ESTRUTURA DE DADOS: PILHAS

Prof. Jean Nunes

PILHA

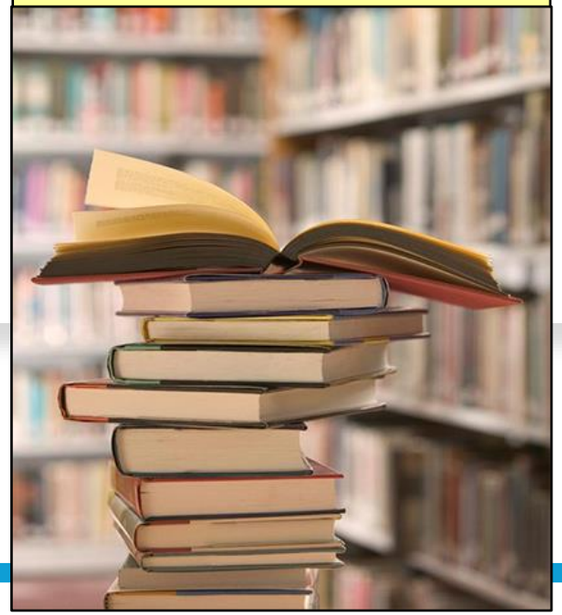
1 – Definição e Características

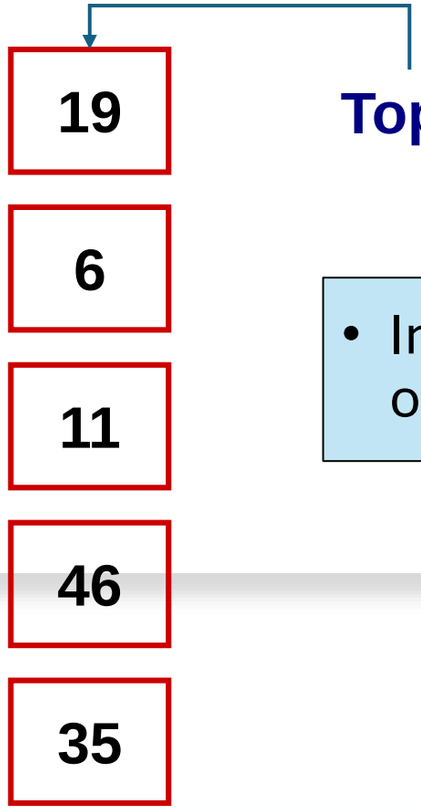
2 – Pilha Dinâmica

PILHA

Uma estrutura do tipo “Pilha” é uma sequência de elementos do mesmo tipo, como as “Listas” e “Pilhas”.

Last-in, First-out (LIFO)

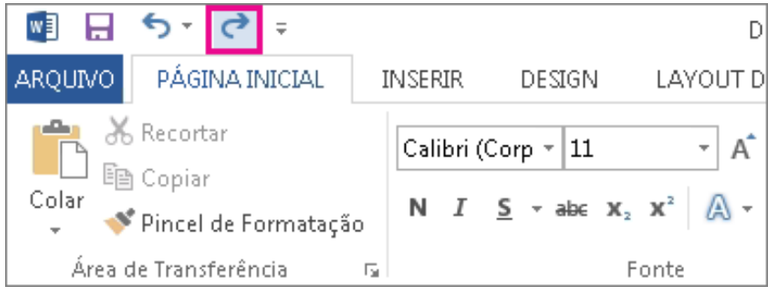




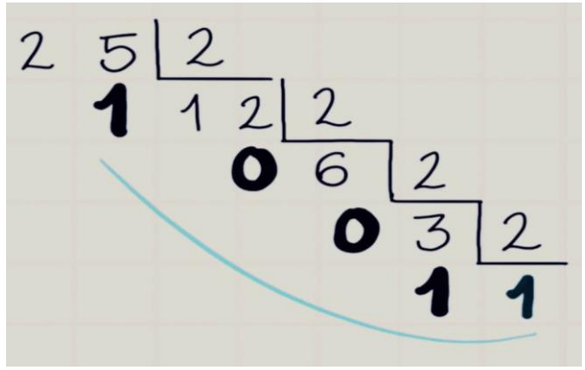
- Inserções e remoções ocorrem apenas no início.

