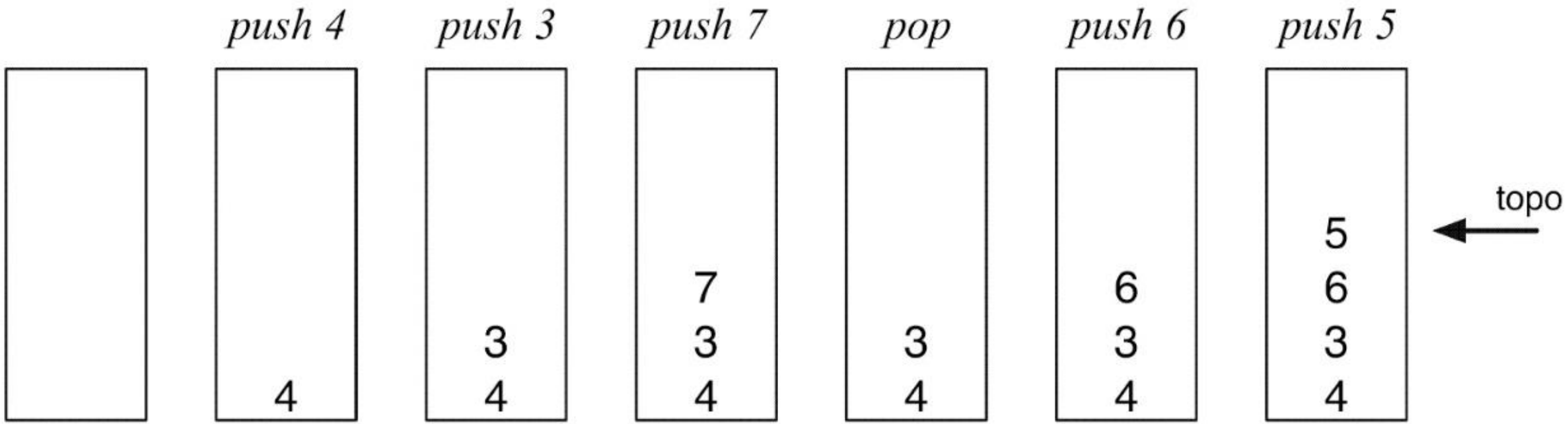
Estrutura de Dados

IDP

Pilhas e Filas

- Estruturas de dados largamente utilizadas em computação;
- Inserção e retirada de elementos seguem regras específicas;
- **Pilha**: LIFO (*last in, first out*), o último elemento a entrar é o primeiro a sair;
- **Fila**: FIFO (*first in, first out*), o primeiro elemento a entrar é o primeiro elemento a sair;
- As regras facilitam a implementação das estruturas, e atendem a demandas de aplicações que precisam armazenar conjunto de valores.

Pilhas



Pilhas

- Uma das estruturas de dados mais simples e a mais utilizada em programação;
- **Ideia fundamental**: todo o acesso a seus elementos é feito através do seu topo;
- Se um elemento é introduzido, passa a ser o elemento do topo, e o único elemento que pode ser removido é o do topo;
- **Operações básicas**: empilhar (*push*), desempilhar (*pop*);
- **Exemplo mais próximo**: pilha de execução da linguagem C. Variáveis locais das funções são dispostas na pilha de execução e uma função só tem acesso às variáveis da função que está no topo (não é possível acessar variáveis locais às outras funções).

Interface do tipo pilha

- **Duas implementações básicas**: usando um vetor e usando uma lista encadeada;
- Independente da estratégia de implementação, podemos definir a **interface do tipo abstrato** que representa a pilha;
- Interface composta pelas operações que estarão disponíveis para manipular a pilha;
- **Operações**: criar pilha vazia, inserir elemento no topo (push), remover elemento do tipo (pop), verificar se a pilha está vazia, liberar a estrutura de pilha.

Interface do tipo pilha

“pilha.h”

```
typedef struct pilha Pilha;

Pilha* pilha_cria (void);
void pilha_push (Pilha* p, float v);
float pilha_pop (Pilha *p);
int pilha_vazia (Pilha *p);
void pilha_libera (Pilha *p);
```

Exemplo de código.

Filas

- A estrutura de fila é uma analogia natural com o conceito de fila que usamos no dia dia;
- **Ideia fundamental**: só podemos inserir um novo elemento no final da fila e só podemos retirar o elemento do início;
- **Exemplo**: fila de impressão. Se uma impressora é compartilhada por várias máquinas, deve-se adotar uma estratégia para determinar que documento será impresso primeiro.
- **Duas implementações**: usando um vetor e usando uma lista encadeada.

