Estruturas de Dados

Profa. Juliana Pinheiro Campos

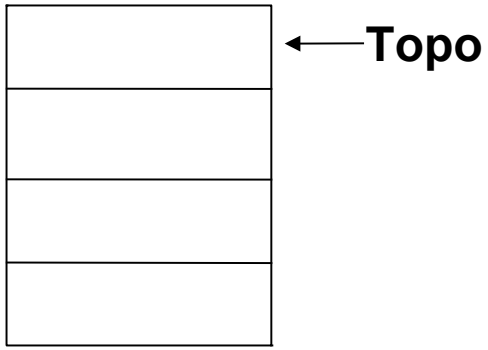
ESTRUTURAS DE DADOS

- Pilhas
  - Estrutura de dados mais **simples** e mais utilizada em programação.
  - Uma pilha pode ser considerada uma **restrição** de lista (com operações especiais de acesso).

- Pilhas

  - **Regra:** O acesso aos elementos de uma pilha é feito a partir do **topo**.

    - Inserções: sempre no topo;
    - Exclusões: sempre no topo;
    - Pesquisa: à partir do topo;



LIFO

ESTRUTURAS DE DADOS

- Pilhas
  - As pilhas são baseadas no princípio **LIFO** (*Last in, First out*):

    O último a entrar na pilha, é o primeiro a sair.
  - Os elementos só podem ser retirados na ordem inversa à ordem em que foram inseridos.
  - Ex: Pilha de provas, pilha de pratos, pilha de caixas em estoque, etc;

ESTRUTURAS DE DADOS

- Pilhas
  - Exemplos práticos de Pilha para informática:
    - Avaliação de expressões aritméticas;
    - Construção de compiladores;
    - Gerência de memória;

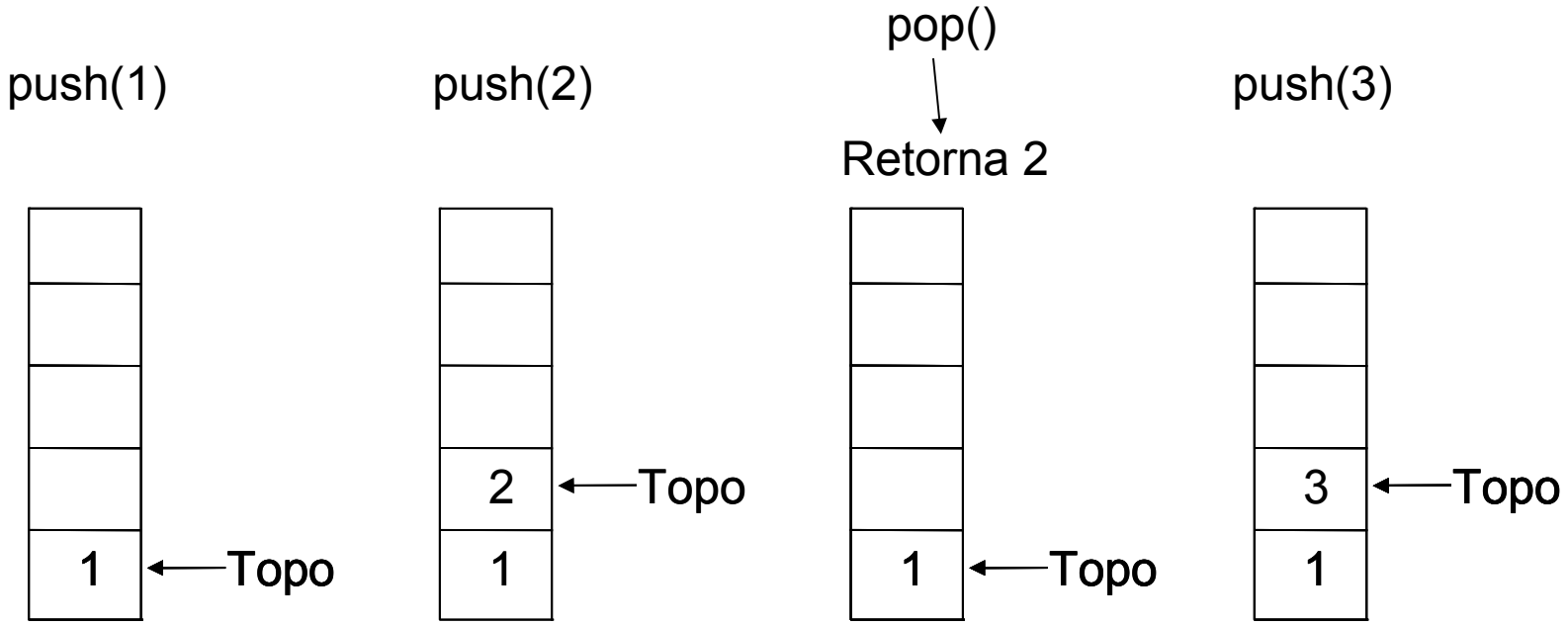
      Ex: As variáveis locais de funções são dispostas em uma pilha.