PILHA

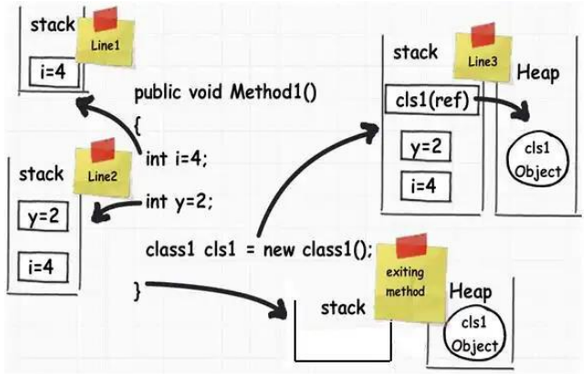
Aplicações



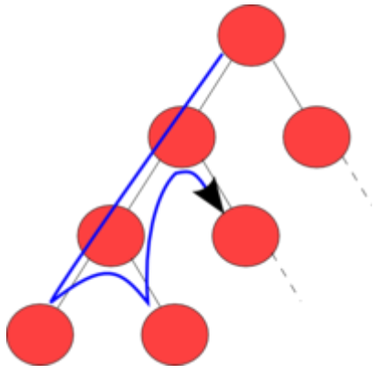
Desfazer/Refazer



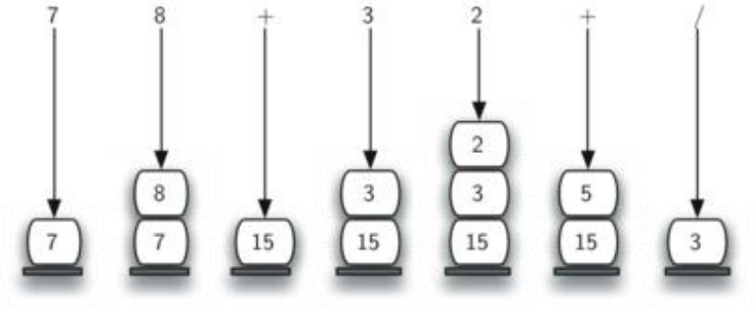
Conversões entre bases



Gerenciamento de memória



Busca em Profundidade (DFS)