Interface do tipo fila

- **Operações do tipo abstrato**: criar uma fila vazia, inserir um elemento no fim, retirar o elemento do início, verificar se a fila está vazia, liberar a fila;

- "fila.h":

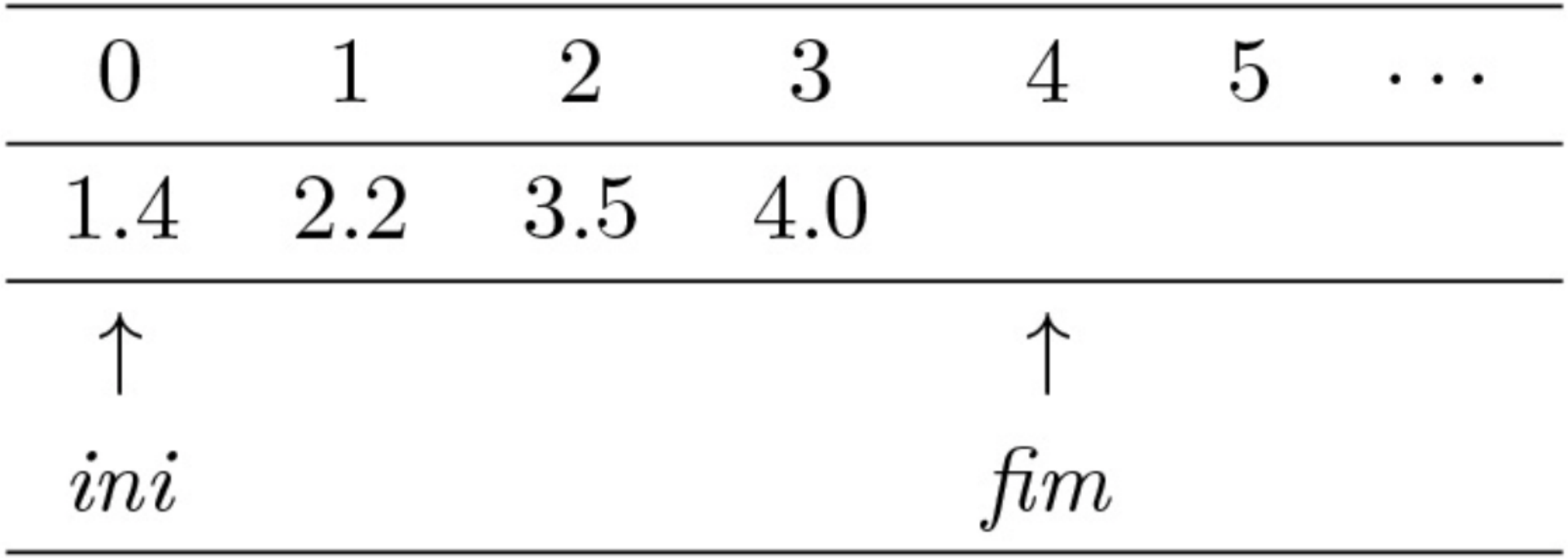
```
typedef struct fila Fila;

Fila* fila_cria (void);
void fila_insere (Fila* f, float v);
float fila_retira (Fila* f);
int fila_vazia (Fila* f);
Void fila_libera (Fila* f);
```

Fila com vetores

- **Estratégia simplista com vetores**: inserção de novos elementos no final do vetor e a retirada do início, deslocando os elementos do vetor para preencher o espaço vazio no início do vetor;
- **Custo computacional é muito elevado:** é melhor evitar a necessidade de deslocar os elementos no vetor;
- **Ideia:** inserção e remoção em extremidades opostas fará com que a fila “ande” no vetor.

Fila com vetores



Fila com vetores

0	1	2	3	4	5	...
1.4	2.2	3.5	4.0			
		↑		↑		
		ini		*fim*		

Fila com vetores

- É fácil observar que, em um dado instante, a parte ocupada do vetor pode chegar à última posição;
- Os elementos antes de *ini* não fazem mais parte da fila, e suas posições devem ser reaproveitadas;
- **Incrementar as posições do vetor de forma "circular"**;
- Se o último elemento da fila ocupa a última posição do vetor, e existirem posições livres no início, inserimos os novos elementos a partir do início do vetor;
- **Consequência: em determinado momento, poderemos ter valores no fim e o início do vetor.**

Fila com vetores

0	1	2	$\cdots$	$n-2$	$n-1$
21.2	24.3			20.0	22.8
		$\uparrow$		$\uparrow$	
		fim		ini	

Fila com vetores

- Este incremento circular, considerando *dim* a dimensão do vetor, pode ser dada de duas maneiras:

$$i = (i == (dim - 1)) \ ? \ 0 : i + 1;$$

$$i = (i + 1) \ \% \ dim;$$

- **Tipo fila**: vetor dinâmico *vet*, dimensão atual do vetor *dim*, número de elementos armazenados na fila *n* e um índice *ini* para o início da fila;

$$fim = (ini + n) \ \% \ dim$$

- **Exemplo de código com vetores;**

Fila com lista