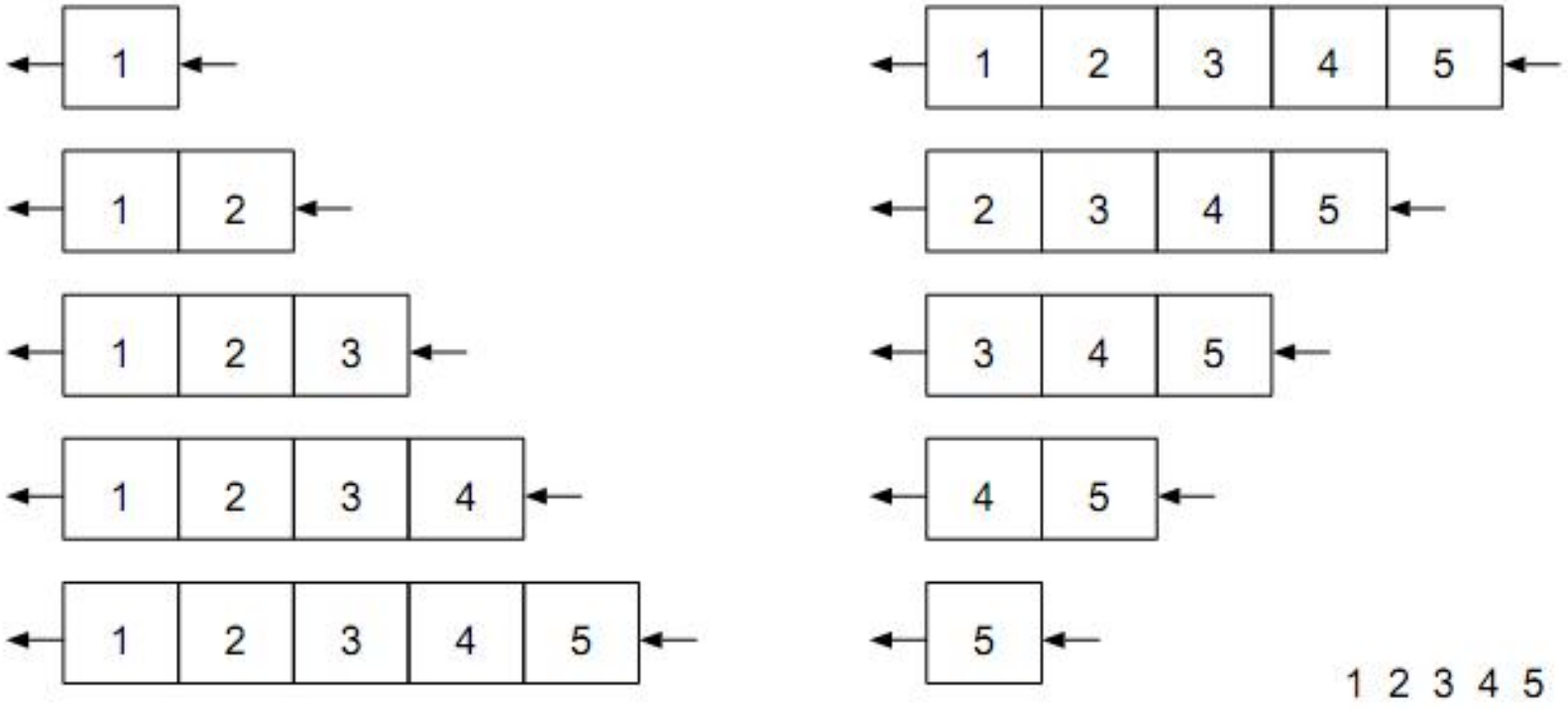
PROGRAMAÇÃO I

Curso Técnico Subsequente em Informática

Lucas Sampaio Leite

Fila (Queue)

- Conceito: Uma fila é uma estrutura FIFO (First In, First Out), ou seja, o primeiro que entra é o primeiro que sai.



Fila (Queue)

INSTITUTO FEDERAL
Baiano

- Conceito: Uma fila é uma estrutura FIFO (First In, First Out), ou seja, o primeiro que entra é o primeiro que sai.
- Operações principais:
  - enqueue(x) → insere um elemento no final da fila.
  - dequeue() → remove o elemento do início da fila.
  - front() → consulta o primeiro elemento da fila.
  - is_empty() → verifica se a fila está vazia.
- Implementação em Python:
  - Usando lista (menos eficiente).
  - Usando collections.deque (recomendado).