Conversão de Expressões Infixa para Pós-fixas

Navegação em Navegadores Web

PILHA

Operações básicas

Criar pilha

Inserir um elemento no início da pilha

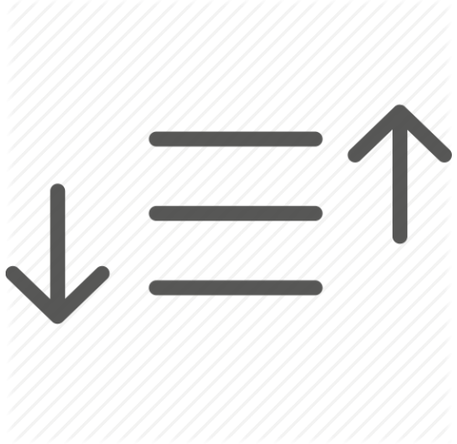
Excluir um elemento no início da pilha

Acessar o elemento do início da pilha

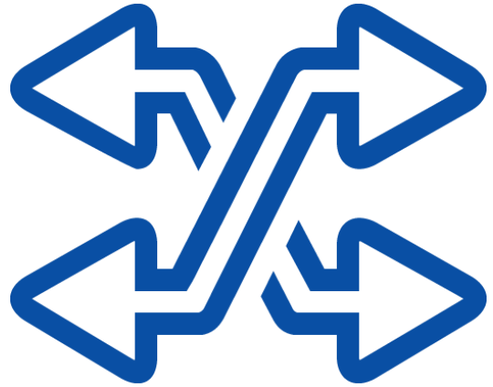
Destruir pilha

PILHA

Essas operações dependem do tipo de alocação de memória usada



Estática



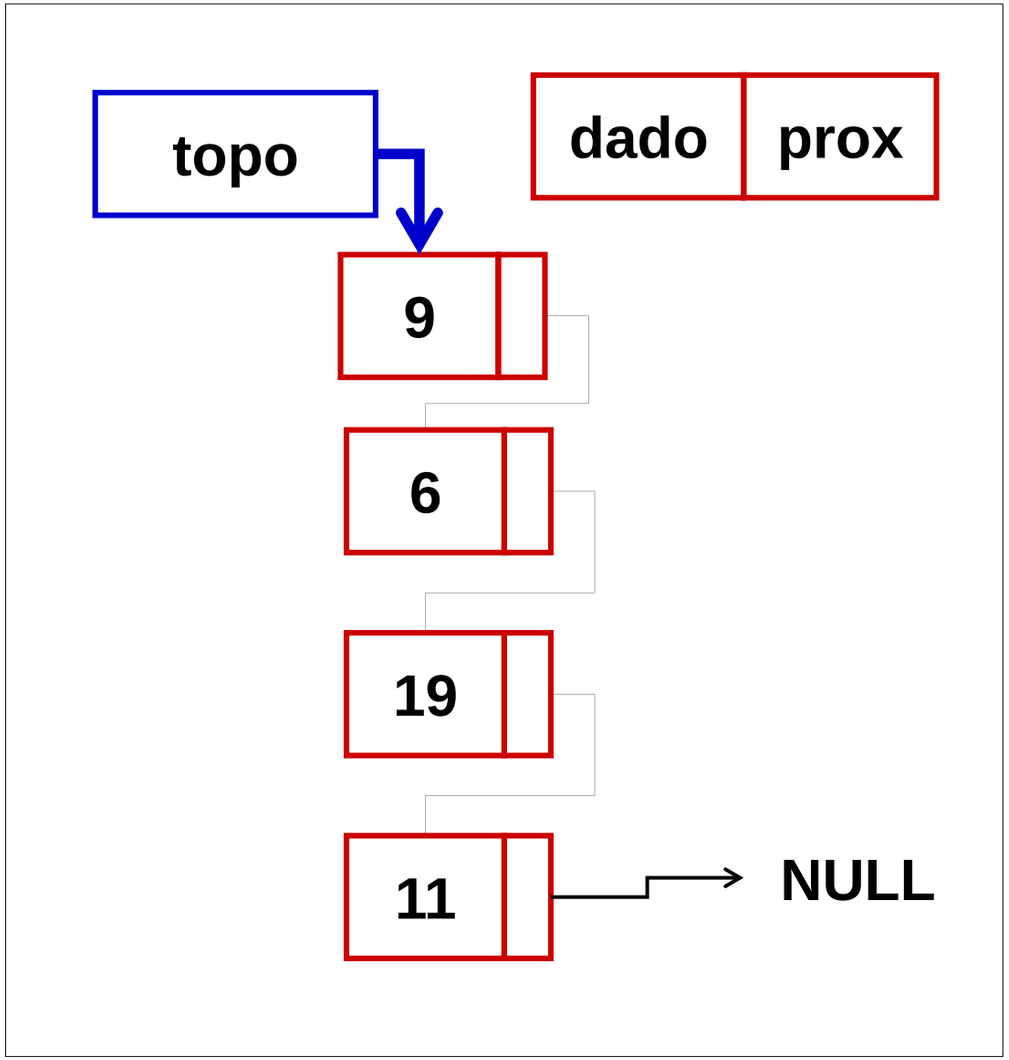
Dinâmica

PILHA

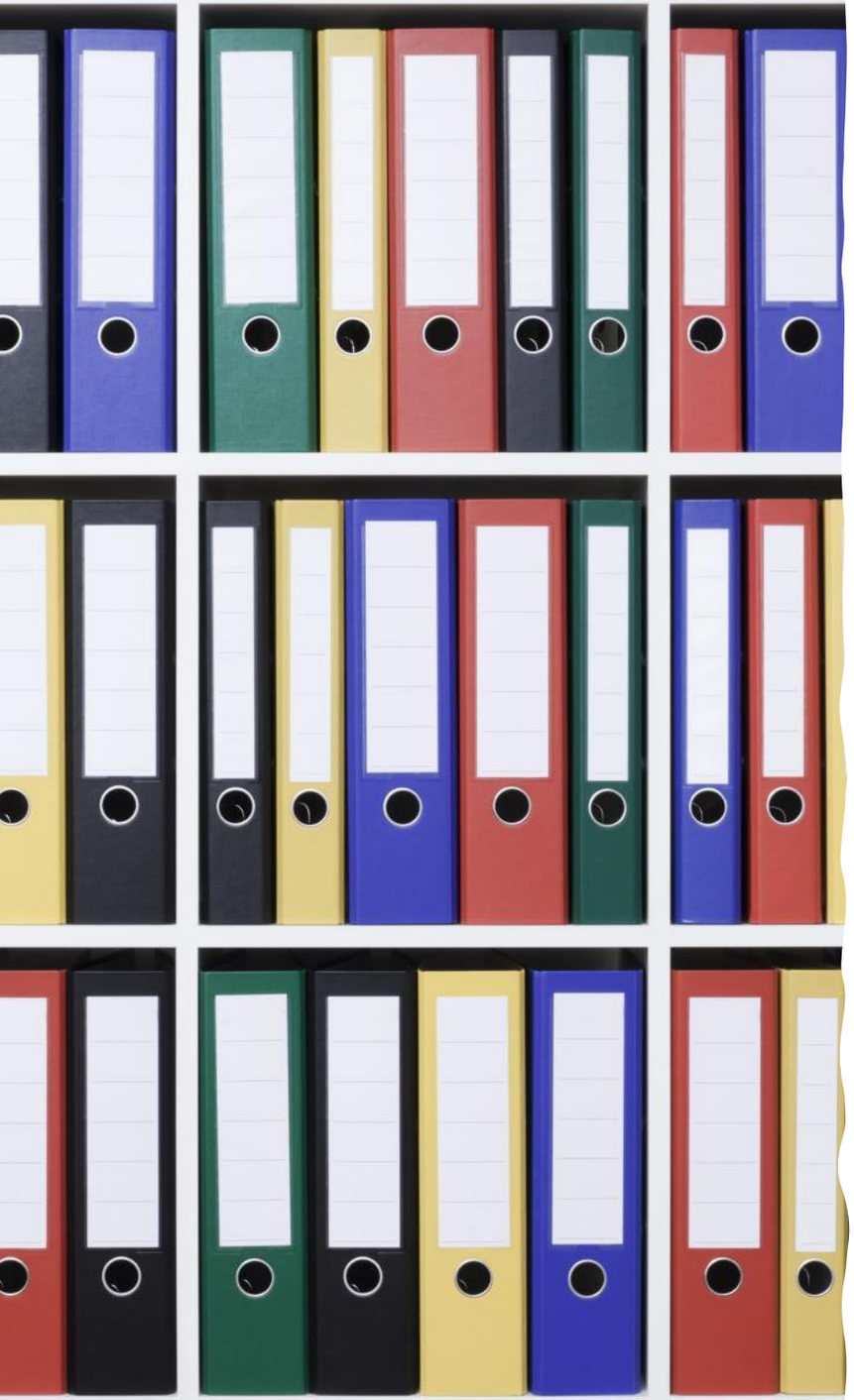
Dinâmica

PILHA DINÂMICA

- Tipo de pilha onde cada elemento aponta para o seu sucessor na “pilha”.
- Usa um ponteiro para o primeiro elemento da pilha e uma indicação de final.



