FAESA
CENTRO UNIVERSITÁRIO

Pilhas

Apresentação

Pilha é uma estrutura de dados que admite remoção de elementos e inserção de novos objetos. Sempre que houver uma remoção, o elemento removido é o que está na estrutura há menos tempo.

Nesta Unidade de Aprendizagem, você vai aprender como é o funcionamento de uma pilha, criar funções e procedimentos para ela, além de ver exemplos de sua utilização.

Bons estudos.

Ao final desta Unidade de Aprendizagem, você deve apresentar os seguintes aprendizados:

- Identificar as regras de funcionamento de uma pilha LIFO: *last in, first out*.
- Criar funções e procedimentos para uma pilha.
- Praticar exemplos de utilização de pilhas.

Desafio

As estruturas de dados definem a organização, métodos de acesso e opções de processamento para coleções de itens de informação manipulados por determinado programa. Elas ajudam a estruturar, organizar, armazenar e acessar os dados.

Considere o seguinte problema.

Em determinado momento, você percebeu que guardou todos os livros que leu e que ganhou ao longo de sua vida. Com a intenção de organizá-los, criou um programa e inseriu os dados de todos os seus livros.

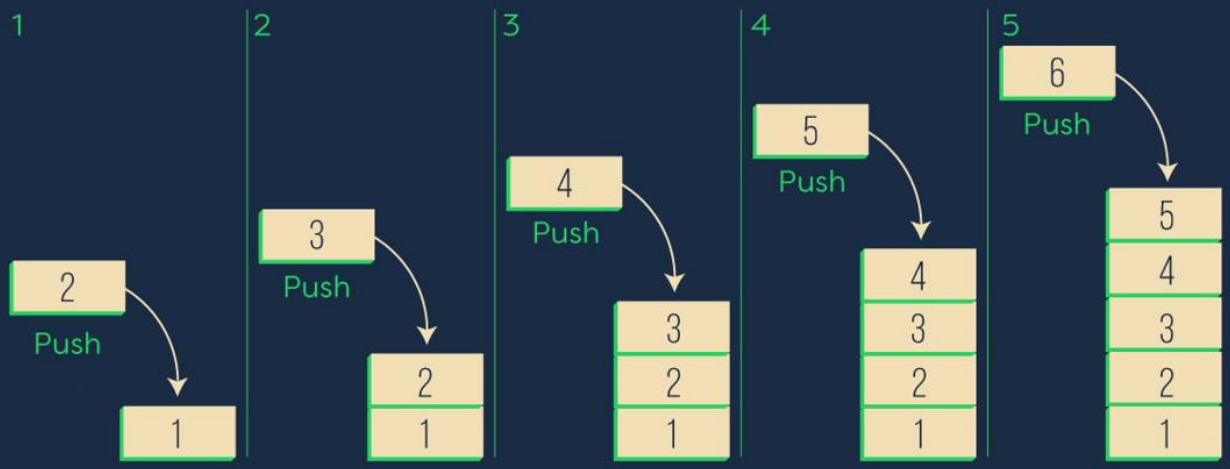
Clique aqui

Passados alguns anos, você resolveu doar alguns livros e com isso veio a necessidade de atualizar o seu programa, removendo os que foram doados. Para tanto, vai precisar escrever o código para removê-los.