Fila (Queue)

INSTITUTO FEDERAL
Baiano

- Usando lista (menos eficiente):

```
fila = []

fila.append('A')
fila.append('B')
fila.append('C')
print("Fila:", fila)

primeiro = fila.pop(0)
print("Removido:", primeiro)
print("Fila após remoção:", fila)
```

Fila (Queue)

INSTITUTO FEDERAL
Baiano

- Usando lista (menos eficiente):

```
fila = []

fila.append('A')
fila.append('B')
fila.append('C')
print("Fila:", fila)

primeiro = fila.pop(0)
print("Removido:", primeiro)
print("Fila após remoção:", fila)
```

```
Fila: ['A', 'B', 'C']
Removido: A
Fila após remoção: ['B', 'C']
```

Fila (Queue)

- Usando lista (menos eficiente):

```
fila = []

def enqueue(fila, elemento):
    fila.append(elemento)

def dequeue(fila):
    if len(fila) == 0:
        print("A fila está vazia!")
        return None
    return fila.pop(0)

enqueue(fila, 'A')
enqueue(fila, 'B')
enqueue(fila, 'C')
print("Fila:", fila)

removido = dequeue(fila)
print("Removido:", removido)
print("Fila após remoção:", fila)
```

INSTITUTO FEDERAL
Baiano

Fila (Queue)

- Usando lista (menos eficiente):

```
fila = []

def enqueue(fila, elemento):
    fila.append(elemento)

def dequeue(fila):
    if len(fila) == 0:
        print("A fila está vazia!")
        return None
    return fila.pop(0)

enqueue(fila, 'A')
enqueue(fila, 'B')
enqueue(fila, 'C')
print("Fila:", fila)

removido = dequeue(fila)
print("Removido:", removido)
print("Fila após remoção:", fila)
```

```
Fila: ['A', 'B', 'C']
Removido: A
Fila após remoção: ['B', 'C']
```

INSTITUTO FEDERAL
Baiano

Exercício rápido

INSTITUTO FEDERAL
Baiano

- Implemente as funções front() e is_empty().

Fila (Queue)

INSTITUTO FEDERAL
Baiano

- Usando collections.deque (recomendado):

```
from collections import deque

fila = deque()

fila.append('A')
fila.append('B')
fila.append('C')
print("Fila:", fila)

primeiro = fila.popleft()
print("Removido:", primeiro)
print("Fila após remoção:", fila)
```

Fila (Queue)

INSTITUTO FEDERAL
Baiano

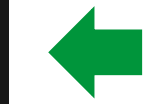
- Usando collections.deque (recomendado):

```
from collections import deque

fila = deque()

fila.append('A')
fila.append('B')
fila.append('C')
print("Fila:", fila)

primeiro = fila.popleft()
print("Removido:", primeiro)
print("Fila após remoção:", fila)
```

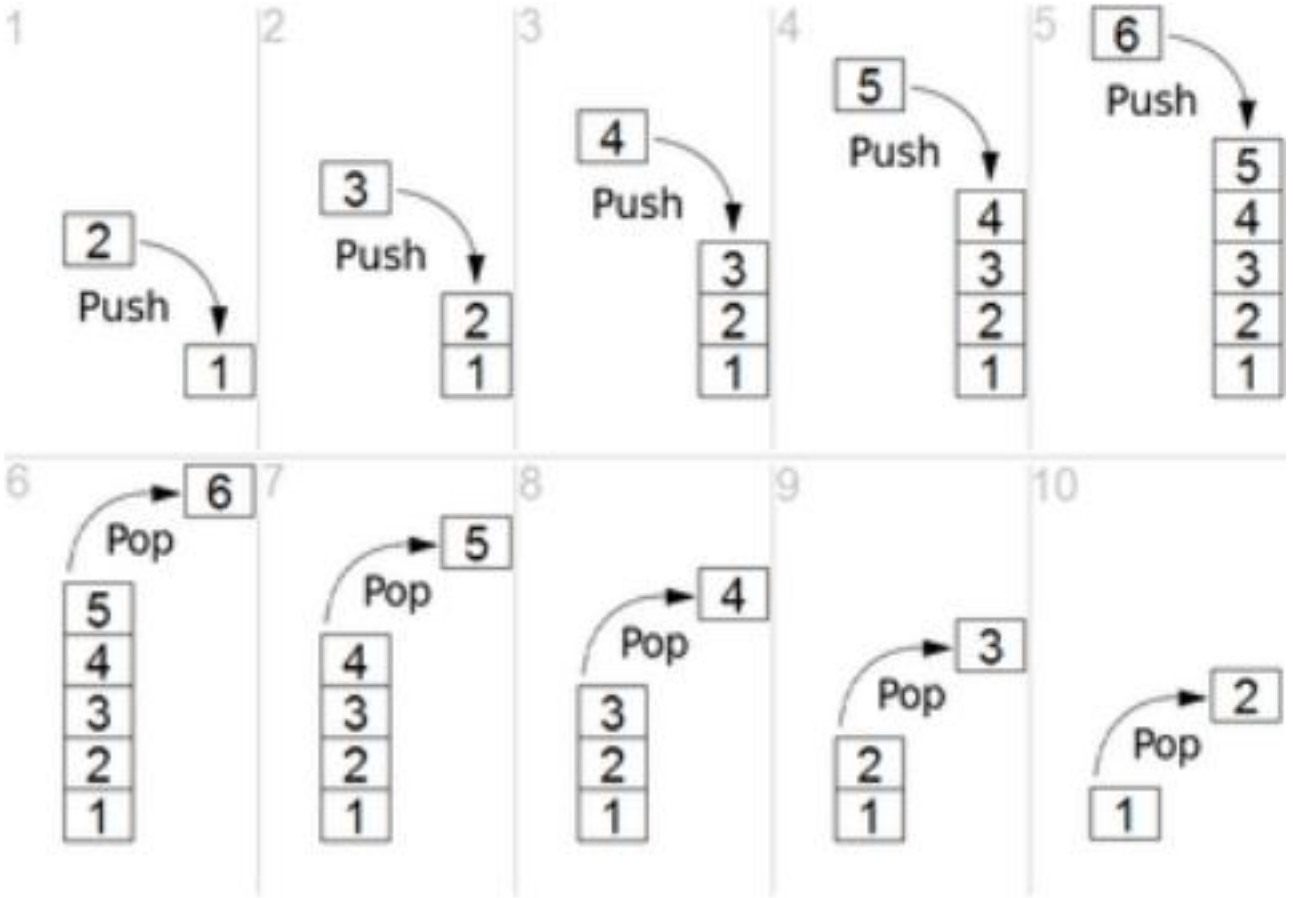


```
Fila: deque(['A', 'B', 'C'])
Removido: A
Fila após remoção: deque(['B', 'C'])
```

