Estrutura de Dados
Estrutura de dados Linear: Pilha

Alessandro Ferreira Leite

26/03/2012

Introdução

- Uma das estruturas de dados mais simples.
- É a estrutura de dados mais utilizada em programação.
- É uma metáfora emprestada do mundo real, que a computação utiliza para resolver muitos problemas de forma simplificada.

Definição

Definição

Um conjunto ordenado de itens no qual novos itens podem ser inseridos e a partir do qual podem ser eliminados em uma extremidade denominada topo da pilha.

Definição

Definição

Um conjunto ordenado de itens no qual novos itens podem ser inseridos e a partir do qual podem ser eliminados em uma extremidade denominada topo da pilha.

Definição

Uma seqüência de objetos, todos do mesmo tipo, sujeita às seguintes regras de comportamento:

1. Sempre que solicitado a remoção de um elemento, o elemento removido é o último da seqüência.
2. Sempre que solicitado a inserção de um novo elemento, o objeto é inserido no fim da seqüência (topo).

Pilha

- Uma pilha é um objeto dinâmico, constantemente mutável, onde elementos são inseridos e removidos.
- Em uma pilha, cada novo elemento é inserido no topo.
- Os elementos da pilha só podem ser retirado na ordem inversa à ordem em que foram inseridos
  - O primeiro que sai é o último que entrou.
  - Por essa razão, uma pilha é dita uma estrutura do tipo: LIFO(*last-in, first* ou UEPS último a entrar é o primeiro a sair.)

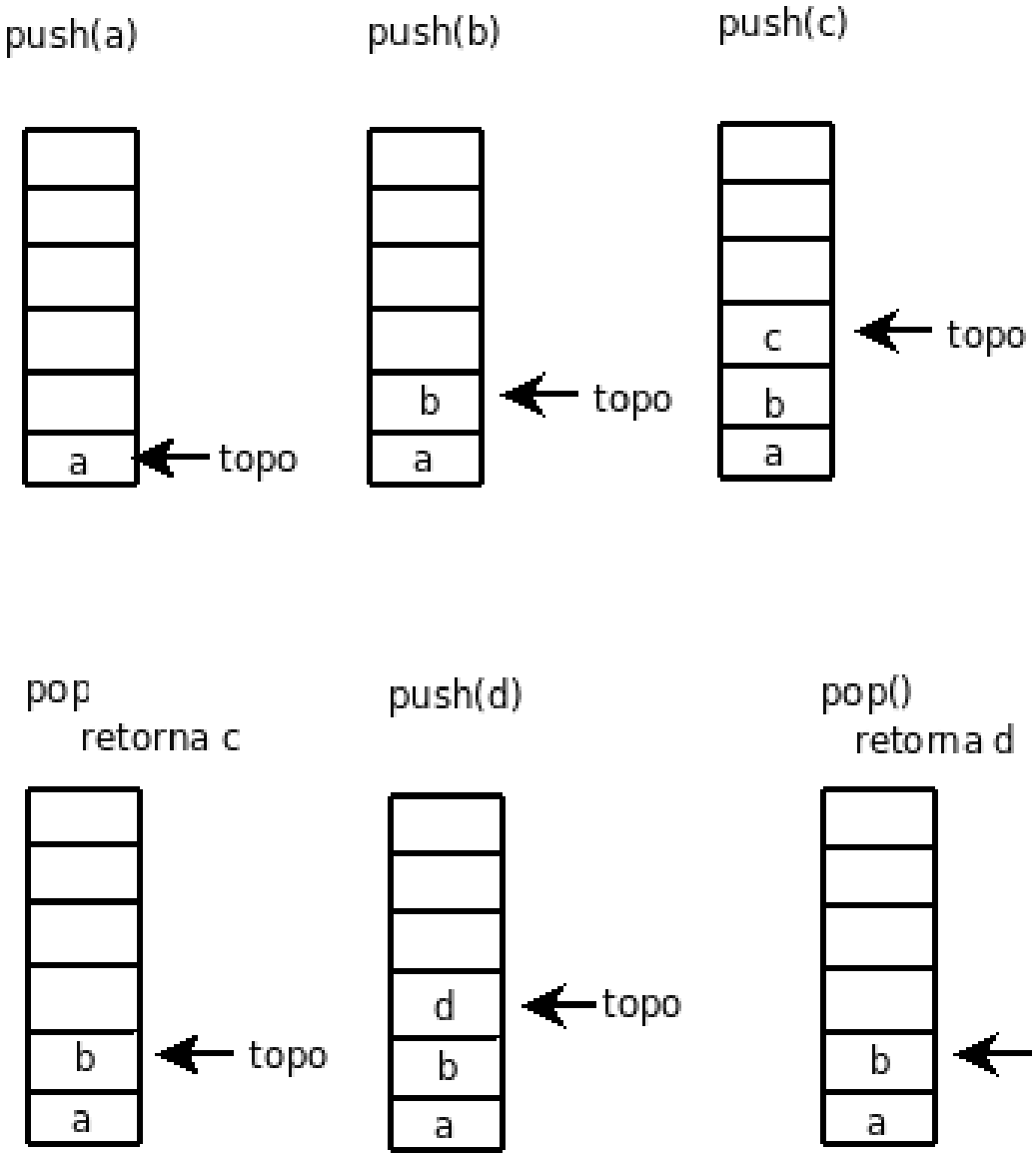
Operações básicas

- As operações básicas que devem ser implementadas em uma estrutura do tipo pilha são:

Operação	**Descrição**
push(p, e)	empilha o elemento *e*, inserindo-o no topo da pilha *p*.
pop(p)	desempilha o elemento do topo da pilha *p*.

Tabela: Operações básicas da estrutura de dados pilha.

Exemplo


push(a)
a
topo
push(b)
b
topo
a
push(c)
c
topo
b
a
pop
retorna c
b
topo
a
push(d)
d
topo
b
a
pop()
retoma d
b
topo
a

Operações auxiliares

- Além das operações básicas, temos as operações "auxiliares". São elas:

Operação	**Descrição**
create	cria uma pilha vazia.
empty(p)	determina se uma pilha *p* está ou não vazia.
free(p)	libera o espaço ocupado na memória pela pilha *p*.

Tabela: Operações auxiliares da estrutura de dados pilha.

Interface do Tipo Pilha

```c
/* Definição da estrutura */
typedef struct pilha Pilha;
/*Aloca dinamicamente a estrutura pilha, inicializando
 *seus campos e retorna seu ponteiro.*/
Pilha* create(void);

/*Insere o elemento e na pilha p.*/
void push(Pilha *p, int e);

/*Retira e retorna o elemento do topo da pilha p*/
int pop(Pilha *p);

/*Informa se a pilha p está ou não vazia.*/
int empty(Pilha *p);
```

Implementação de Pilha com Vetor

- Normalmente as aplicações que precisam de uma estrutura pilha, é comum saber de antemão o número máximo de elementos que precisam estar armazenados simultaneamente na pilha.
- Essa estrutura de pilha tem um limite conhecido.
- Os elementos são armazenados em um vetor.
- Essa implementação é mais simples.
- Os elementos inseridos ocupam as primeiras posições do vetor.

Implementação de Pilha com Vetor

- Seja p uma pilha armazenada em um vetor VET de N elementos:
  1. O elemento vet[topo] representa o elemento do topo.
  2. A parte ocupada pela pilha é vet[0 .. topo - 1].
  3. A pilha está vazia se topo = -1.
  4. Cheia se topo = N - 1.
  5. Para desempilhar um elemento da pilha, não vazia, basta

$$x = vet[topo - -]$$

.

  6. Para empilhar um elemento na pilha, em uma pilha não cheia, basta

$$vet[t + +] = e$$

.

Implementação de Pilha com Vetor

```c
#define N 20 /* número máximo de elementos*/
#include <stdio.h>
#include "pilha.h"

/*Define a estrutura da pilha*/
struct pilha{
   int topo; /* indica o topo da pilha */
   int elementos[N]; /* elementos da pilha*/
};

Pilha* create(void){
   Pilha* p = (Pilha*) malloc(sizeof(Pilha));
   p->topo = -1; /* inicializa a pilha com 0 elementos
   return p;
}
```

Implementação de Pilha com Vetor

- Empilha um elemento na pilha

```
void push(Pilha *p, int e){
     if (p->topo == N - 1){ /* capacidade esgotada */
        printf("A pilha está cheia");
        exit(1);
     }
     /* insere o elemento na próxima posição livre */
     p->elementos[++p->topo] = e;
}
```

Implementação de Pilha com Vetor

- Desempilha um elemento da pilha

```
int pop(Pilha *p)
{
      int e;
      if (empty(p)){
          printf("Pilha vazia.\n");
          exit(1);
      }

      /* retira o elemento do topo */
      e = p->elementos[p->topo--];
      return e;
}
```

Implementação de Pilha com Vetor

```c
/**
 * Verifica se a pilha p está vazia
 */
int empty(Pilha *p)
{
    return (p->t == -1);
}
```

Exemplo de uso

- Na área computacional existem diversas aplicações de pilhas.
- Alguns exemplos são: caminhamento em árvores, chamadas de sub-rotinas por um compilador ou pelo sistema operacional, inversão de uma lista, avaliar expressões, entre outras.
- Uma das aplicações clássicas é a conversão e a avaliação de expressões algébricas. Um exemplo, é o funcionamento das calculadoras da HP, que trabalham com expressões pós-fixadas.