ESTRUTURAS DE DADOS

- Pilhas
  - Operações básicas que devem ser implementadas:
    - push (empilhar): insere um novo elemento no topo da pilha;
    - pop (desempilhar): remove um elemento do topo da pilha;

ESTRUTURAS DE DADOS

- Funcionamento da pilha



- Implementações de pilhas

  - Exemplo: pilha que armazena valores reais.
    - Usando vetor;
    - Usando lista encadeada;

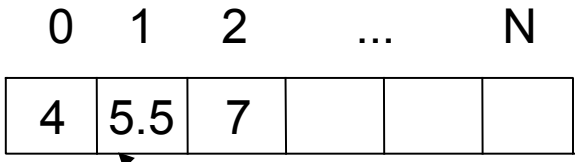
  - Interface: Protótipos das funções

```
Pilha* criarPilha( );          ⟶  Cria uma pilha vazia
void push(Pilha* p, float v);  ⟶  Insere um elemento no topo
float pop(Pilha* p);           ⟶  Remove elemento do topo
int estaVazia(Pilha* p);       ⟶  Verifica se a pilha está vazia
void liberarPilha(Pilha* p);   ⟶  Libera a estrutura de pilha
```

- Implementação de pilha com vetor

  - Utilizada quando sabemos de antemão o número máximo (N) de elementos que podem estar armazenados na pilha.

  - Um vetor de informações (info) armazena os elementos da pilha:



Se temos n elementos armazenados na pilha, onde está o elemento que representa o topo?

Na posição n – 1 do vetor

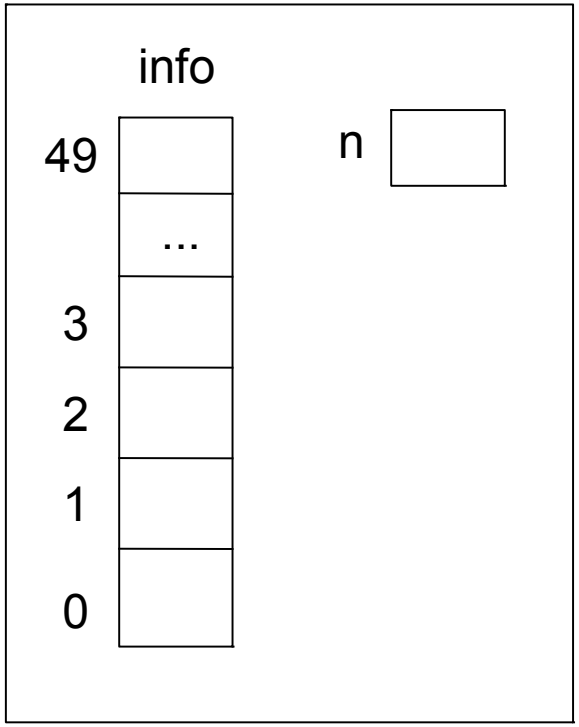
Os elementos inseridos ocupam as primeiras posições do vetor

- Implementação de pilha com vetor

  - Estrutura que representa a pilha:

```
#define N 50

typedef struct pilha {
  float info[N];
  int n;
}Pilha;
```

