um piso não processado:

  Soma 1 na contagem de cômodos.

  Empilha a posição e marca como processado.
- Enquanto a pilha não estiver vazia:

  Desempilha o topo.

  Para cada vizinho piso não processado:

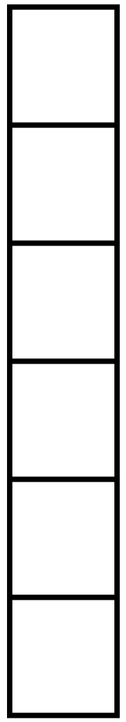
  Empilha-os e marca como processado.
- Quando a pilha esvazia, significa que todo o cômodo foi explorado.
- Continua o percurso do mapa até verificar todas as posições.

Problema 3

Lógica do algoritmo

```
# . . . # # # # # # # #
# . . . # . . # # # # # #
# # # # # . . # . . . . #
# # # # . . . # . . . . #
# # # # . . # # # # # # #
# # # # . . # . . . . . #
# # # # # # # . . . . . #
# # # # # # # . . . . . #
# # # # # # # # # # # # #
```

Mapa não processado



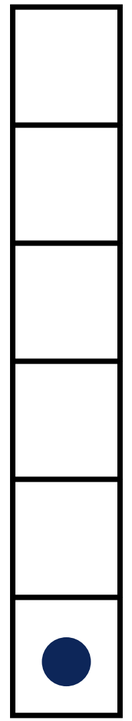
Pilha vazia

Problema 3

Lógica do algoritmo

```
# ● . . # # # # # # # #
# . . . # . . # # # # #
# # # # # . . # . . . #
# # # # . . . # . . . #
# # # # . . # # # # # #
# # # # . . # . . . . #
# # # # # # # . . . . #
# # # # # # # . . . . #
# # # # # # # # # # # #
```

