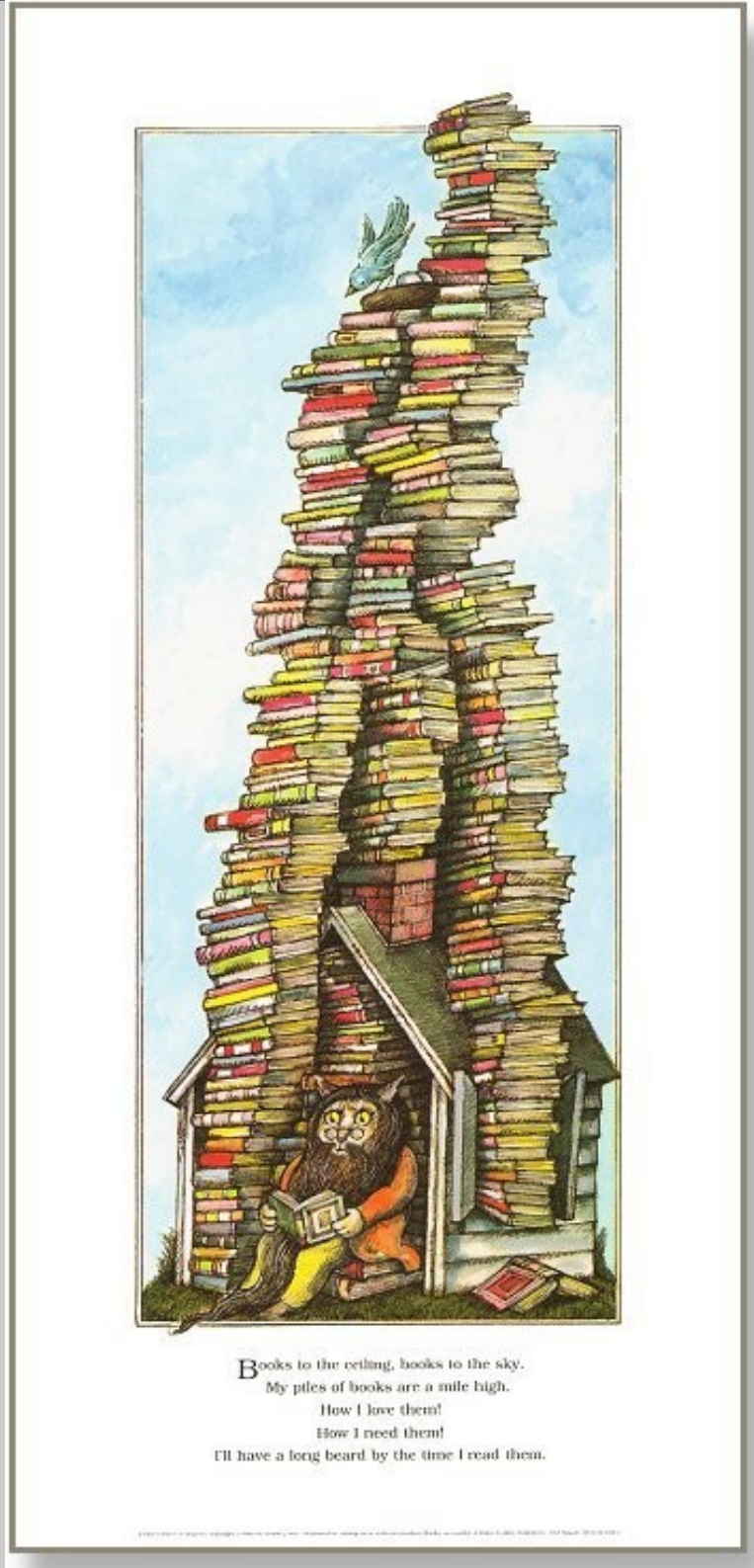
Pilhas e Filas

Estrutura de Dados

Profa. Carla Koike - CIC

O que é uma pilha?

- A posição dos dados depende da ordem de chegada.
- Os dados mais recentes são os primeiros a saírem
- Os primeiros dados são os últimos a serem recuperados.
- First In Last Out (FILO)



O que é uma fila?

