

# PILHAS E FILAS

Gustavo Carvalho  
(ghpc@cin.ufpe.br)

Universidade Federal de Pernambuco  
Centro de Informática, 50740-560, Brazil



# Agenda

1 Pilhas

2 Filas

3 Bibliografia



# ADT Pilha

Baseada em lista: inserção + remoção somente em uma ponta

- **LIFO**: last-in, first-out

Operações:

- void clear(Stack s);
- void push(Stack s, E it);
- E pop(Stack s);
- E topValue(Stack s);
- int length(Stack s);

Implementações: array e **lista ligada**



# Implementação baseada em lista ligada

Estrutura de dados (Stack):

---

---

```

1  Link top ;           // ponteiro para o primeiro elemento
2  int size ;           // quantidade de elementos

```

---

**Algoritmo:** Stack create\_stack()

---

```

1  s.top  $\leftarrow$  NULL;
2  s.size  $\leftarrow$  0;
3  return s;

```

---

**Algoritmo:** void clear(Stack s)

---

```

1  s.top  $\leftarrow$  NULL;
2  s.size  $\leftarrow$  0;

```

---



# Implementação baseada em lista ligada (cont.)

---

**Algoritmo:** void push(Stack s, E it)

---

- 1  $s.top \leftarrow create\_link(it, s.top);$
  - 2  $s.size++;$
- 

---

**Algoritmo:** E pop(Stack s)

---

- 1 **if**  $s.top = NULL$  **then** error ;
  - 2  $it \leftarrow s.top.element;$
  - 3  $s.top \leftarrow s.top.next;$
  - 4  $s.size--;$
  - 5 **return**  $it;$
-

# Implementação baseada em lista ligada (cont.)

---

**Algoritmo:** E topValue(Stack s)

---

```
1 if s.top = NULL then error ;  
2 return s.top.element;
```

---

---

**Algoritmo:** int length(Stack s)

---

```
1 return s.size;
```

---

# Agenda

1 Pilhas

2 Filas

3 Bibliografia



# ADT Fila

Baseada em lista: inserção no final | remoção no início

- **FIFO**: first-in, first-out

Operações:

- void clear(Queue q);
- void enqueue(Queue q, E it);
- E dequeue(Queue q);
- E frontValue(Queue q);
- int length(Queue q);

Implementações: array e **lista ligada**

Baseada em array: implementação eficiente não é “direta”

- Arrays circulares
- Fila vazia ou cheia?





# Implementação baseada em lista ligada

Estrutura de dados (Queue):

---



---

```

1  Link front ;           // ponteiro para o início (nó header)
2  Link rear ;           // ponteiro para o final
3  int size ;            // quantidade de elementos
  
```

---

**Algoritmo:** Queue create\_queue()

---

```

1  q.front  $\leftarrow$  q.rear  $\leftarrow$  create_link(NULL);
2  q.size  $\leftarrow$  0;
3  return q;
  
```

---

**Algoritmo:** void clear(Queue q)

---

```

1  q.front  $\leftarrow$  q.rear  $\leftarrow$  create_link(NULL);
2  q.size  $\leftarrow$  0;
  
```

---

# Implementação baseada em lista ligada (cont.)

---

**Algoritmo:** void enqueue(Queue q, E it)

---

```
1 q.rear.next  $\leftarrow$  create_link(it, NULL);  
2 q.rear  $\leftarrow$  q.rear.next;  
3 q.size++;
```

---

---

**Algoritmo:** E dequeue(Queue q)

---

```
1 if q.size = 0 then error ;  
2 it  $\leftarrow$  q.front.next.element;  
3 q.front.next  $\leftarrow$  q.front.next.next;  
4 if q.front.next = NULL then q.rear  $\leftarrow$  q.front ;  
5 q.size--;  
6 return it;
```

---

# Implementação baseada em lista ligada (cont.)

---

**Algoritmo:** E frontValue(Queue q)

---

```
1 if  $q.size = 0$  then error ;  
2 return  $q.front.next.element$ ;
```

---

---

**Algoritmo:** int length(Queue q)

---

```
1 return  $q.size$ ;
```

---

# Agenda

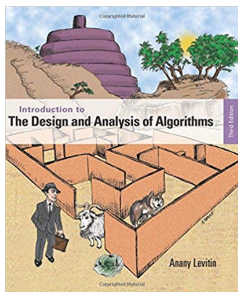
1 Pilhas

2 Filas

3 Bibliografia



# Bibliografia + leitura recomendada



**Capítulo 1 (pp. 25–28).**

**Anany Levitin.**

*Introduction to the Design and Analysis of Algorithms.*

3a edição. Pearson. 2011.



**Capítulo 4 (pp. 117–131)**

**Clifford Shaffer.**

*Data Structures and Algorithm Analysis.*

Dover, 2013.



# PILHAS E FILAS

Gustavo Carvalho  
(ghpc@cin.ufpe.br)

Universidade Federal de Pernambuco  
Centro de Informática, 50740-560, Brazil

