Vinícius do Amaral Brunheroto

Estrutura de dados I

Profº Carlos Fischer

Unesp RC

Existem duas formas principais para fazer implementações em uma pilha usando ponteiros.

A primeira delas é fazer uma pilha em um arranjo ou vetor com tamanho predefinido, controlando a posição do elemento que está no topo.

Para fazer as implementações, controles e gerenciamento dessa pilha será usando um ponteiro para ela e essas operações poderão ser feitas dentro de subrotinas ou funções.

Para o programa, há duas structs definidas: uma conhecida como registro(que conterá o campo chave) e outra conhecida como pilha que conterá a posição do topo e o vetor de struct registro.

Para a operação de inserção, a ideia é ir na posição seguinte ao topo e inserir o elemento, dessa forma há um novo topo na pilha.

Como parâmetros para a função estão um ponteiro para a pilha(informando o endereço) e um registro a ser usado na pilha.

Como retorno pode-se usar um bool para indicar true se a inserção ocorreu e false se não ocorreu.

Primeiro verifica-se se a pilha está ou não cheia, através da posição do último elemento comparando-se com a última posição declarada no arranjo ou vetor, se estiver cheia retorna false.

Se ela não estiver cheia, incrementa em um o valor de topo e coloca-se no novo valor de topo, o registro que foi passado como parâmetro.

Por fim, retorna true se foi feita a inserção.

ANTES:

A	5	2	7	?	?
TOPO	POSIÇÃO 2				

DEPOIS:

A	5	2	7	**8**	?

TOPO	**POSIÇÃO 3**

Para a operação de exclusão, geralmente não é informado qual elemento se quer excluir, pois a forma padrão de funcionamento da pilha é excluir sempre o elemento que está no topo, então deve-se olhar para o elemento que está no topo e exclui-lo e diminuir 1 na variável topo.

Como parâmetros para a função também é passado um ponteiro para a pilha(informando o endereço) e opcionalmente, pode ser feito um ponteiro para a struct registro para informar qual registro foi retirado.

Primeiro verifica-se se a pilha está ou não vazia perguntando se o topo vale -1 e retorna false se ela estiver vazia.

O ponteiro para registro recebe o valor a ser retirado, diminui-se 1 no topo e retorna true se foi feita a exclusão do elemento.

ANTES:

A	5	2	7	8	?

TOPO	POSIÇÃO 3

DEPOIS:

A	5	2	7	**?**	?

TOPO	**POSIÇÃO 2**

A segunda maneira de implementar uma pilha é semelhante ao modelo de uma lista encadeada.

Dessa forma cada elemento indicará quem é seu sucessor (quem está “abaixo” dele na pilha).

Será controlado o endereço do elemento que está no topo da pilha.

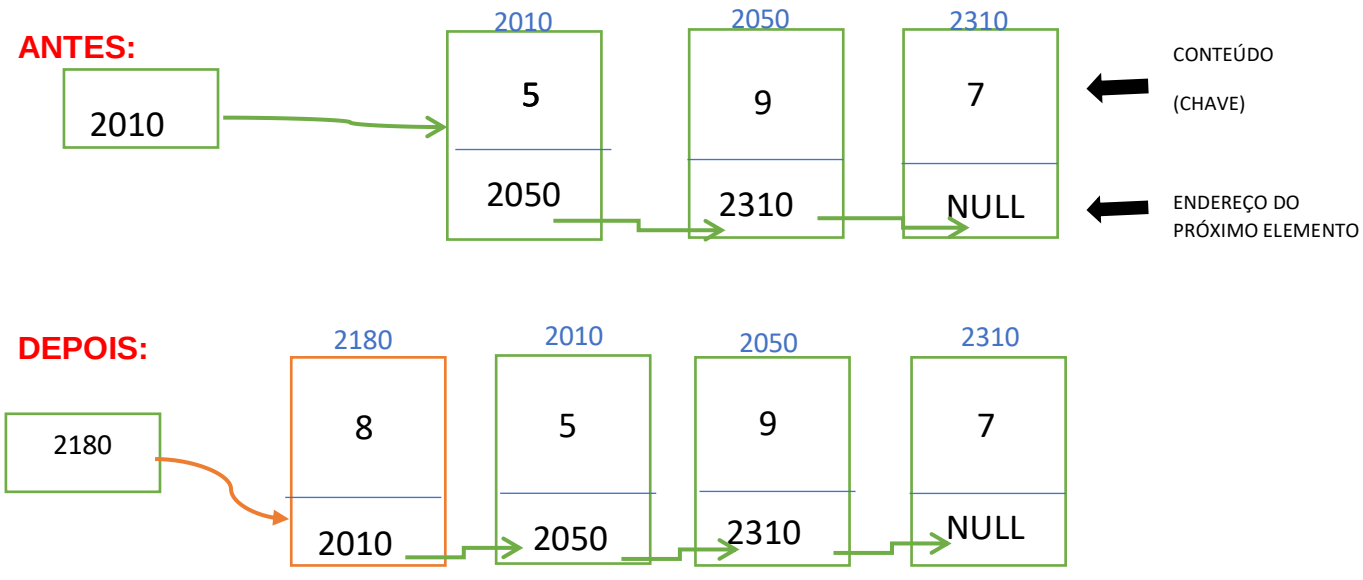
Para o programa, há três structs definidas: uma conhecida como registro(que conterá o campo chave) , outra conhecida como elemento que conterá uma variável para struct registro e um ponteiro para o próximo elemento, por fim a struct pilha que conterá um ponteiro do tipo elemento para indicar o topo da pilha.

