

SIN 211 - Algoritmos e Estruturas de Dados

(Pilha Dinâmica)

Profº: Joelson Antônio dos Santos

Universidade Federal de Viçosa
Instituto de Ciências Exatas e Tecnológicas
Campus de Rio Paranaíba - MG

joelsonn.santos@gmail.com
Sala: BBT 233

26 de abril de 2018

Aula de Hoje

- 1 Pilha Dinâmica
 - Implementação
 - Aplicações

O material desta aula é composto por adaptações e extensões dos originais gentilmente cedidos pelos professores **Moacir Pereira Ponti** e **Rachel Reis**.

Pilha Dinâmica

- A implementação se assemelha à lista encadeada.
- Para onde o ponteiro externo deve apontar nesta implementação?
- Como implementamos cada elemento de uma **lista**?

Pilha Dinâmica - definição(1)

- Esta é uma das maneiras existentes de definição e declaração de pilha.

Pilha Dinâmica - definição(1)

- Esta é uma das maneiras existentes de definição e declaração de pilha.

definição

```
typedef struct sCell{  
    int info;  
    struct sCell* prox;  
}CELULA;
```

declaração

```
int main(){  
  
    CELULA* ptrTopo;
```

ptrTopo

?

Pilha Dinâmica - definição(1)

- Esta é uma das maneiras existentes de definição e declaração de pilha.

definição

```
typedef struct sCell{  
    int info;  
    struct sCell* prox;  
}CELULA;
```

declaração

```
int main(){  
  
    CELULA* ptrTopo;
```

ptrTopo

?

- Qual a diferença de uma **lista simplesmente encadeada**?

Pilha Dinâmica - definição(2)

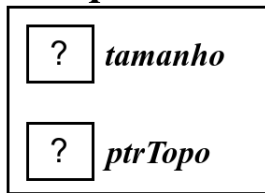
definição

```
typedef struct sCell{  
    int info;  
    struct sCell* prox;  
}CELULA;  
  
typedef struct sPilha{  
    int tamanho;  
    CELULA* ptrTopo;  
}PILHA;
```

declaração

```
int main(){  
  
    PILHA pilha;
```

pilha



- O que representa os campos *ptrTopo* e *tamanho*?

Pilha Dinâmica - definição(2)

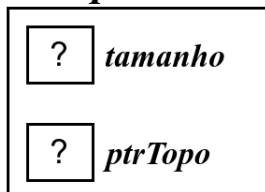
definição

```
typedef struct sCell{  
    int info;  
    struct sCell* prox;  
}CELULA;  
  
typedef struct sPilha{  
    int tamanho;  
    CELULA* ptrTopo;  
}PILHA;
```

declaração

```
int main(){  
  
    PILHA pilha;
```

pilha



- O que representa os campos *ptrTopo* e *tamanho*?
- Quais as vantagens desta definição em relação a do slide anterior?

Pilha Dinâmica

- Quais são as operações usuais de uma pilha dinâmica?

Pilha Dinâmica

- Inicializar.
- Verificar se pilha está vazia.
- Acessar topo sem removê-lo.
- *push* (empilhar).
- *pop* (desempilhar).
- Imprimir.
- Esvaziar a pilha inteira.

Pilha Dinâmica - Operações

- E se utilizarmos a definição(2) apresentada no **slide 6**, como seria a inicialização da pilha?

Pilha Dinâmica - Operações

- E se utilizarmos a definição(2) apresentada no **slide 6**, como seria a inicialização da pilha?

```
void inicializarPilha(PILHA* pilha){  
    pilha->tamanho = 0;  
    pilha->ptrTopo = NULL;  
}
```

Pilha Dinâmica - Operações

- E se utilizarmos a definição(2) apresentada no **slide 6**, como seria a inicialização da pilha?

```
void inicializarPilha(PILHA* pilha){  
    pilha->tamanho = 0;  
    pilha->ptrTopo = NULL;  
}
```

- Como seria a chamada da função `inicializarPilha(PILHA* pilha)` na função *main*?

Pilha Dinâmica - Operações

- Verificar se a pilha está vazia:

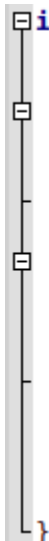
Pilha Dinâmica - Operações

- Verificar se a pilha está vazia:
 - Deve verificar se o ponteiro externo à pilha aponta ou não para **NULL**.

Pilha Dinâmica - Operações

- Verificar se a pilha está vazia:
 - Deve verificar se o ponteiro externo à pilha aponta ou não para **NULL**.
- Como seria a implementação desta função?