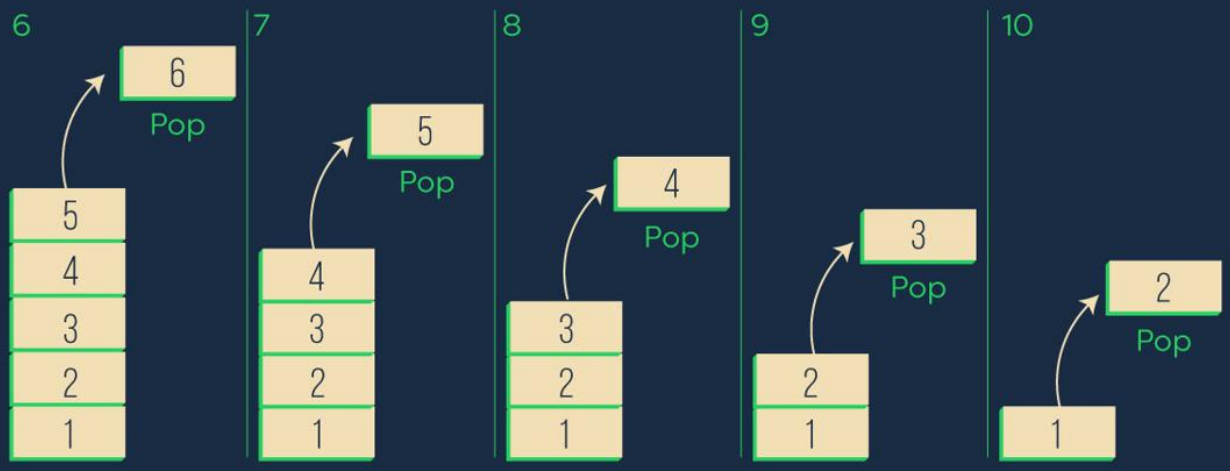
Infográfico

Vários *softwares* utilizados no dia a dia contam com estrutura de dados, porém nem sempre as pessoas se dão conta. Bons exemplos são os *softwares* aplicativos, como editores de texto, editores de planilhas, etc. A estutura de pilhas funciona como se fosse uma pilha de objetos mesmo, ou seja, você insere e retira elementos sempre pelo topo. Veja como.

OS ELEMENTOS VÃO SENDO INSERIDOS UM A UM COM O COMANDO PUSH.



E RETIRADOS COM O COMANDO POP.



ASSIM SÃO EMPILHADOS E DESEMPILHADOS SEMPRE PELO TOPO DA PILHA.

Conteúdo do Livro

Na Ciência da Computação, estrutura de dados é o modo particular de armazenamento e organização de dados no computador de maneira que possam ser usados eficientemente, facilitando sua busca e modificação.

No capítulo Pilhas, da obra *Estrutura de dados*, você vai compreender o funcionamento da estrutura que leva esse nome e serve para que o programa possa acessar informações nele armazenadas por meio de operações apropriadas.

ESTRUTURA DE DADOS

Adriana de Souza Vettorazzo

SAGAH
SOLUÇÕES EDUCACIONAIS INTEGRADAS

Pilhas

Objetivos de aprendizagem

Ao final deste texto, você deve apresentar os seguintes aprendizados:

- Identificar as regras de funcionamento de uma pilha *last in, first out* (LIFO).
- Criar funções e procedimentos de uma pilha.
- Praticar exemplos de utilização de pilhas.

Introdução

Na ciência da computação, uma estrutura de dados é um modo particular de armazenamento e organização de dados em um computador, de modo que possam ser usados eficientemente, facilitando sua busca e modificação.

Neste capítulo, você estudará as pilhas, que são estruturas de dados já definidas na linguagem de programação. Essas estruturas servem para que o programa possa acessar informações neles armazenados, por meio de operações apropriadas.

Pilha

Pilhas são estruturas de dados do tipo *last in*, *first out* (LIFO), ou seja, o último elemento a ser inserido na estrutura, será o primeiro a ser retirado da estrutura.

Podemos fazer uma comparação com uma pilha de pratos, em que, se quisermos adicionar um prato na pilha, devemos colocá-lo topo, e, para pegar um prato da pilha, retiramos o do topo. Dessa forma, temos que retirar o prato do topo para ter acesso ao próximo prato.

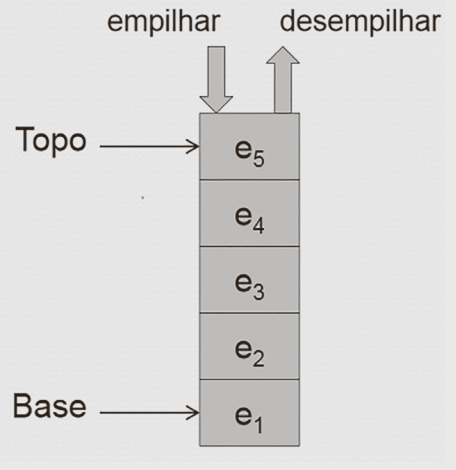
Portanto, essa manipulação é feita apenas por uma das extremidades da lista, pelo topo. Para processar/acessar o penúltimo item da estrutura, deve-se remover o último item. Observe a representação de pilhas na Figura 1.