Mapa sendo processado
1 Cômodo

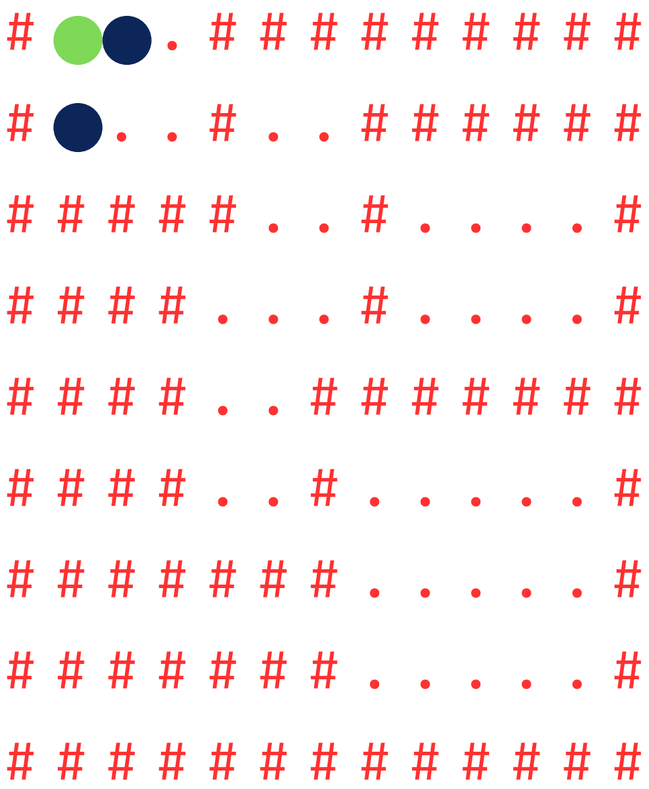


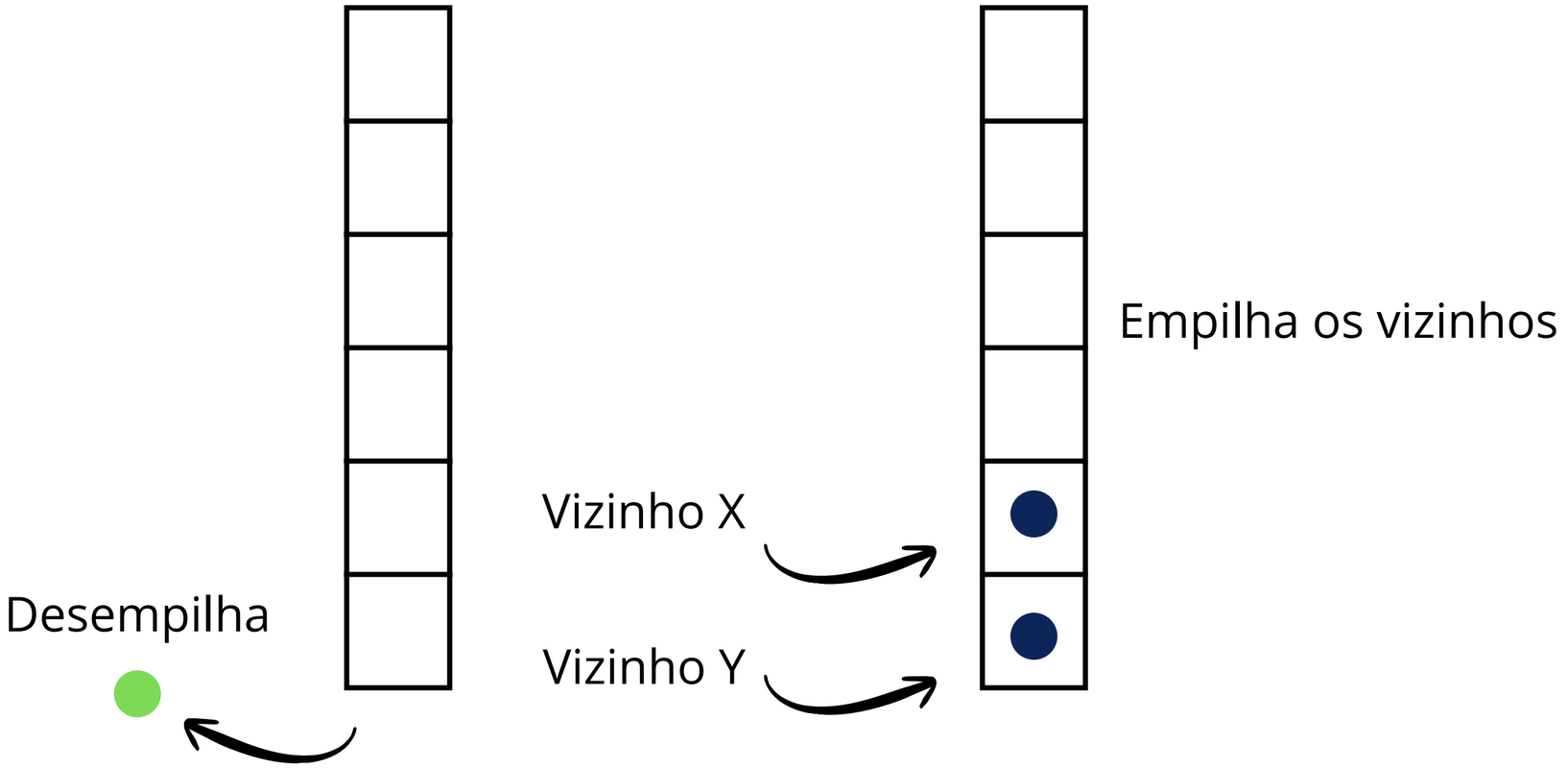
Pilha com 1 posição

Posição(i, j)