Pilha *pi



PILHA DINÂMICA

//Arquivo PilhaDinamica.h

- Protótipos das funções
- O tipo de dado armazenado na lista
- O ponteiro “pilha”

//Arquivo PilhaDinamica.c

- O tipo de dado “pilha”
- Implementa as suas funções

//main.c

- Interface com o usuário

PILHA DINÂMICA

Implementando

//Arquivo FilaDinamica.h

```
struct aluno{
    int matricula;
    char nome[30];
    float n1,n2,media;
};
typedef struct elemento* Pilha;
```

// FilaDinamica.c

```
#include <stdio.h>
#include <stdlib.h>
#include “PilhaDinamica.h"

//Definição do tipo Pilha
struct elemento{
    struct aluno dados;
    struct elemento *prox;
};
typedef struct elemento Elem;
```

PILHA DINÂMICA

Implementando

// PilhaDinamica.h

```
Pilha* cria_pilha();
```

//main.c

```
#include “PilhaDinamica.h"
Pilha *pi = cria_pilha();
```

// PilhaDinamica.c

```
Pilha* cria_pilha(){
    Pilha* pi = (Pilha*) malloc(sizeof(Pilha));
    if(pi != NULL)
        *pi = NULL;
    return pi;
}
```

pilha *pi;

PILHA DINÂMICA

Implementando

LIBERA PILHA

// PilhaDinamica.h
```
void libera_pilha(Pilha* pi);
```

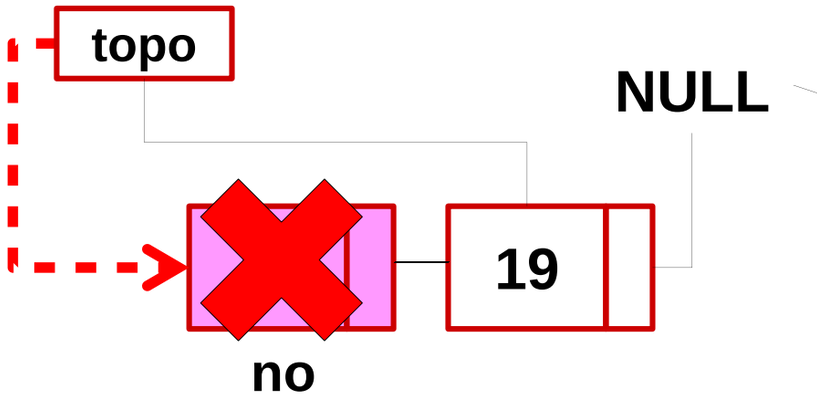
//main.c
```
libera_pilha(pi);
```

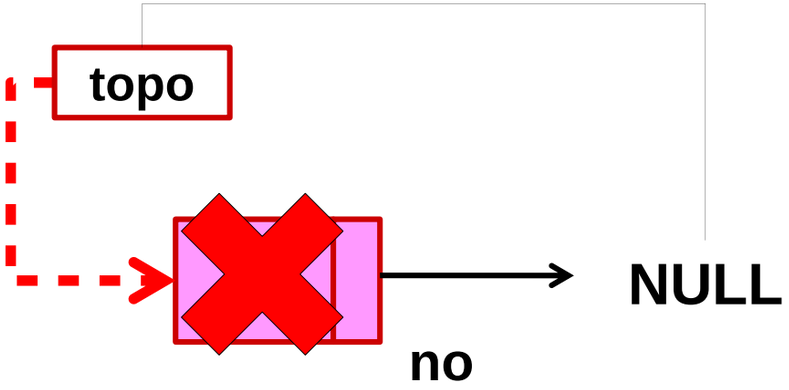
//Arquivo PilhaDinamica.c
```
void libera_pilha(Pilha* pi){
    if(pi != NULL){
        Elem* no;
        while((*pi) != NULL){
            no = *pi;
            *pi = (*pi)->prox;
            free(no);
        }
        free(pi);
    }
}
```

PILHA DINÂMICA

Implementando

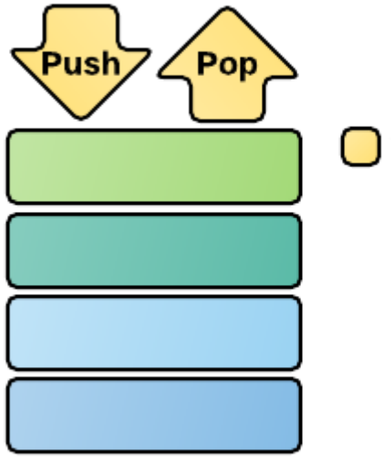
LIBERA PILHA






```
while((*pi) != NULL){
    no = *pi;
    *pi = (*pi)->prox;
    free(no);
}
free(pi);
```

PILHA DINÂMICA
Informações básicas



Tamanho | Fila vazia

PILHA DINÂMICA

Informações básicas

Tamanho

//PilhaDinamica.h

```
int tamanho_pilha(Pilha* pi);
```

//main.c

```
int x = tamanho_pilha(pi);
```

Se quiser, posso inserir o campo quantidade e incrementá-lo ao inserir um novo elemento.