(a) (b)

Figura 1. (a) Pilha de livros, (b) Pilha de carta de baralho.
Fonte: (a) Billion Photos/Shutterstock.com, (b) Wikipedia.

Fique atento

Na implementação de uma pilha, em apenas uma das extremidades, que chamamos de topo, é possível realizar a manipulação dos dados. A outra extremidade da estrutura chamamos de base.



Quanto ao tipo de alocação, as pilhas podem ser estáticas ou dinâmicas.

Implementando uma pilha estática

Para manipular os dados em uma pilha estática, você pode usar as seguintes operações:

- criação (*pull*);
- inserção (*push*) (Figura 2);
- remoção (*pop*) (Figura 2);
- acessa elemento (*top*).

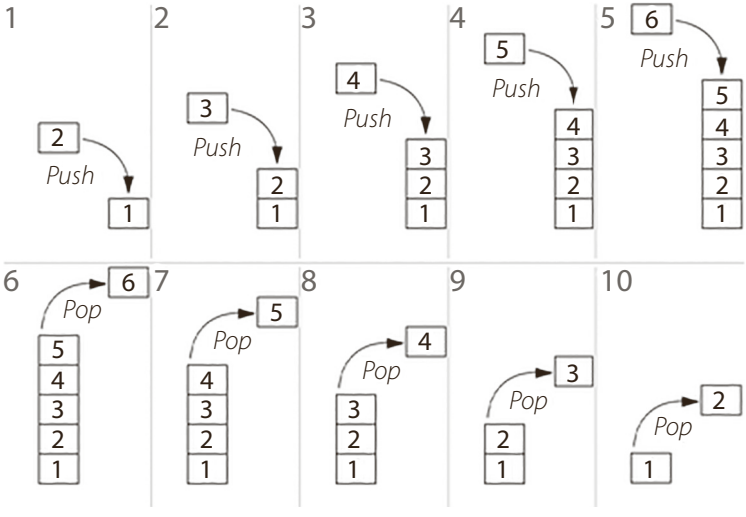


Figura 2. Operações *push* e *pop*.

Criação

Na alocação estática, o espaço de memória é alocado no momento da compilação, definindo o número máximo de elementos da pilha, e o acesso é sequencial.

Exemplo:

```
struct aluno{
        int matricula;
        char nome [30];
        float n1, n2, n3;
};
Typedef struct pilha Pilha;
struct pilha{
        int qtd;
        struct aluno dados [MAX];
};

Pilha *pi;

Pi=cria_Pilha();

Pilha* cria_Pilha (){

        Pilha *pi;

        pi = (Pilha*) malloc (sizeof(struct pilha));

        if (pi !=NULL)

                pi -> qtd = 0;

        return pi;

libera_Pilha(pi);

void libera_Pilha* pi);

void libera_Pilha (Pilha* pi){

        free (pi) ;

}
```

Inserção

```
int x = insere_Pilha (pi, dados_aluno);
int insere_Pilha (Pilha* pi, struct aluno al);
int insere_Pilha (Pilha* pi, struct aluno al) {
        if (pi == null) return 0;
        if (Pilha_cheia (pi)) return 0;
        pi - > dados [pi -> qtd] = al;
        pi -> qtd++;
        return 1;
```

Remoção

```
int x = remove_Pilha (pi);
int remove_Pilha (Pilha* pi);
int remove_Pilha (Pilha* p){
        if (pi == NULL | | pi -> qtd == 0)
                return 0;
        pi -> qtd --;
        return 1;
}
```

Acessar

```
int x = consulta_topo_Pilha (pi, &dados_aluno);
int Consulta_topo_Pilha (Pilha* pi, struct aluno *al);
int consulta_ipo_Pilha (Pilha* pi, struct aluno *al){
        if (pi == NULL | | pi -> qtd == 0)
                return 0;
        *al = pi -> dados [pi -> qtd-1];
        Return 1;
```

Saiba mais

Em uma pilha, a inserção e a remoção são sempre feitas no seu início, mas também temos o caso de inserção em uma pilha vazia, pois não se pode inserir em uma pilha que está cheia.