- Os dados são armazenados de acordo com a ordem de chegada
- Os primeiros a chegarem são os primeiros a sair
- First In First Out (FIFO)

TAD Pilha

- Estrutura linear de dados que pode ser acessada somente por uma das extremidades para armazenar e recuperar dados.
- Operações possíveis:
  - limpar a pilha
  - verificar se a pilha está vazia
  - coloca elemento no topo da pilha
  - retira o elemento no topo da pilha
  - retorna o elemento no topo, sem removê-lo
  - tamanho da pilha

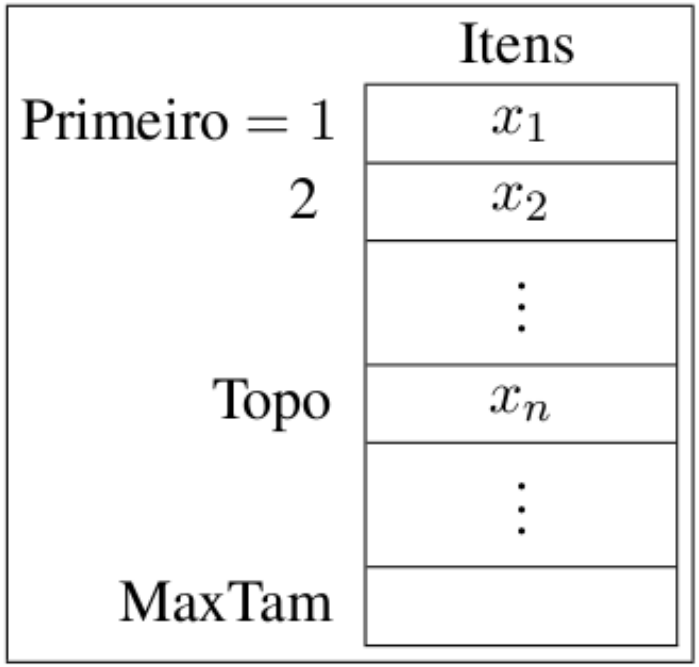
Aplicações de Pilhas

- É ideal para processamento de estruturas aninhadas de profundidade imprevisível.
- Uma pilha contém uma seqüência de obrigações adiadas. A ordem de remoção garante que as estruturas mais internas serão processadas antes das mais externas.

Aplicações de Pilhas, cont.

- Aplicações em estruturas aninhadas:
  - Quando é necessário caminhar em um conjunto de dados e guardar uma lista de coisas a fazer posteriormente.
  - O controle de seqüências de chamadas de subprogramas.
  - A sintaxe de expressões aritméticas.
- As pilhas ocorrem em estruturas de natureza recursiva (como árvores). Elas são utilizadas para implementar a recursividade.

Implementações do TAD Pilha – Usando Vetor



- Um índice chamado Topo é utilizado para controlar a posição do item no topo da pilha.