//PilhaDinamica.c

```
int tamanho_Pilha(Pilha* pi){
    if(pi == NULL)
        return 0;
    int cont = 0;
    Elem* no = *pi;
    while(no != NULL){
        cont++;
        no = no->prox;
    }
    return cont;
}
```

FILA DINÂMICA

Informações básicas

Fila vazia

—

// PilhaDinamica.h

```
int pilha_vazia(Pilha* pi);
```

//main.c

```
if(pilha_vazia(pi))
```

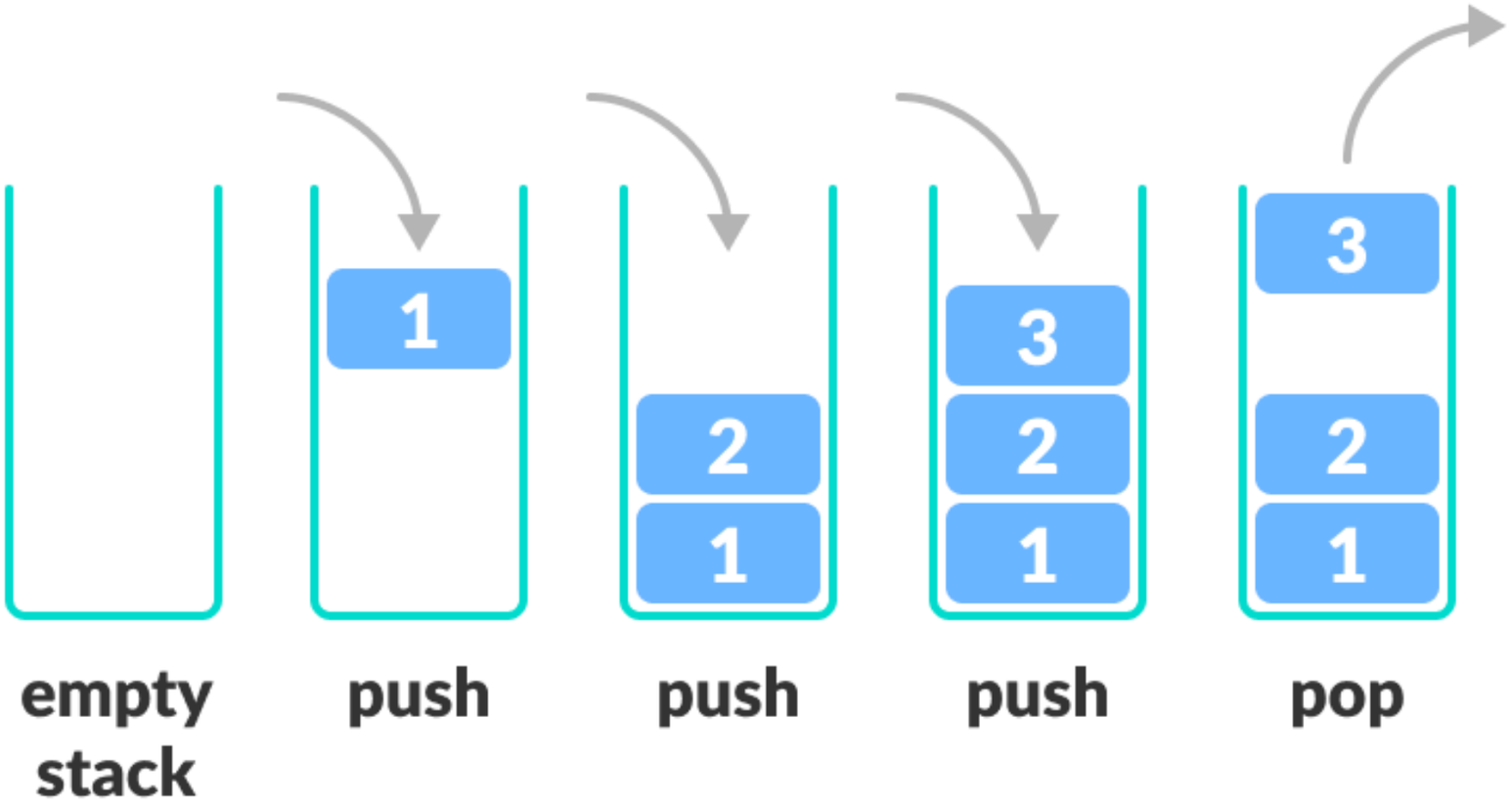
Fila *pi;