Implementando uma pilha dinâmica

Para pilha dinâmica, trabalhamos com ponteiros, pois eles sempre apontam para o seu sucessor na pilha.

Criando uma pilha dinâmica

```
Pi= cria_Pilha ();
Pilha* cria_Pilha();
Pilha* cria_Pilha (){
        Pilha* pi = (Pilha*) malloc (sizeof (Pilha));
        If (pi != NULL)
                *pi = NULL;
        Return pi;
```

Liberando a pilha

```
libera_Pilha (pi);
void libera_Pilha (Pilha* pi);
void libera_Pilha (Pilha* pi) {
        if (pi != NULL ){
                Elem* no;
                While (( *pi) != NULL)[
                        no = *pi;
                        *pi = (*pi) -> prox;
                        Free (no);
                }
                Free (pi);
        }
}
```

Inserção, remoção e consulta de valores em uma pilha dinâmica

```
//Inserção
int x = insere_Pilha (pi, dados_aluno);
int insere_Pilha (Pilha* pi, struct aluno al);
int insere_Pilha (Pilha* pi, struct aluno al){
        if (pi == NULL) return 0;
        Elem* no = (Elem*) malloc (sizeof (Elem));
        if (no ==NULL) return 0;
        no -> dados = al;
        no -> prox = (*pi);
        *pi = no;
        return 1;
```

```
//Remoção
int x = remove_Pilha (pi);
int remove_Pilha (Pilha* pi);
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;
}
```

```
//Consulta
int x = consulta_topo_Pilha (pi, &dados_aluno);
int consulta_topo_Pilha (Pilha* pi, struct aluno *al);
int consulta_topo_Pilha (Pilha* pi, struct aluno *al) {
        if (pi == NULL) return 0;
        if (( *pi) == NULL) return 0;
        *al = (*pi)-> dados;
        return 1;
```

Fique atento

Na alocação estática, o tamanho da pilha já é definido na sua criação. Quando a alocação é dinâmica, a pilha pode ter um tamanho inicial, porém é possível alocar mais elementos, quando necessário, e o tamanho da pilha expande dinamicamente.

Exemplo

Veja a seguir um exemplo de pilha de números reais.

```
#include
struct Pilha {

        int topo; /* posição elemento topo */
        int capa;
        float *pElem;

};
void criarpilha( struct Pilha *p, int c ){

  p->topo = -1;
  p->capa = c;
  p->pElem = (float*) malloc (c * sizeof(float));

}
int estavazia ( struct Pilha *p ){

  if( p-> topo == -1 )

    return 1;   // true

  else

    return 0;   // false

}
```

```
int estacheia ( struct Pilha *p ){

        if (p->topo == p->capa - 1)

                return 1;

        else

                return 0;

}
void empilhar ( struct Pilha *p, float v){

        p->topo++;
        p->pElem [p->topo] = v;

}

float desempilhar ( struct Pilha *p ){

  float aux = p->pElem [p->topo];
  p->topo--;
  return aux;

}

float retornatopo ( struct Pilha *p ){

  return p->pElem [p->topo];

}

int main(){

        struct Pilha minhapilha;
        int capacidade, op;
        float valor;

        printf( "\nCapacidade da pilha? " );
        scanf( "%d", &capacidade );

        criarpilha (&minhapilha, capacidade);
```

```
        while( 1 ){ /* loop infinito */

                printf("\n1- empilhar (push)\n");
                printf("2- desempilhar (POP)\n");
                printf("3- Mostrar o topo \n");
                printf("4- sair\n");

printf("\nopcao? ");
scanf("%d", &op);

switch (op){

        case 1: //push

                if( estacheia( &minhapilha ) == 1 )

                        printf("\nPILHA CHEIA! \n");

                else {

                        printf("\nVALOR? ");
                        scanf("%f", &valor);
                        empilhar (&minhapilha, valor);

                }
                break;

        case 2: //pop
                if ( estavazia(&minhapilha) == 1 )

                        printf( "\nPILHA VAZIA! \n" );

                else{

                        valor = desempilhar (&minhapilha);
                        printf ( "\n%.1f DESEMPILHADO!\n", valor );

                }
                break;

        case 3: // mostrar o topo
                if ( estavazia (&minhapilha) == 1 )

                        printf( "\nPILHA VAZIA!\n" );
```

```
                    else {

                        valor = retornatopo (&minhapilha);
                        printf ( "\nTOPO: %.1f\n", valor );

                    }
                    break;

                case 4:
                    exit(0);

                default: printf( "\nOPCAO INVALIDA! \n");
            }
        }

}
```

Leituras recomendadas

SHACKELFORD, R. L. *Introduction to computing and algorithms*. Boston: Addison-Wesley, 1997. 399 p.