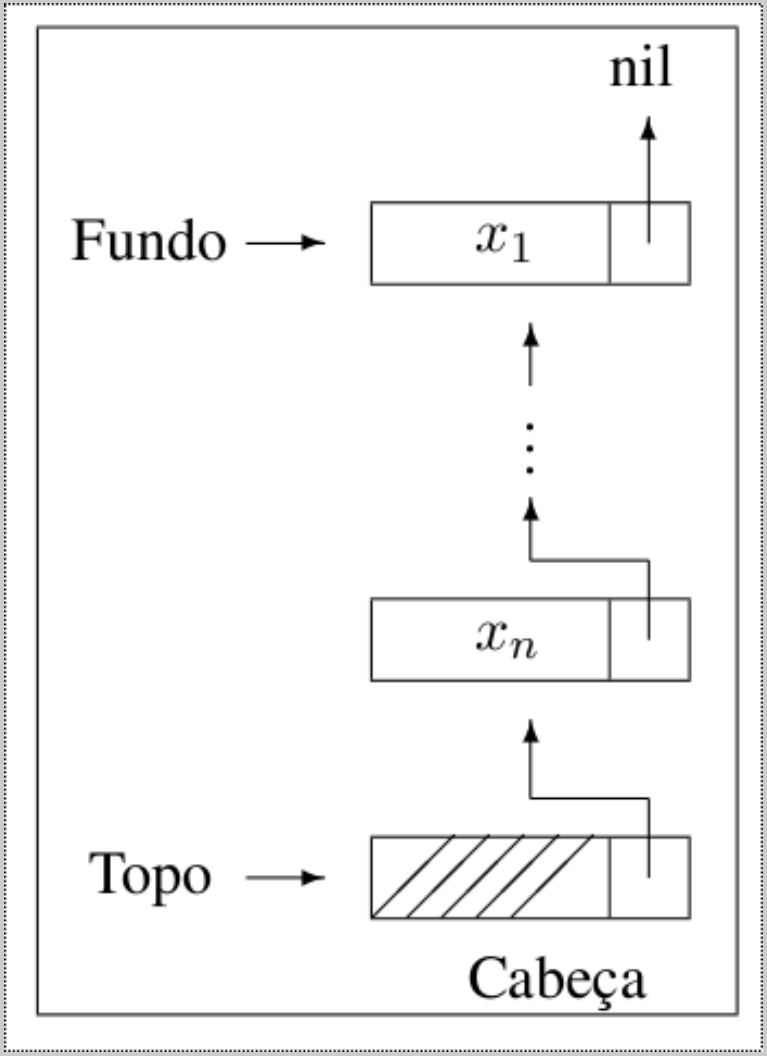
```
typedef int Apontador;
typedef int TipoChave;
typedef struct {
  TipoChave Chave;
  /*outros componentes */
} TipoItem;
typedef struct {
  TipoItem Item[MaxTam];
  Apontador Topo;
} TipoPilha;
```

Implementações do TAD Pilha – Usando Vetor

```
void Empilha(TipoItem x, TipoPilha *Pilha)
{
  if (Pilha->Topo == MaxTam)
  printf(" Erro   pilha esta  cheia\n");
  else { Pilha->Topo++;
         Pilha->Item[Pilha->Topo - 1] = x;
       }
}  /* Empilha */

void Desempilha(TipoPilha *Pilha, TipoItem *Item)
{
  if (Vazia(*Pilha))
  printf(" Erro   pilha esta  vazia\n");
  else { *Item = Pilha->Item[Pilha->Topo - 1];
         Pilha->Topo--;
       }
}  /* Desempilha */
```

Implementações do TAD Pilha – usando Lista Encadeada




```
typedef int TipoChave;

typedef struct {
  TipoChave Chave;
  /* outros componentes*/
} TipoItem;

typedef struct Celula_str
  *Apontador;

typedef struct Celula_str {
  TipoItem Item;
  Apontador Prox;
} Celula;

typedef struct {
  Apontador Fundo, Topo;
  int Tamanho;
} TipoPilha;
```

Implementações do TAD Pilha – usando Lista Encadeada

```
void Empilha(TipoItem x, TipoPilha *Pilha){
  Apontador Aux;
  Aux = (Apontador) malloc(sizeof(Celula));
  Pilha->Topo->Item = x;
  Aux->Prox = Pilha->Topo;
  Pilha->Topo = Aux;
  Pilha->Tamanho++;
}  /* Empilha */

void Desempilha(TipoPilha *Pilha, TipoItem *Item){
  Apontador q;
  if (Vazia(*Pilha))
  { printf(" Erro   lista vazia\n");
    return; }
  q = Pilha->Topo;
  Pilha->Topo = q->Prox;
  *Item = q->Prox->Item;
  free(q);
  Pilha->Tamanho--; }  /* Desempilha */
```

TAD Fila

- Uma fila é uma estrutura na qual ambas as extremidades são usadas: uma para adicionar novos elementos e outra para removê-los.
- Operações possíveis:
  - limpa a fila
  - verifica se a fila está vazia
  - verifica se a fila está cheia
  - coloca elemento no final da fila
  - retira o primeiro elemento da fila
  - tamanho da fila