//PilhaDinamica.c

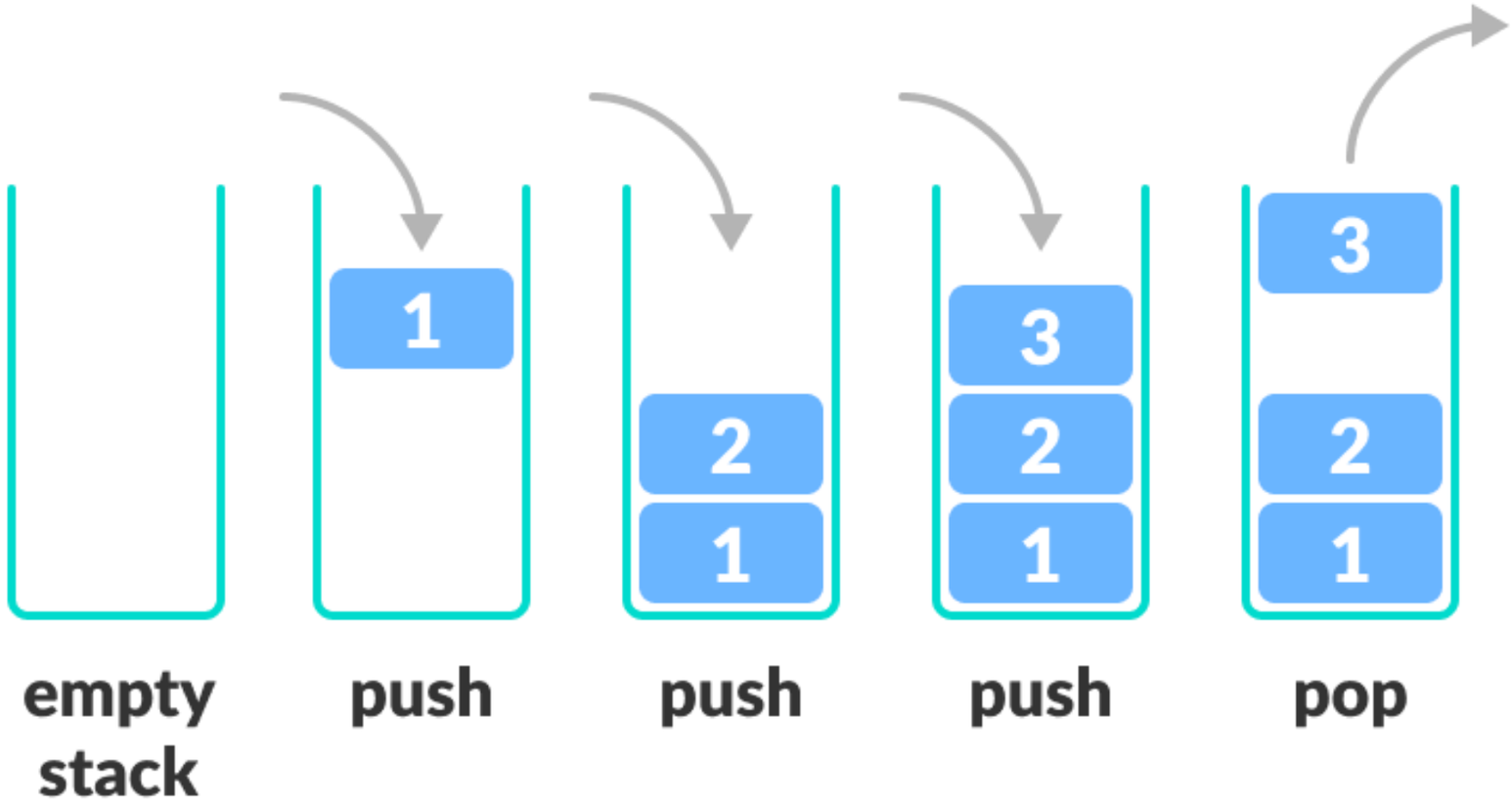
```
int Pilha_vazia(Pilha* pi){
    if(pi == NULL)
        return 1;
    if(*pi == NULL)
        return 1;
    return 0;
}
```

PILHA DINÂMICA

Inserção



PILHA DINÂMICA
Inserção



- Na “pilha” a inserção é sempre no seu topo, ou em pilha vazia.
- Não se pode inserir em pilha cheia.

PILHA DINÂMICA: Inserção -> Empilhar (push)

//Arquivo PilhaDinamica.h

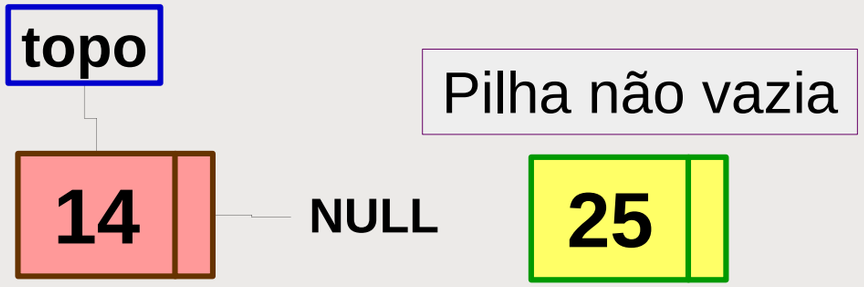
```
int insere_pilha(Pilha* pi, struct aluno al);
```