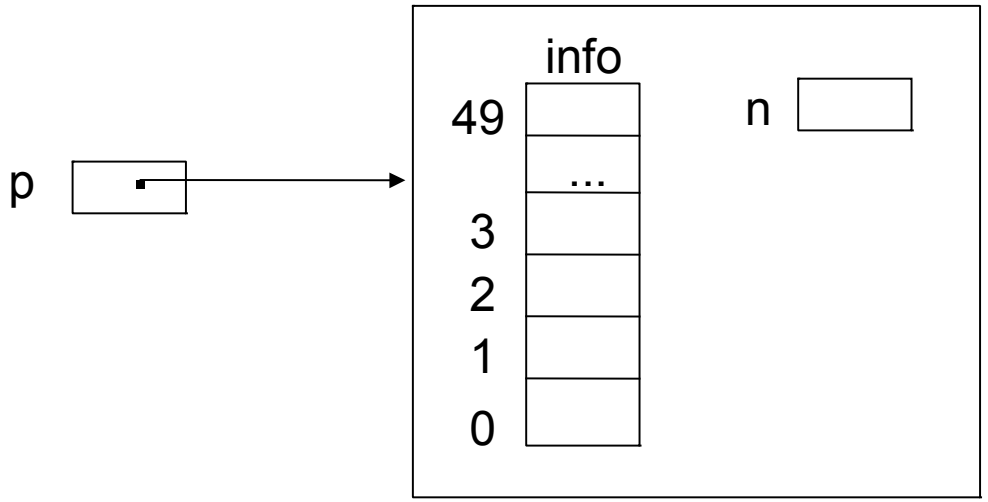
Número de elementos armazenados



- Implementação de pilha com vetor

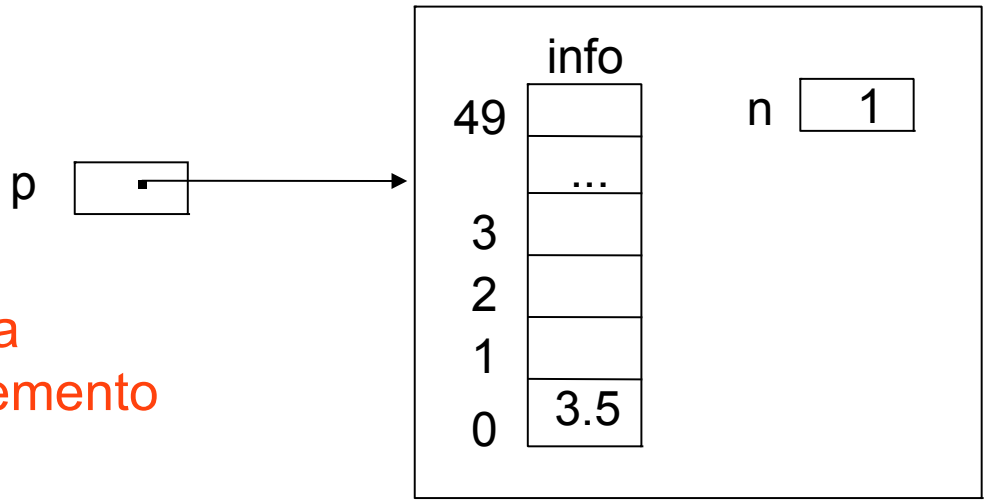
  - Função para criar a pilha:

```
Pilha* criarPilha() {
  Pilha* p = (Pilha*) malloc(sizeof(Pilha));
  p->n = 0;
  return p;
}
```



- Implementação de pilha com vetor
  - Função para inserir um elemento no topo:

```
void push(Pilha* p, float v) {
    if(p->n == N) {
       printf("Capacidade da pilha estourou");
       exit(1);
    }
     p->info[p->n] = v;
     p->n++;
}
```



Observe que o n representa
o índice onde o próximo elemento
será inserido.

- Implementação de pilha com vetor

  - Função que verifica se a lista está vazia:

```
int estaVazia(Pilha* p) {
  return (p->n==0);
}
```

Retorna 1 se a pilha está vazia,
0 caso contrário

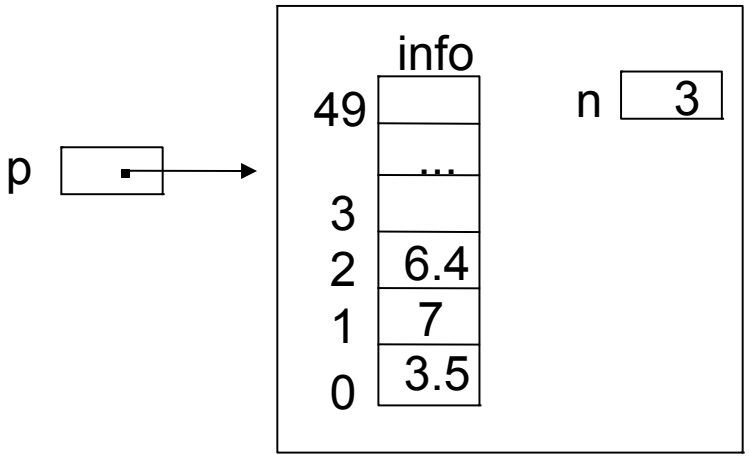
- Implementação de pilha com vetor

  - Função para retirar o elemento do topo e fornecer seu valor como retorno:

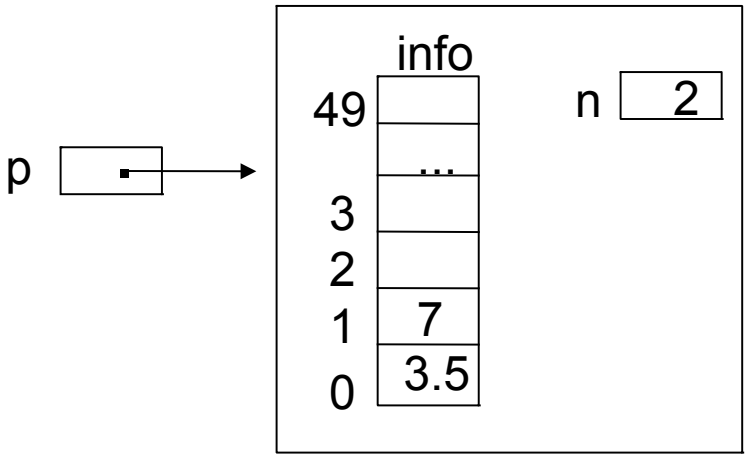
```
float pop(Pilha* p) {
  if(estaVazia(p)){
       printf("Pilha vazia");
       exit(1);

  }
  float v = p->info[p->n - 1];
  p->n--;
  return v;
}
```