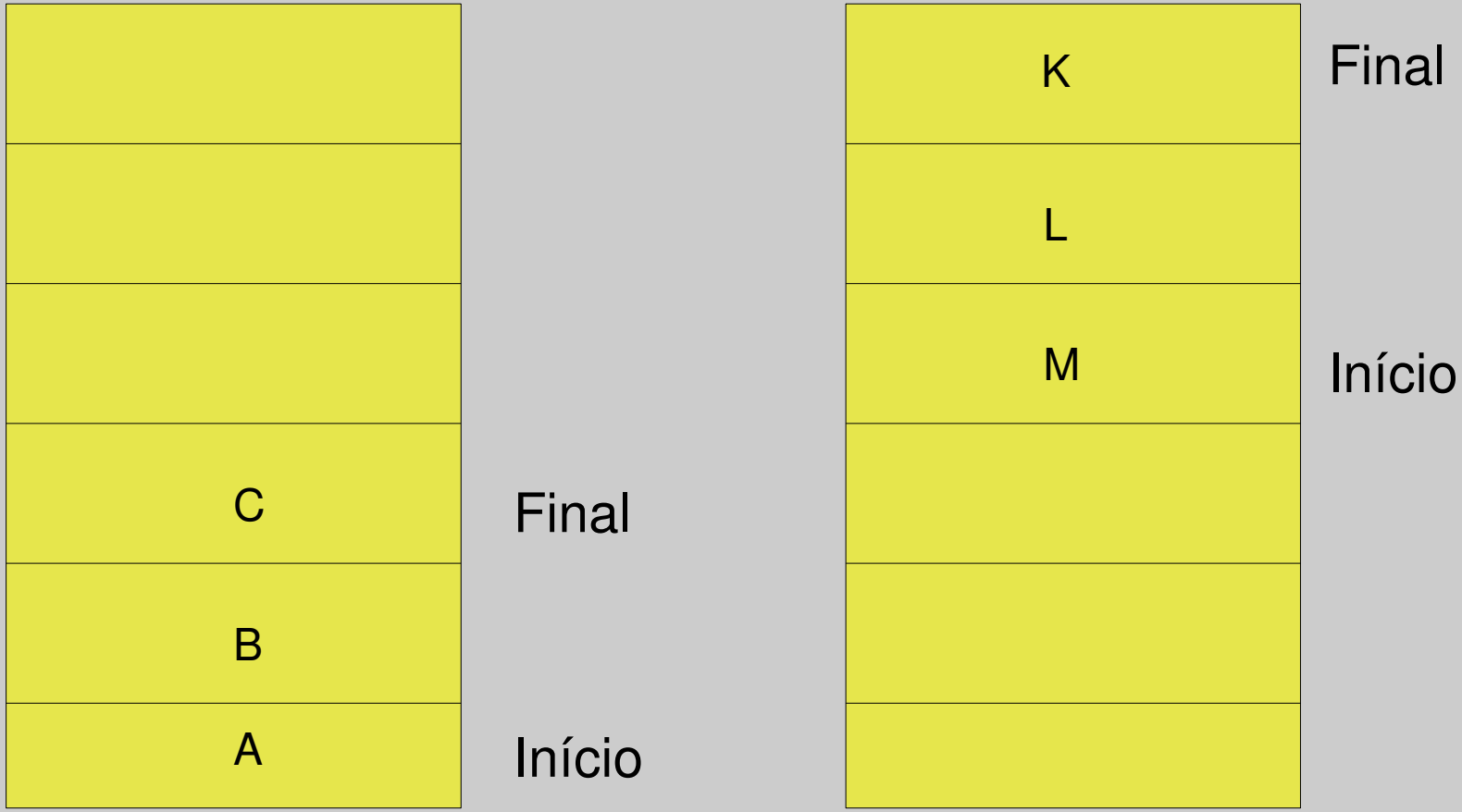
Aplicações de Filas

- São utilizadas quando desejamos processar itens de acordo com a ordem "primeiro-que-chega, primeiro-atendido".
- Sistemas operacionais utilizam filas para regular a ordem na qual tarefas devem receber processamento e recursos devem ser alocados a processos.