//main.c

```
int x = insere_pilha(pi, <dados_aluno>);
```

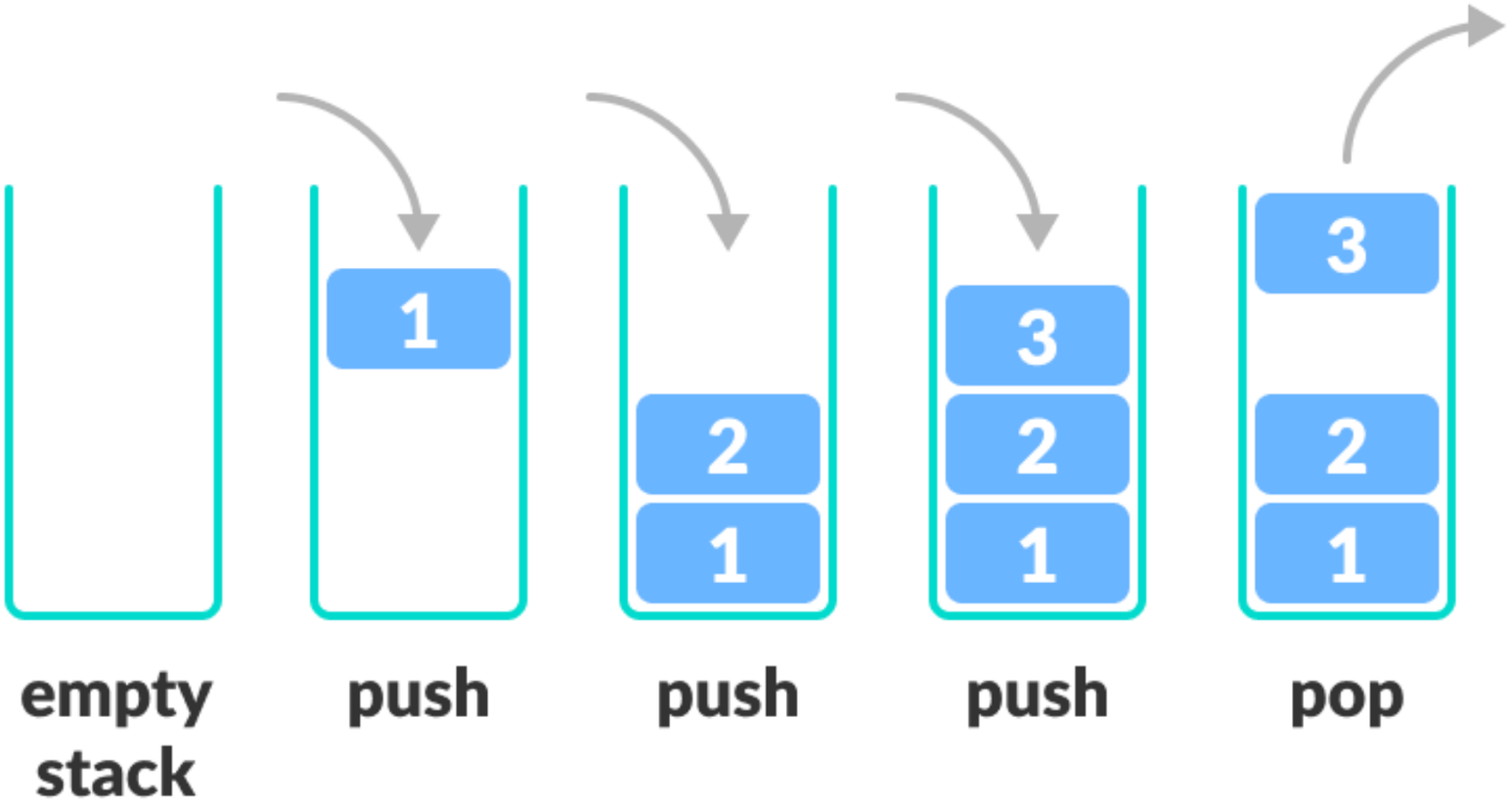
Pilha *pi;

Pilha *pi;



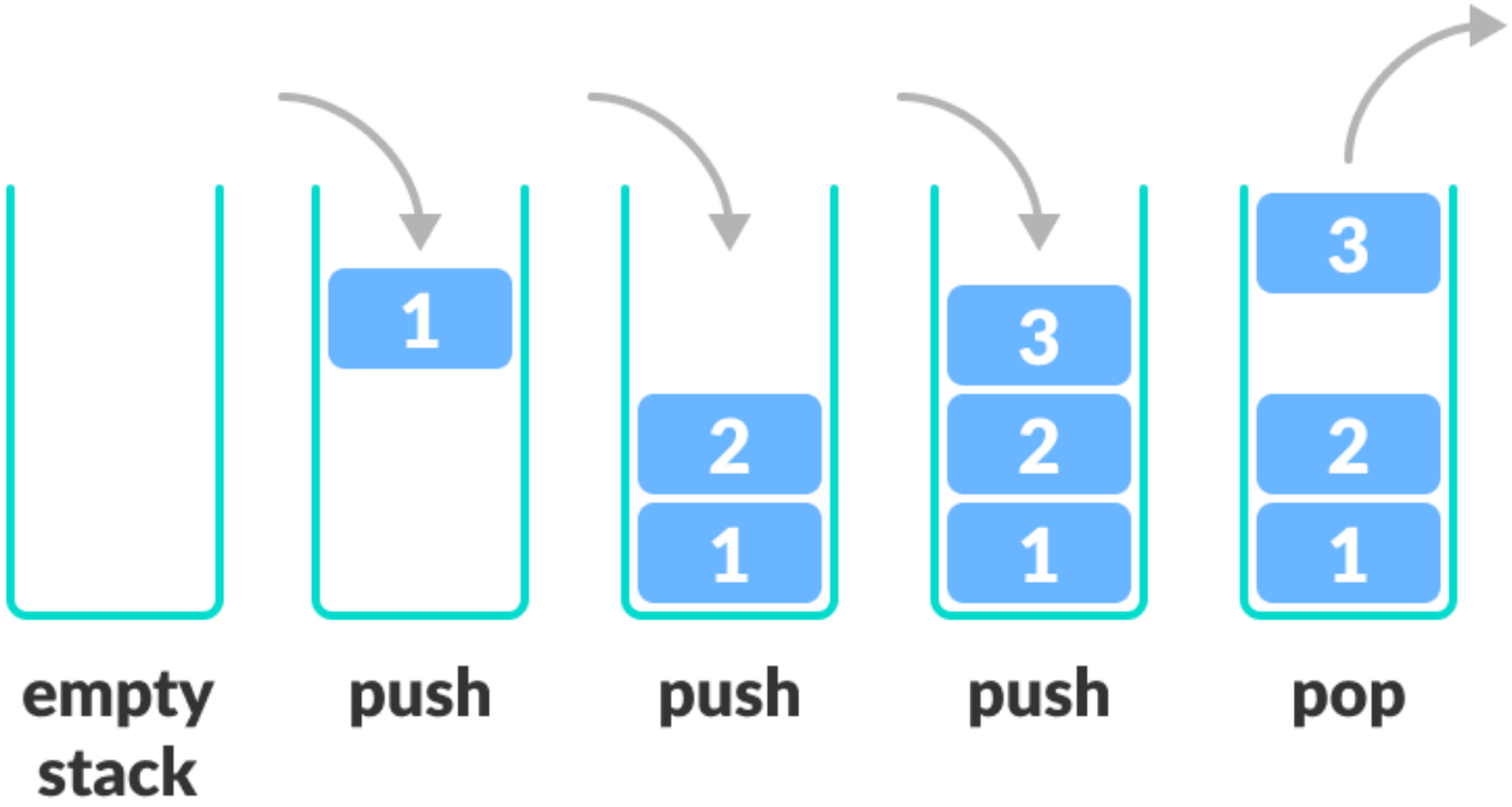
//Arquivo PilhaDinamica.c

```
int insere_pilha(Pilha* pi, struct aluno al){
    if(pi == NULL)
        return 0;
    Elem* no;
    no = (Elem*) malloc(sizeof(Elem));
    if(no == NULL)
        return 0;
    no->dados = al;
    no->prox = (*pi);
    *pi = no;
    return 1;
}
```