- Implementação de pilha com vetor

  - Função para retirar o elemento do topo e fornece seu valor como retorno:



Antes do pop



Depois do pop

Retorna 6.4

- Implementação de pilha com vetor

  - Função para liberar a memória alocada pela pilha:

```
void liberarPilha(Pilha* p) {
  free(p);
}
```

- Implementação de pilha com vetor
  - Implemente as seguintes funções para pilha:
    - Função que verifica se a pilha está cheia:

      int estaCheia(Pilha* p) ;
    - Função que imprime os valores armazenados na pilha:

      void imprimePilha(Pilha* p);
    - Função que retorna o valor do topo da pilha:

      float verTopo(Pilha* p);

ESTRUTURAS DE DADOS

- Implementação de pilha com lista encadeada
  - Usada quando o número máximo de elementos que serão armazenados na pilha não é conhecido.
  - Os elementos são armazenados em uma lista encadeada.
  - A pilha é representada por um ponteiro para o primeiro nó da lista.

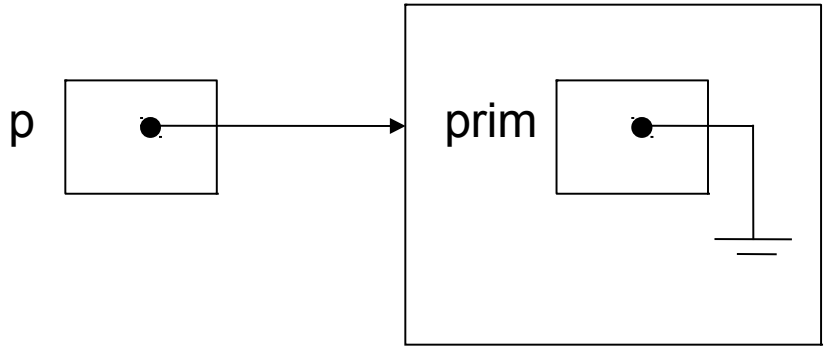
ESTRUTURAS DE DADOS

- Implementação de pilha com lista encadeada

```
struct noLista {
  float info;
  struct noLista *prox;
};

typedef struct noLista NoLista;

struct pilha {
  NoLista* prim;
};

typedef struct pilha Pilha;
```

- Implementação de pilha com lista encadeada
  - Função de criação:

```
Pilha* criarPilha( ) {
  Pilha* p = malloc(sizeof(Pilha));
  p->prim = NULL;
  return p;
}
```



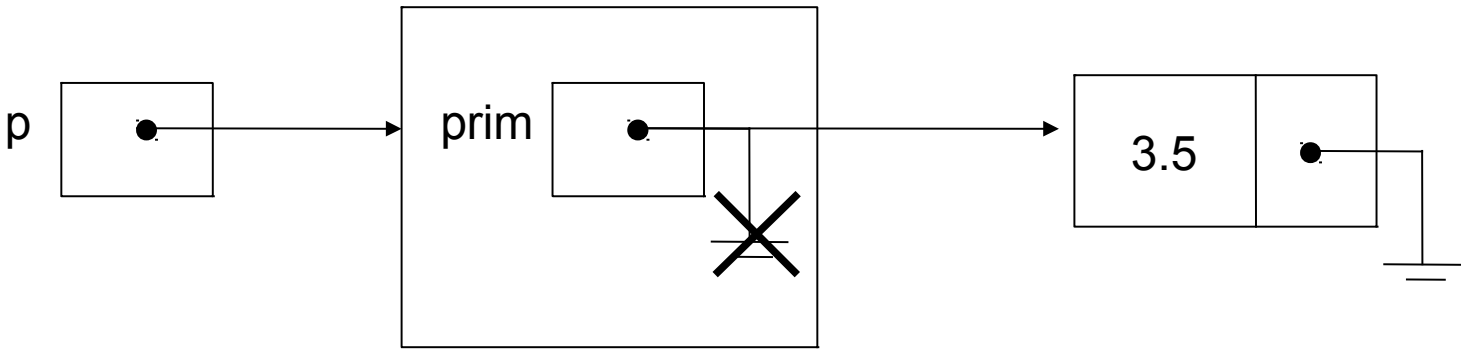
- Implementação de pilha com lista encadeada

  - Função de inserção:

```
void push(Pilha* p, float v) {
  NoLista* novo = malloc(sizeof(NoLista));
  if(novo != NULL){
      novo->info = v;
      novo->prox = p->prim;
      p->prim = novo;
  }
  else
      exit(1);
}
```

O primeiro elemento representa o topo da pilha. Os elementos são inseridos no início da lista.