TENENBAUM, A. M.; LANGSAM, Y.; AUGENSTEIN, M. J. *Estruturas de dados usando C*. São Paulo: Makron Books, 1995. 904 p.

Encerra aqui o trecho do livro disponibilizado para esta Unidade de Aprendizagem. Na Biblioteca Virtual da Instituição, você encontra a obra na íntegra.

Conteúdo:

SAGAH
SOLUÇÕES EDUCACIONAIS INTEGRADAS

Dica do Professor

O vídeo apresenta as diferenças entre a pilha estática e pilha dinâmica, além de indicar qual a melhor aplicação para cada uma delas.

Aponte a câmera para o código e acesse o link do conteúdo ou clique no código para acessar.

Exercícios

1) **No contexto estrutura de dados, pilha é:**

A) Uma lista LILO.

B) Uma lista FIFO.

C) Um tipo de lista linear em que o último elemento a ser inserido é o primeiro retirado.

D) Um tipo de lista linear em que as operações de inserção e remoção são realizadas aleatoriamente.

E) Um tipo de lista linear em que as operações de inserção são realizadas em determinada extremidade e as operações de remoção são realizadas em outra.

2) **Em estruturas de dados, é encontrada a estrutura pilha. Avalie as assertivas abaixo e identifique a alternativa correta.**

I. Para excluir (remover ou desempilhar) o elemento da pilha, basta excluir o elemento para o qual aponta o ponteiro de início. Esta operação permite recuperar o dado no topo da pilha, e também removê-lo.

II. Uma das possíveis utilizações de uma pilha é a implementação da sequência de desfazer (Ctrl + Z) de um editor de texto.

III. Na estrutura pilha, o último elemento a entrar também é o último a sair.

IV. Na pilha, as operações de exclusão e inclusão são realizadas na mesma extremidade, chamada topo.

V. As operações de exclusão e inclusão são realizadas em qualquer parte da pilha.

Assinale a alternativa correta:

A) Somente a I e III

B) Somente a II e IV

C) Somente a III e IV.

D) Somente a I e II

E) Somente a I e V.

3) Assinale a opção correta relativa às operações básicas suportadas por pilhas.

A) PUSH insere um novo elemento na base da pilha.