Para a operação de inserção, aloca memória para novo elemento e esse novo elemento precisa apontar para quem estava no topo da pilha anteriormente e o campo topo da pilha agora precisa apontar pra quem foi inserido.

Como parâmetros da função há o registro a ser inserido na pilha e um ponteiro para pilha, aloca-se memória dinamicamente para o novo elemento e joga-se o endereço em uma variável qualquer.

Copiar o registro passado como parâmetro nesse novo elemento alocado e indicar que o próximo elemento é quem estava no topo da pilha.

Por fim, aponta-se o campo topo apontar para essa variável alocada(com registro novo).



Para a operação de exclusão, geralmente não é informado qual elemento se quer excluir, pois a forma padrão de funcionamento da pilha é excluir sempre o elemento que está no topo, então deve-se liberar a memória do elemento do topo e fazer o campo topo da pilha apontar para o próximo elemento.

Como parâmetros da função, há o ponteiro para a pilha e um ponteiro para registro(indicar o elemento que será excluído).

Primeiro verifica se a pilha está vazia e se estiver retorna false.

Caso contrário, no endereço apontado do registro será armazenado o valor a ser retirado.

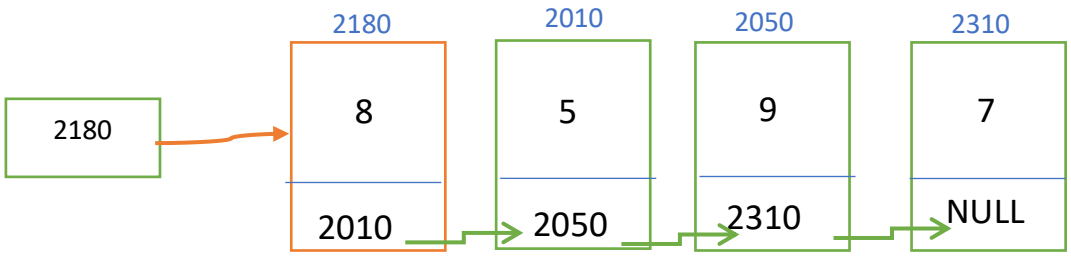
Em seguida, uma variável auxiliar é criada para guardar o endereço de memória que se quer liberar(o endereço de quem está no topo).

O campo topo aponta para o próximo elemento (seguinte).

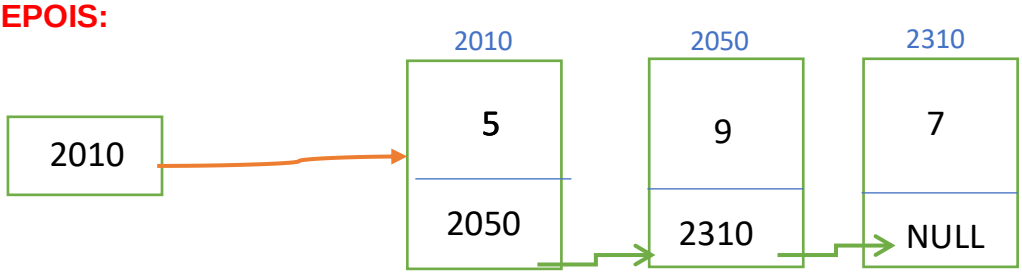
Libera-se a memória do registro que está sendo excluído.

Retorna-se true se foi possível excluir o elemento.

ANTES:



DEPOIS:



Vantagens: qual forma de implementação tem uma aplicação melhor?

Entre essas duas formas, aquela que aloca e desaloca elementos da memória segundo demanda não precisa gastar memória que não está sendo usada, então ela apresenta uma maior economia de memória em relação à pilha construída em um vetor.

Além disso,há maior flexibilidade quanto ao tamanho, pois a pilha construída de forma encadeada pode inserir infinitos elementos(se necessário) já a pilha construída em um vetor tem sempre um tamanho limite e fixo que precisa ser respeitado.