- Implementação de pilha com lista encadeada
  - Função de inserção:



- Implementação de pilha com lista encadeada
  - Função que verifica se a lista está vazia:

```
int estaVazia(Pilha* p) {
  return (p->prim==NULL);
}
```

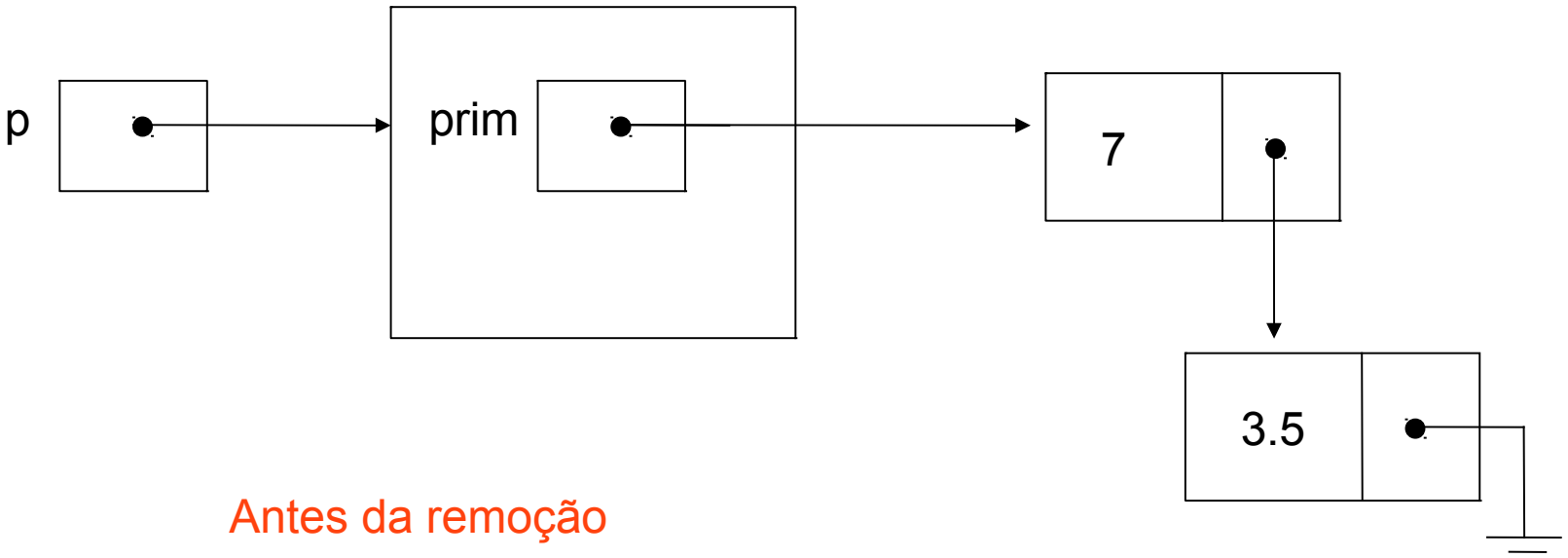
A pilha está vazia se a lista estiver vazia

- Implementação de pilha com lista encadeada
  - Função de remoção:

```
float pop(Pilha* p) {

  if(estaVazia(p)){
     printf("\nPilha vazia\n");
     exit(1);
  }

  NoLista* temp = p->prim;
  float v = p->prim->info;
  p->prim = p->prim->prox;
  free(temp);
  return v;
}
```

O primeiro elemento da lista representa o topo. Logo, retiramos sempre o elemento do início da lista.