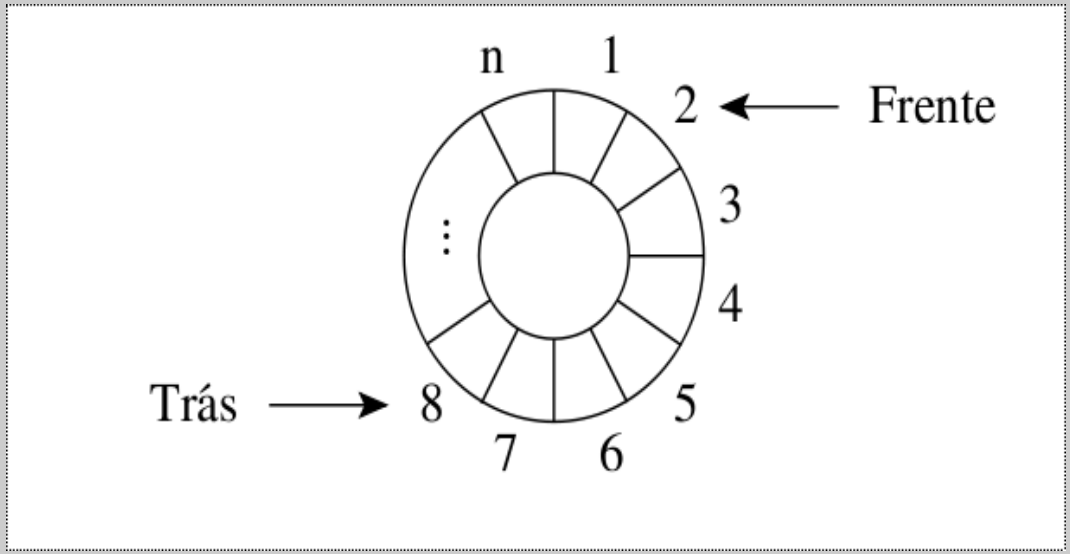
Implementações TAD Fila

Usando Vetor

Vetor com N elementos



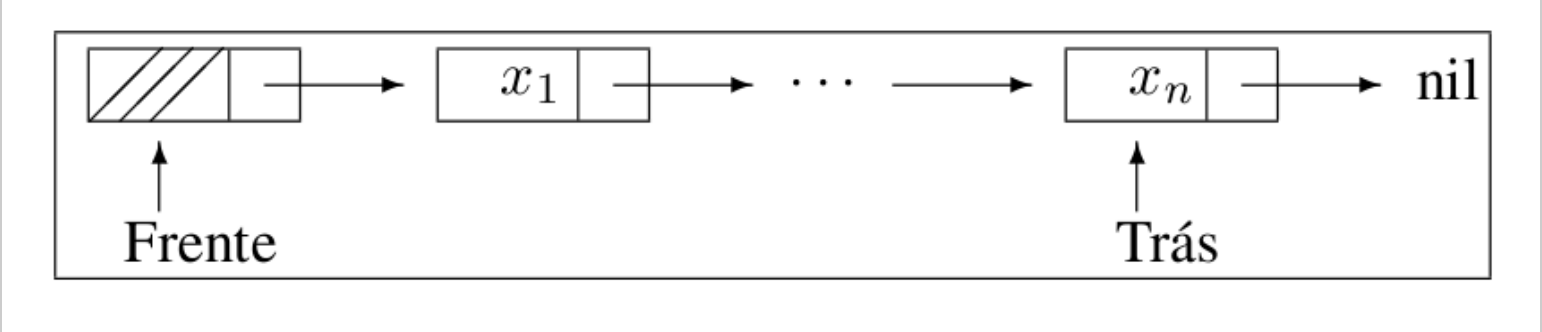
Implementações TAD Fila Usando Vetor Circular



- A fila se encontra em posições contíguas de memória, em alguma posição do círculo, delimitada pelos apontadores Frente e Trás.
- Para enfileirar, basta mover o apontador Trás uma posição no sentido horário.
- Para desenfileirar, basta mover o apontador Frente uma posição no sentido horário.