Problema 3

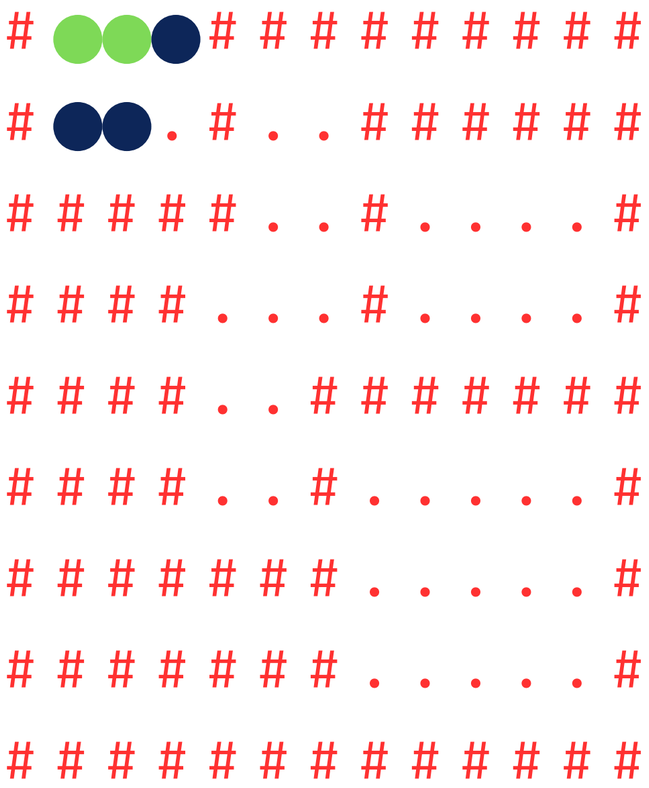
Lógica do algoritmo

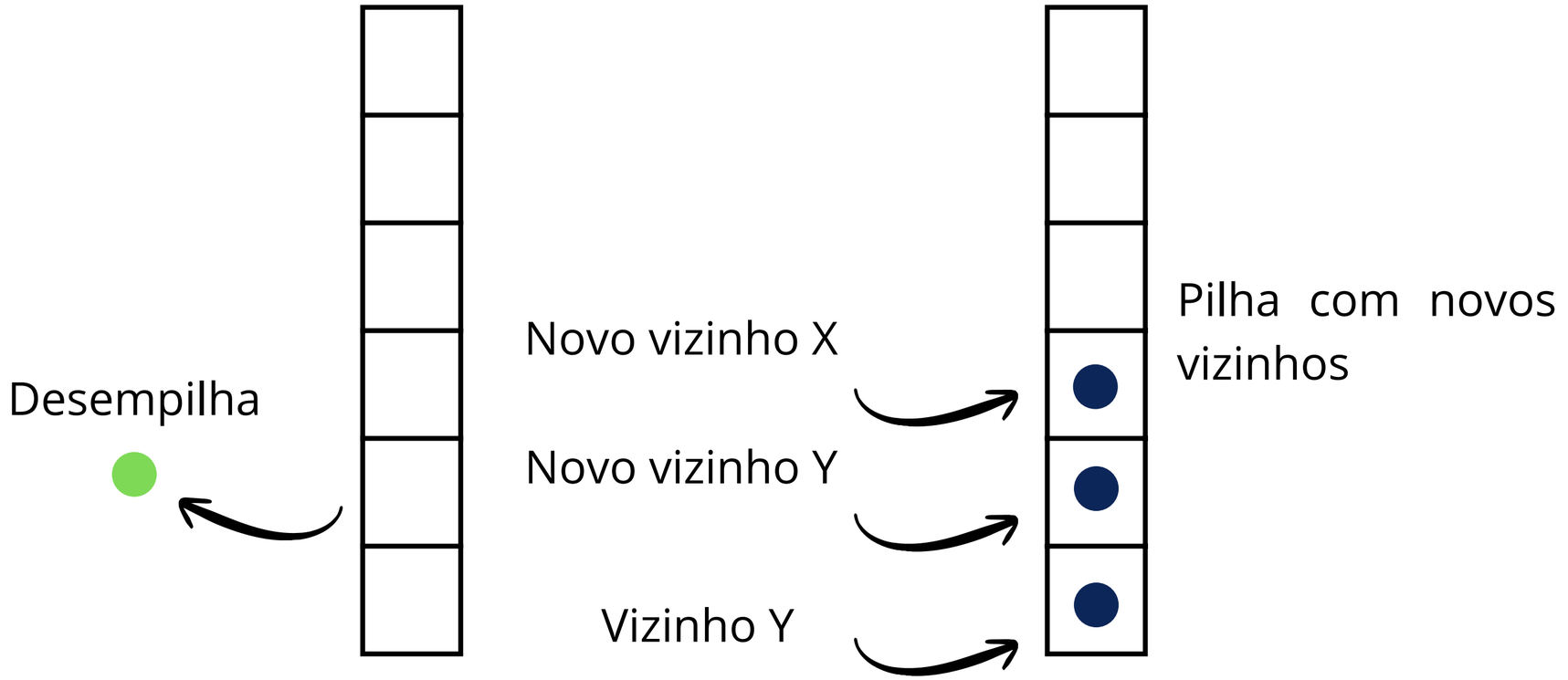




Problema 3

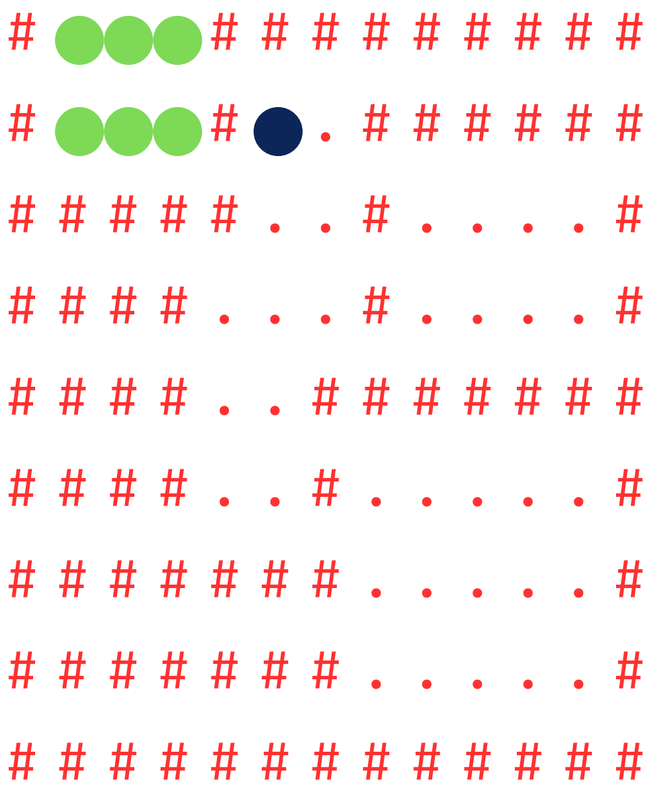
Lógica do algoritmo



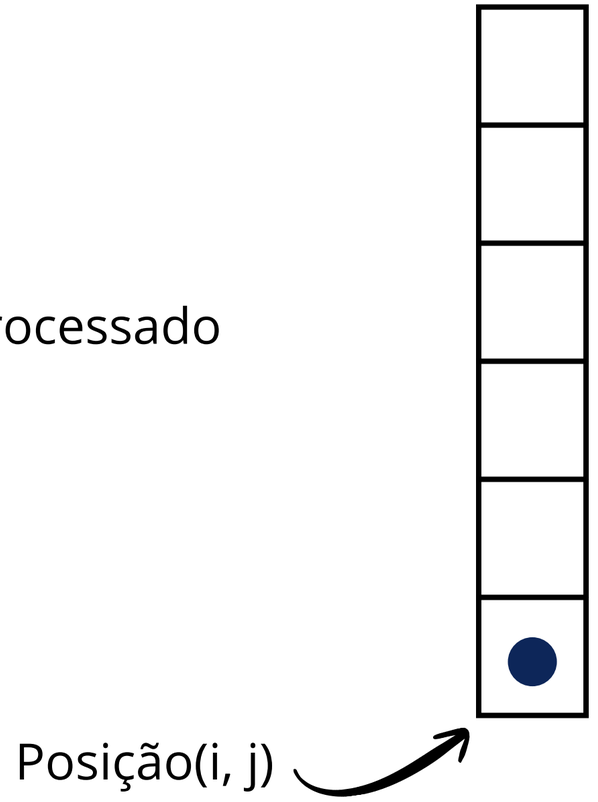


Problema 3