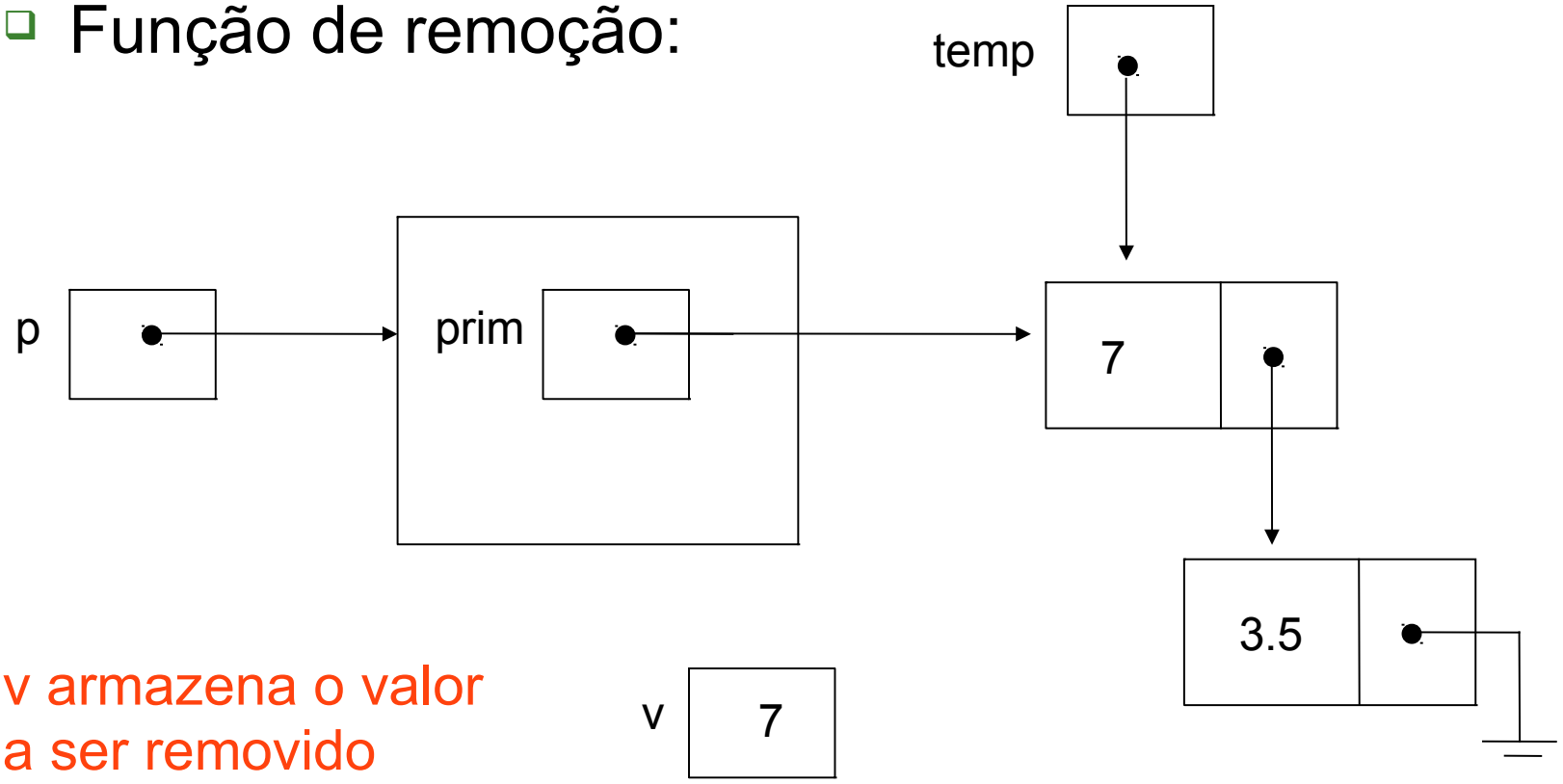
- Implementação de pilha com lista encadeada
  - Função de remoção:



Antes da remoção

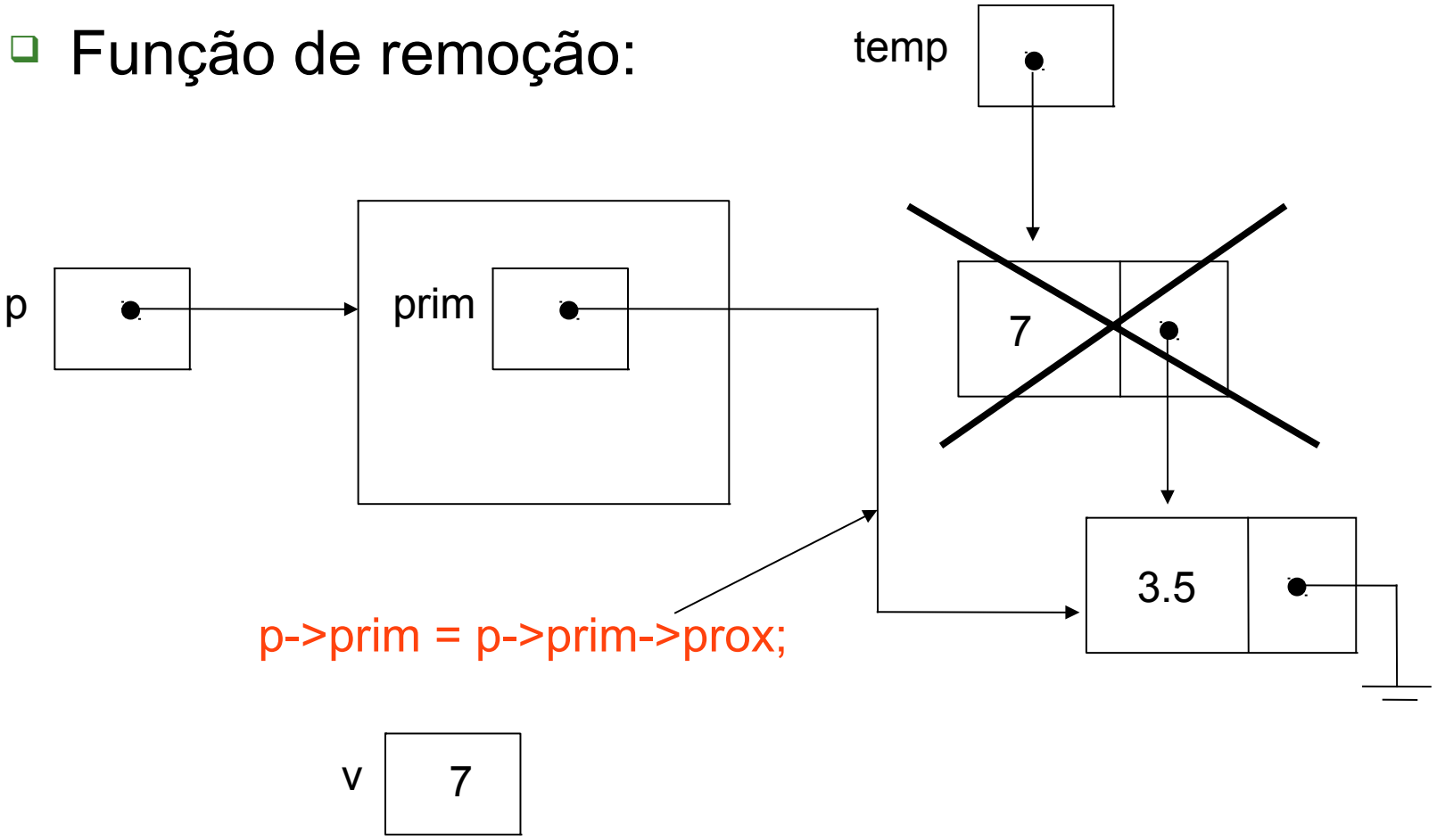
- Implementação de pilha com lista encadeada

  - Função de remoção:



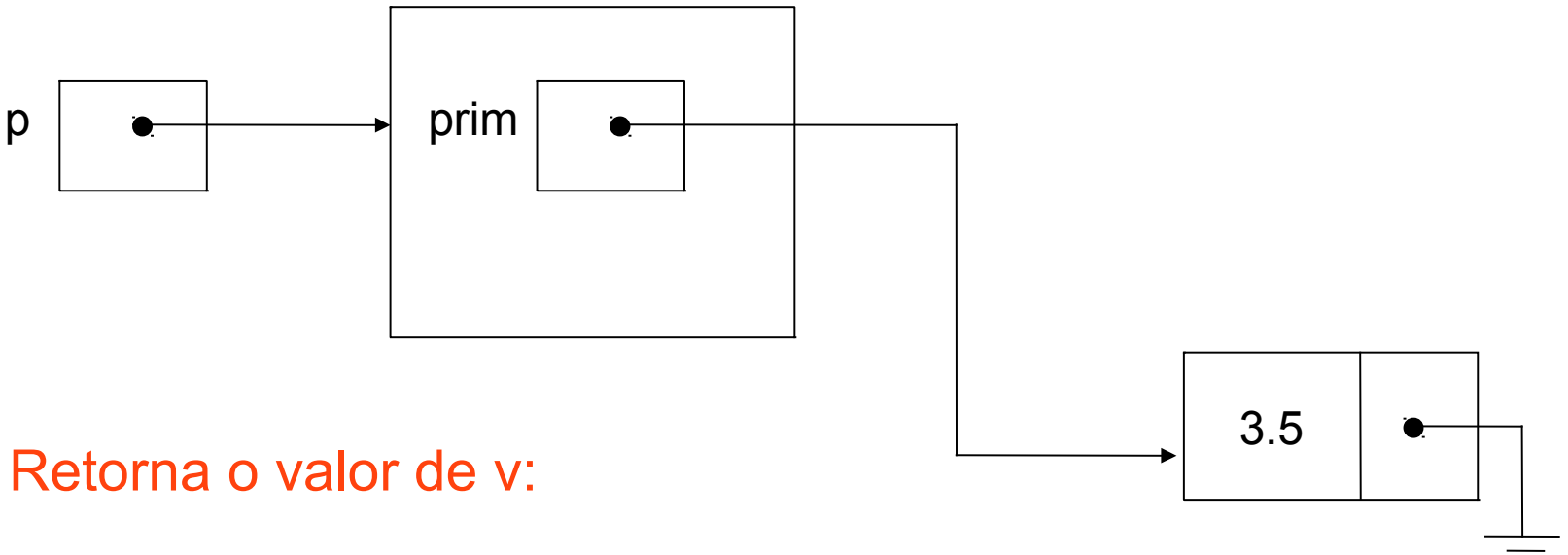
v armazena o valor
a ser removido

- Implementação de pilha com lista encadeada
  - Função de remoção:

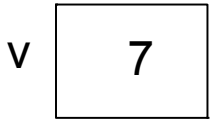


- Implementação de pilha com lista encadeada

  - Função de remoção:



Retorna o valor de v:



- Implementação de pilha com lista encadeada

  - Função que libera a pilha:

```
void liberarPilha(Pilha* p) {
  NoLista* q = p->prim;
  while(q!=NULL) {
    NoLista* temp = q->prox;
    free(q);
    q = temp;
  }
  free(p);
}
```

- Implementação de pilha com lista encadeada
  - Implemente as seguintes funções para a pilha
    - Função para apresentar o elemento do topo da pilha:

      float verTopo(Pilha* p);

    - Função que imprime os valores armazenados na pilha

      void imprimePilha(Pilha* p);

- Exemplo de uso: Calculadora pós-fixada
  - Um bom exemplo de aplicação de pilha é o funcionamento das calculadoras HP (Hewlett-Packard).
  - Elas trabalham com expressões pós-fixadas:

Para avaliar a expressão (1 – 2) * (4 + 5)

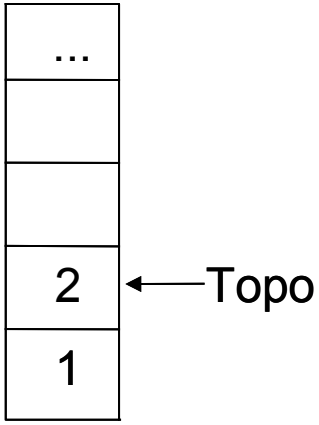
podemos digitar: 1 2 – 4 5 + *

- Exemplo de uso: Calculadora pós-fixada

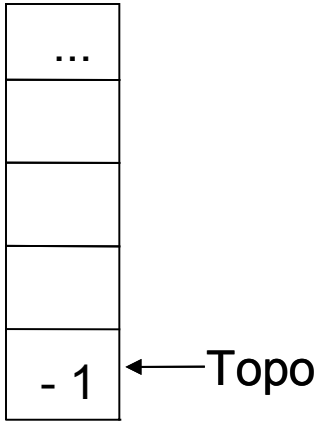
  - O funcionamento é simples: cada operando é empilhado em uma pilha de valores. Quando se encontra um operador, desempilha-se o número apropriado de operandos, realiza-se a operação devida e empilha-se o resultado.

- Exemplo de uso: Calculadora pós-fixada
  - Avaliação da expressão: 1 2 – 4 5 + *

1º passo: desempilha resultado de

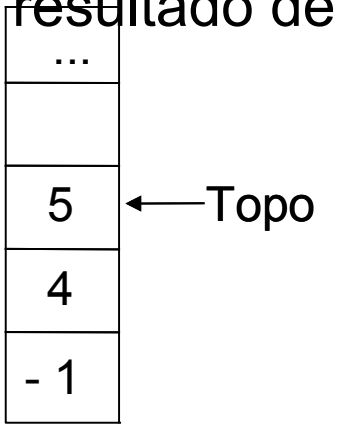


2º passo: Aparece -, 1 e 2 e empilha o 1 - 2

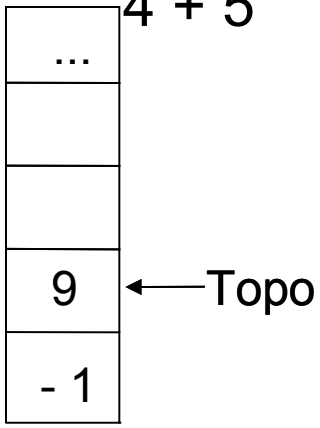


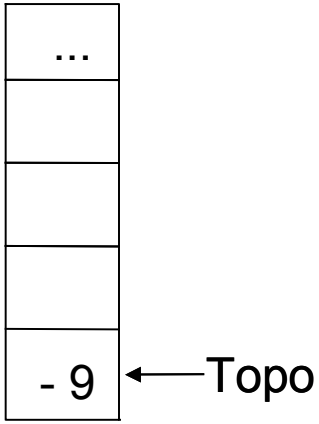
- Exemplo de uso: Calculadora pós-fixada
  - Avaliação da expressão: 1 2 – 4 5 + *

3º passo: desempilha resultado de



4º passo: Aparece +, 4 e 5 e empilha o 4 + 5



- Exemplo de uso: Calculadora pós-fixada
  - Avaliação da expressão: 1 2 – 4 5 + *

6º passo: O operador * desempilha 9 e -1 e empilha o resultado de 9 * (-1)