Pilha Dinâmica - Operações - *Push*

A vertical diagram on the left side of the slide represents a stack. It consists of a vertical line with three square boxes at different heights, each containing a minus sign (-). Horizontal tick marks extend from the line between the boxes and at the bottom, indicating the structure of the stack.

```
int push(PILHA* pilha, int elemento){
    CELULA* p = criarCelula();

    if(p == NULL){
        printf("\n Erro de alocao!");
        return 0;
    }
    p->info = elemento;
    if(pilhaVazia(pilha)){
        pilha->ptrTopo = p;
        pilha->tamanho++;
        return 1;
    }
    p->prox = pilha->ptrTopo;
    pilha->ptrTopo = p;
    pilha->tamanho++;
    return 1;
}
```

Pilha Dinâmica - Operações

- E como implementar a função *pop* (desempilhar)?

Pilha Dinâmica - Operações

- E como implementar a função *pop* (desempilhar)?
- Podemos relacioná-la a função de remover no início de uma **lista simplesmente encadeada**?

Pilha Dinâmica - Operações

- Acessar o topo da pilha
 - Função utilizada para retornar/imprimir o elemento que está no topo da pilha sem removê-lo.
- Imprimir e esvaziar a lista
 - Imprime o conteúdo do topo e o remove da pilha. Repete o procedimento até que a pilha fique vazia.
- Com seriam as implementações das funções correspondentes aos itens anteriores?

Aplicações - Notação pós-fixa

- O operador é expresso após os seus operandos.
- Dispensa o uso de parênteses.
- Exemplos: infixa \rightarrow pós-fixa
 - $a * b + c \rightarrow a b * c +$
 - $a * (b + c) \rightarrow a b c + *$

Aplicações - Notação pós-fixa

- Considerando a notação pós-fixa de alguma expressão matemática, o algoritmo de pilha para resolvê-la pode ser descrito como:
 - 1 Lê todos (um a um) os símbolos de uma expressão e:
 - 2 Se operando:
 - *push* for operando
 - 3 Se for operador:
 - 1 *pop* número adequado de operandos;
 - 2 Realiza a operação.
 - 3 *push* resultado

Aplicações - Notação pós-fixa - Exemplo

- Considere a expressão pós-fixa: $a = 3\ 2\ +\ 1\ 5\ +\ *$

Tabela: Operações sobre a em uma pilha hipotética.

iteração	símbolo	pop1	pop2	operação	push
1 ^a	3				3
2 ^a	2				3,2
3 ^a	+	2	3	5	5
4 ^a	1				5,1
5 ^a	5				5,1,5
6 ^a	+	5	1	6	5,6
7 ^a	*	6	5	30	30

- Os números em vermelho representam o topo da pilha em cada iteração.

Aplicações - Casamento de Delimitadores

- Utilizado por compiladores.
- Nenhum programa é considerado correto se esse casamento estiver errado.
- Dados armazenados e posteriormente, recuperados em ordem inversa.
- $\{\}$, $[]$, $()$
- **Exemplos:**
 - $a = b + (c - d) * (e - f);$
 - $A[i] = B[C[i]] + (j + h);$

Aplicações - Casamento de Delimitadores

- Delimitadores podem ser aninhados.
- Um delimitador em particular está casado somente depois que todos os delimitadores que o seguem e que o precede tenham sido casados!
- **Exemplo:** $\text{while}(m < n[8] + o)$

Aplicações - Casamento de Delimitadores

- Considerando um conjunto de delimitadores, o algoritmo para verificar o casamento entre cada um deles pode ser descrito como:
 - 1 Lê todos (um a um) os caracteres de um arquivo, conjunto ou expressão e:
 - 2 Se for um delimitador de abertura:
 - *push* for delimitador.
 - 3 Se for um delimitador de fechamento:
 - Se houver casamento com topo da pilha:
 - *pop* e continua.
 - Senão:
 - Erro e encerra.
 - 4 Retorna ao passo 2 até que chegue no final do arquivo/conjunto/expressão.

Aplicações - Casamento de Delimitadores

- Considere os delimitadores: $\mathbf{a} = \{[(())]\}$

Tabela: Operações sobre \mathbf{a} em uma pilha hipotética.

iteração	símbolo	pop	estado da pilha
1ª	{		{
2ª	[{, [
3ª	({, [, (
4ª)	({, [
5ª]	[{
6ª	}	{	

- Os delimitadores em vermelho representam o topo da pilha em cada iteração.

Bibliografia Básica

- DROZDEK, Adam. Estrutura de Dados e Algoritmos em C++. Editora Pioneira Thomson Learning, 2005. Capítulo 4.
- Notação Posfixa. Ivan L. M. Ricarte (2003)
<http://www.dca.fee.unicamp.br/cursos/EA876/apostila/HTML/node75.html>