Algoritmos
+
Estruturas de Dados
=
Programas

TIPO ABSTRATO DE DADOS

anaeliza.moura@unicap.br

1

Tipos Abstratos de Dados

- **Definição**
  - Um tipo abstrato de dados (TAD) é um tipo de dados definido através de seu **comportamento**.
- **Exemplos clássicos**
  - Stack (pilha)
  - Queue (fila)
  - Binary Search Tree (árvore binária de busca)

anaeliza.moura@unicap.br

2

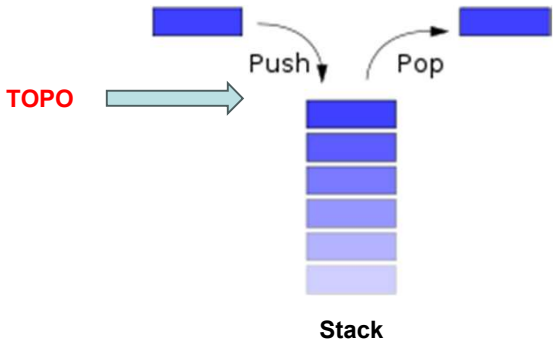
Stacks (Pilhas)

- **Definições**
  - Uma pilha é um TAD que obedece a disciplina de acesso **LIFO** ("*Last In First Out*"), ou seja, o último elemento a entrar é, obrigatoriamente, o primeiro a sair.

anaeliza.moura@unicap.br

3

Stacks (Pilhas)



Stack

TOPO da pilha:
Posição na qual serão feitas as inserções e retiradas de elementos da pilha.

anaeliza.moura@unicap.br

4

Stacks (Pilhas)

- **Comportamento**

O conjunto de operações que definem o comportamento de uma pilha são:

- **PUSH** : Empilhar, ou seja, colocar um elemento no topo da pilha. **Só pode ser executada em pilhas não cheias.**
- **POP**: Desempilhar, ou seja, retirar um elemento do topo da pilha. **Só pode ser executada em pilhas não vazias.**
- **TOP**: Informar que elemento encontra-se no topo da pilha. **Só pode ser executada em pilhas não vazias.**
- **ISEMPTY**: Verificar se a pilha está vazia;
- **ISFULL**: Verificar se a pilha está cheia.

anaeliza.moura@unicap.br

5

Stacks (Pilhas)

- **Aplicações do TAD Stack**
  - Pilha de execução de um programa;
  - Avaliação de expressões aritméticas;
  - Compiladores: geração de código em linguagem de máquina;
  - Implementação de alguns métodos não recursivos de árvores.

anaeliza.moura@unicap.br

6

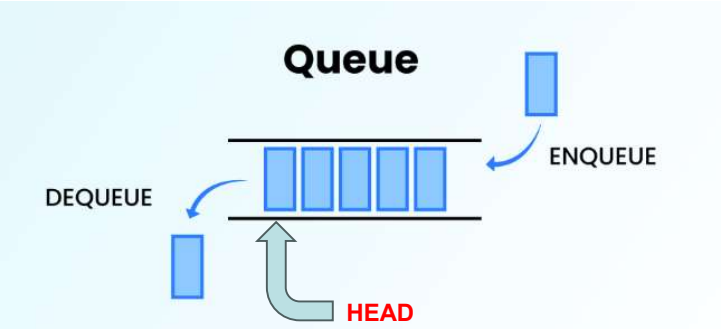
Queues (Filas)

- **Definições**
  - Uma fila é um TAD que obedece a disciplina de acesso **FIFO** ("*First In First Out*"), ou seja, o primeiro elemento a entrar é, obrigatoriamente, o primeiro a sair.

anaeliza.moura@unicap.br

7

Queues (Filas)



HEAD (cabeça da fila):
Elemento que se encontra no início da fila.