Programa: Fila_Circular.c

Implementações TAD Fila Usando Lista Encadeada



Programa: FilaLista.c

Expressões na forma Polonesa reversa

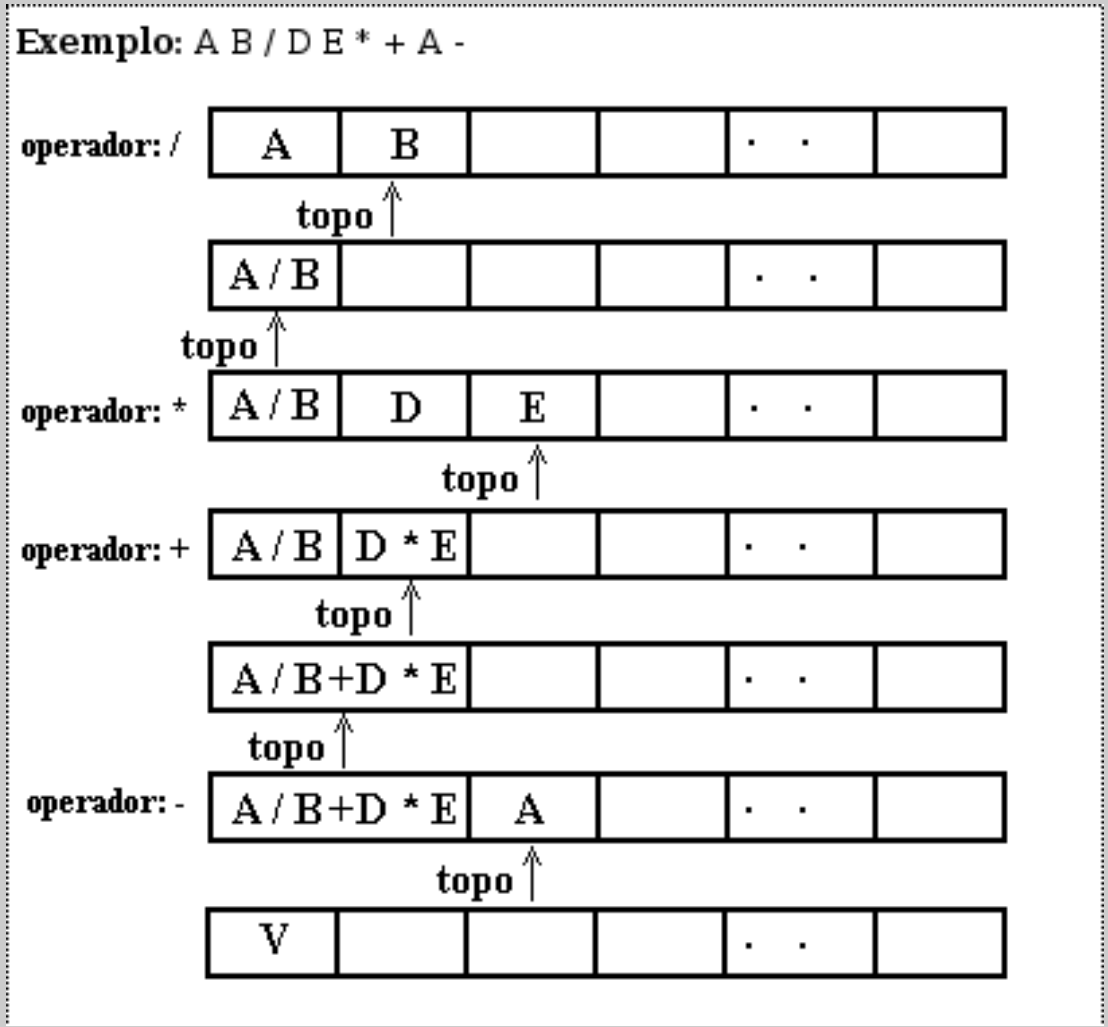
Infixada	Pós-fixada correspondente
A × B / C	A B × C /
A / B × C + D × E – A × C	A B / C × D E × + A C × -
(A / B) × (C + D) × (E - A) × C	A B / C D + × E A – × C ×

Operadores aparecem *entre* os operandos

Operadores aparecem *após* os operandos

Sugestão: faça um programa para converter uma expressão da forma infixada para a forma pós-fixada.