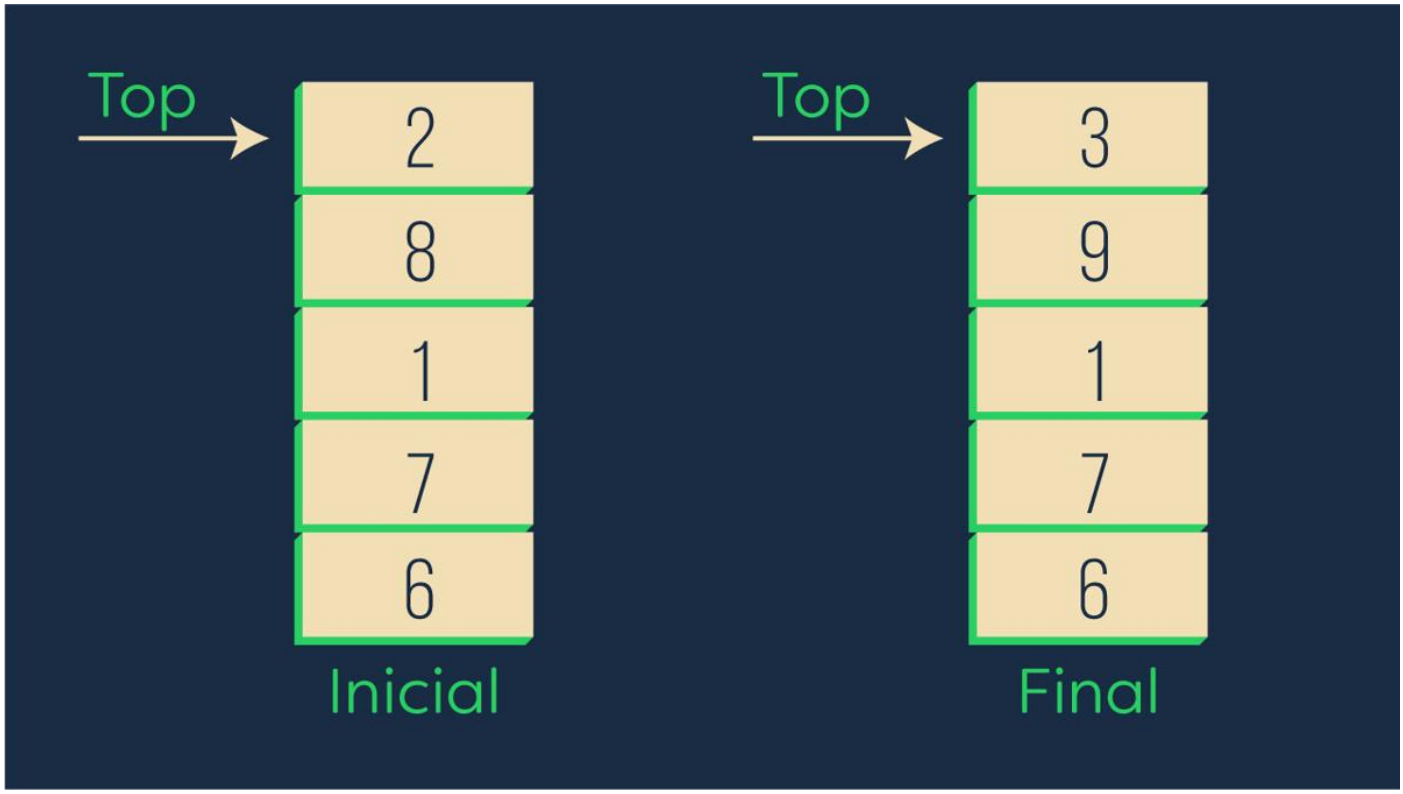
B) *PUSH* coloca um elemento no topo da pilha.

C) TOP transfere o último elemento para o topo da pilha.

D) POP adiciona elementos ao topo da pilha.

E) *PULL* altera o elemento no final da pilha.

4) Considere os estados (inicial e final) da pilha a seguir, na qual *top* corresponde ao seu topo.



Para atingir o estado final dessa pilha, deve-se usar a seguinte sequência de operações básicas:

A) pop(), pop(), push(9), push(3).

B) push(2), push (8), pop(), pop().

C) push(), push(), pop(8), pop(2).

D) pop(3), pop (9), push(), push().

E) pop(9), pop (3), push(), push().

5) Considere que os itens W, X, Y, Z e K foram inseridos nessa ordem em uma pilha. Necessariamente, o último elemento é:

A) W.

B) X.

C) Y.

D) Z.

E) K.

Na prática

As estruturas de dados estão presentes em muitos momentos do dia das pessoas. A estrutura pilha é uma das mais utilizadas.

Quando o *software* aplicativo é iniciado, por exemplo, é criada também uma pilha para armazenar as ações que estão sendo executadas. A cada caractere digitado, formatação inserida ou qualquer outra ação durante o uso do *software* aplicativo, essa ação é colocada dentro de uma pilha, ou seja, as informações estão sendo empilhadas.

Conteúdo interativo disponível na plataforma de ensino!

Saiba mais

Para ampliar o seu conhecimento a respeito desse assunto, veja abaixo as sugestões do professor:

Pilha Dinâmica – Inserção e Remoção

Entenda o funcionamento de uma Pilha a partir da programação descomplicada.

Aponte a câmera para o código e acesse o link do conteúdo ou clique no código para acessar.

Como fazer pilhas na linguagem C

Saiba mais sobre as funções com Pilhas em C.

Aponte a câmera para o código e acesse o link do conteúdo ou clique no código para acessar.

Listas, pilhas e filas

Conheça outros tipos de Estrutura de Dados.

Aponte a câmera para o código e acesse o link do conteúdo ou clique no código para acessar.