TAIL (calda da fila):
Elementos que compõem a fila menos o HEAD.

anaeliza.moura@unicap.br

8

Queues (Filas)

- **Comportamento**

O conjunto de operações que definem o comportamento de uma fila são:

- **ENQUEUE** : Enfileirar, ou seja, colocar um elemento no final da fila. **Só pode ser executada em filas não cheias.**
- **DEQUEUE**: Desenfileirar, ou seja, retirar o elemento do início da fila (retirar o HEAD). **Só pode ser executada em filas não vazias.**
- **HEAD**: Informar que elemento encontra-se no início da fila. **Só pode ser executada em filas não vazias.**
- **ISEMPTY**: Verificar se a fila está vazia.
- **ISFULL** Verificar se a fila está cheia.

anaeliza.moura@unicap.br

9

Queues (Filas)

- **Aplicações do TAD Queue**
  - Fila de espera de aplicativos para serem executados pelo processador;
  - Fila de espera de pacotes de dados no roteador para serem enviados;
  - Filas de solicitações de acesso a um determinado arquivo em um servidor de arquivos.

anaeliza.moura@unicap.br

10

Estruturas de Dados

- **Implementação**
  - Um tipo abstrato de dados (TAD) é implementado utilizando-se uma estrutura de dados.
  - Uma estrutura de dados é um tipo de dados compostos (estruturado), por exemplo, um vetor, um arquivo ou uma lista encadeada.

anaeliza.moura@unicap.br

11

Stacks (Pilhas)

- **Exemplo de Implementação: Sequencial**
  - Pilha de inteiros implementada utilizando uma estrutura de dados do tipo vetor.
  - Exemplo:

```
#define TAM 10
typedef struct stack {
      int dados [TAM];
      int topo;
} Stack;
Stack minhaPilha;
```

anaeliza.moura@unicap.br

12

Stacks (Pilhas)

- **Exemplo de Implementação: Encadeada**
  - Pilha de inteiros implementada utilizando o tipo de dados ponteiro e memória de alocação dinâmica.
  - Exemplo:

```
typedef struct stackNode {
      int info;
      struct stackNode * prox;
} StackNode;
typedef struct stack {
      StackNode* topo;
} Stack;
Stack minhaPilha;
```

anaeliza.moura@unicap.br

13

Queues (Filas)

- **Exemplo de Implementação: Sequencial**
  - Fila de inteiros implementada utilizando uma estrutura de dados do tipo vetor.
  - Exemplo:

```
#define TAM 10
typedef struct queue {
     int dados [TAM];
     int inicio, fim;
} Queue;
Queue minhaFila;
```

anaeliza.moura@unicap.br

14

Queues (Filas)

- **Exemplo de Implementação: Encadeada**
  - Fila implementada utilizando o tipo de dados ponteiro e memória de alocação dinâmica.
  - Exemplo:

```
typedef struct queueNode {
        int info;
        struct queueNode * prox;
} QueueNode;
typedef struct queue {
        QueueNode * inicio, * fim;
} Queue;
Queue minhaFila;
```

anaeliza.moura@unicap.br

15

Tipos Abstratos de Dados

- **Utilização**
  - A utilização do TAD independe da forma como o mesmo foi implementado;
  - Modificações na implementação do TAD não implicam em modificações nos programas que os utilizam.

anaeliza.moura@unicap.br

16

Stacks (Pilhas)

- **Exemplo de criação e uso de uma pilha de inteiros:**

```
Stack minhaPilha;
inicializar (&minhaPilha);
if (isFull(minhaPilha) == 0) {
    push (&minhaPilha,10);
}
if (isEmpty(minhaPilha) == 0) {
    int n = pop(&minhaPilha);
}
if (isEmpty(minhaPilha) == 0) {
    printf ("%d \n", top(minhaPilha));
}
```

anaeliza.moura@unicap.br

Queues (Filas)

- **Exemplo de criação e uso de uma fila:**

```
Queue minhaFila;
inicializar (&minhaFila);
if (isFull(minhaFila) == 0) {
    enQueue (&minhaFila,10);
}
if (isEmpty(minhaFila) == 0) {
    int n = deQueue(&minhaFila);
}
if (isEmpty(minhaFila) == 0) {
    printf("%d \n", head(minhaFila));
}
```

anaeliza.moura@unicap.br