Exemplos: Calculadora em Notação Polonesa Reversa



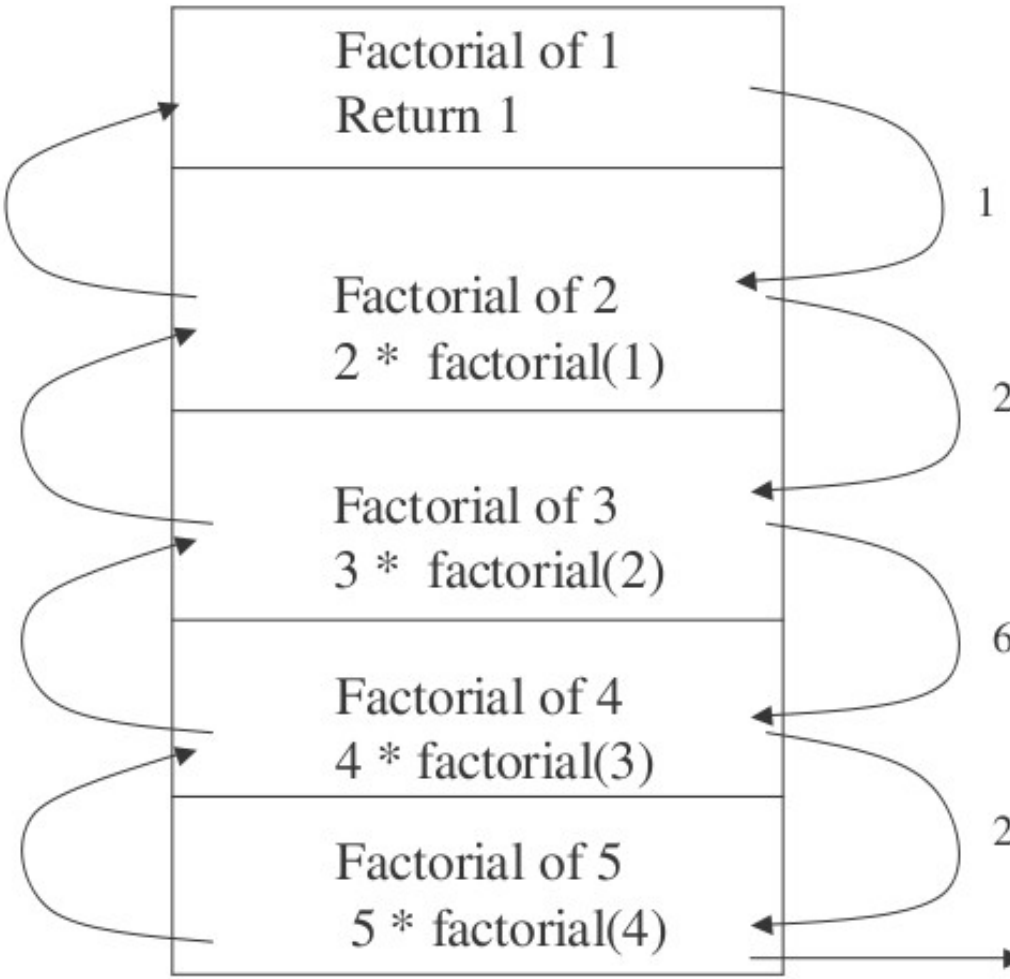
- Um algoritmo para a avaliação de Expressões PosFix:
  - empilha operandos até encontrar um operador
  - retira o número de operandos; calcula e empilha o valor resultante
  - até que chegue ao final da expressão

Programa: polonesa.c

Exemplo: recursividade usando a pilha

- Uso de recursividade para solucionar problemas
  - cálculo fatorial, série de fibonaci, torre de hanoi, ...
- passagem de parâmetros e retorno de valores
  - First in last out: pilha

Fatorial Recursivo




```
long factorial (int n) {
    if (n <= 1)
        return 1;
    else
        return n * factorial(n-1);
}
```