Lógica do algoritmo



Mapa sendo processado
2 Cômodos

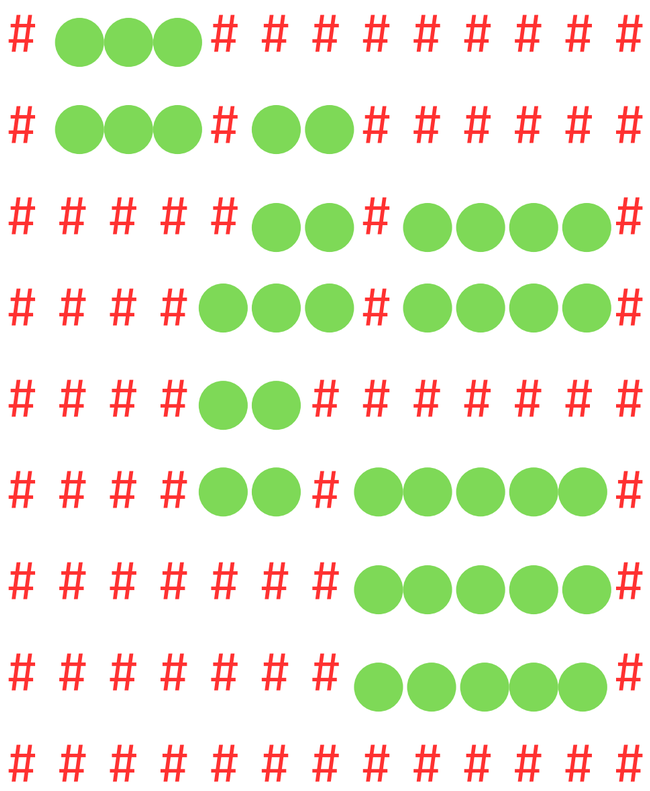


Pilha com 1 posição

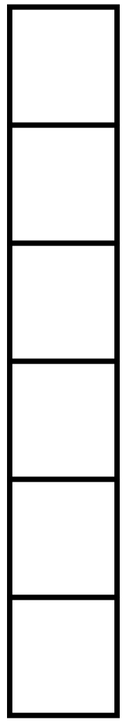
Posição(i, j)

Problema 3

Lógica do algoritmo



Mapa com 4 cômodos



Pilha vazia

Fim