PILHA DINÂMICA
Remoção

PILHA DINÂMICA
Remoção



- Na “pilha” a remoção é sempre no seu topo.
- Não se pode remover em pilha vazia.

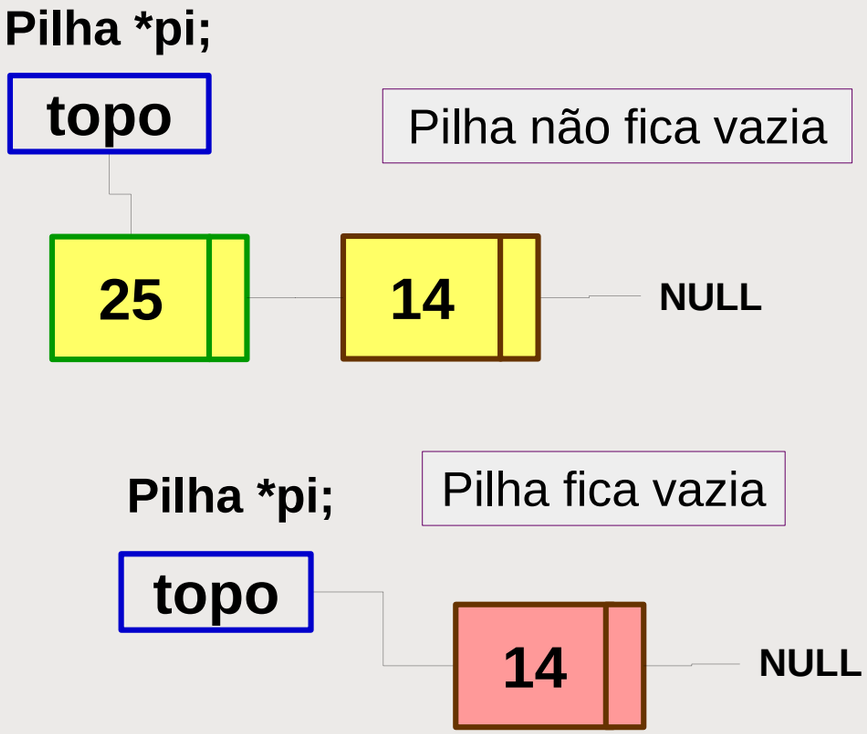
PILHA DINÂMICA: Remoção -> desempilhar (pop)

//Arquivo PilhaDinamica.h

```
int remove_pilha(Pilha* pi);
```

//main.c

```
int x = remove_pilha(pi);
```



//Arquivo PilhaDinamica.c

```
int remove_pilha(Pilha* pi){
    if(pi == NULL)
        return 0;
    if((*pi) == NULL)
        return 0;
    Elem *no = *pi;
    *pi = no->prox;
    free(no);
    return 1;
}
```

CONVERSÃO DE BASES

Decimal para Binário

Técnica

- Divida por dois, **guardando os restos**
- Primeiro resto é o **bit 0** (bit menos significativo)
- Segundo resto é o **bit 1**
- etc.

CONVERSÃO DE BASES

Decimal para Binário

$125_{10} = ?_2$

125	2
62	2
31	2
15	2
7	2
3	2
1	

1 1 1 1 1 0 1

1. PILHAS (AP3)

A conversão de números inteiros, na base 10, para outras bases numéricas se dá através de sucessivas divisões de um dado valor ***n*** pelo valor da base na qual se queira converter. Faça um programa para obter a conversão numérica, de acordo com a opção do usuário, utilizando a uma pilha:

(a) Decimal para Binário.

(b) Decimal para Octal.

(c) Decimal para Hexadecimal