Pilha (Stack)

INSTITUTO FEDERAL
Baiano

- Conceito: Uma pilha é uma estrutura LIFO (Last In, First Out), ou seja, o último elemento que entra é o primeiro que sai — como uma pilha de pratos.



Pilha (Stack)

INSTITUTO FEDERAL
Baiano

- Conceito: Uma pilha é uma estrutura LIFO (Last In, First Out), ou seja, o último elemento que entra é o primeiro que sai.
- Operações principais:
  - push(x) → empilha (adiciona) um elemento no topo.
  - pop() → desempilha (remove) o elemento do topo.
  - peek() → consulta o topo da pilha (sem remover).
  - is_empty() → verifica se a pilha está vazia.
- Implementação em Python:
  - Python não tem uma classe Stack nativa, mas podemos usar uma lista (list) ou a classe ollections.deque.

Pilha (Stack)

```python
pilha = []

pilha.append('A')
pilha.append('B')
pilha.append('C')
print("Pilha:", pilha)

topo = pilha.pop()
print("Removido:", topo)
print("Pilha após remoção:", pilha)

print("Topo da pilha:", pilha[-1])

print("Está vazia?", len(pilha) == 0)
```

INSTITUTO FEDERAL
Baiano

Pilha (Stack)

INSTITUTO FEDERAL
Baiano

```python
pilha = []

pilha.append('A')
pilha.append('B')
pilha.append('C')
print("Pilha:", pilha)

topo = pilha.pop()
print("Removido:", topo)
print("Pilha após remoção:", pilha)

print("Topo da pilha:", pilha[-1])

print("Está vazia?", len(pilha) == 0)
```

```
Pilha: ['A', 'B', 'C']
Removido: C
Pilha após remoção: ['A', 'B']
Topo da pilha: B
Está vazia? False
```

Pilha (Stack)

```python
pilha = []

def push(pilha, elemento):
    pilha.append(elemento)

def pop(pilha):
    if not is_empty(pilha):
        return pilha.pop()
    else:
        print("A pilha está vazia!")
        return None

def peek(pilha):
    if not is_empty(pilha):
        return pilha[-1]
    else:
        print("A pilha está vazia!")
        return None

def is_empty(pilha):
    return len(pilha) == 0
```

INSTITUTO FEDERAL
Baiano

Pilha (Stack)

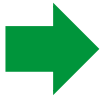
INSTITUTO FEDERAL
Baiano

```
push(pilha, 'A')
push(pilha, 'B')
push(pilha, 'C')
print("Pilha:", pilha)

removido = pop(pilha)
print("Removido:", removido)
print("Pilha após remoção:", pilha)

topo = peek(pilha)
print("Topo da pilha:", topo)

print("Está vazia?", is_empty(pilha))
```



```
Pilha: ['A', 'B', 'C']
Removido: C
Pilha após remoção: ['A', 'B']
Topo da pilha: B
Está vazia? False
```

Pilha (Stack)

```python
from collections import deque

pilha = deque()

def push(pilha, elemento):
    pilha.append(elemento)

def pop(pilha):
    if not is_empty(pilha):
        return pilha.pop()
    else:
        print("A pilha está vazia!")
        return None

def peek(pilha):
    if not is_empty(pilha):
        return pilha[-1]
    else:
        print("A pilha está vazia!")
        return None

def is_empty(pilha):
    return len(pilha) == 0
```

INSTITUTO FEDERAL
Baiano

Pilha (Stack)

```
push(pilha, 'A')
push(pilha, 'B')
push(pilha, 'C')
print("Pilha:", pilha)

removido = pop(pilha)
print("Removido:", removido)
print("Pilha após remoção:", pilha)

topo = peek(pilha)
print("Topo da pilha:", topo)

print("Está vazia?", is_empty(pilha))
```

```
from collections import deque

pilha = deque()

def push(pilha, elemento):
    pilha.append(elemento)

def pop(pilha):
    if not is_empty(pilha):
        return pilha.pop()
    else:
        print("A pilha está vazia!")
        return None

def peek(pilha):
    if not is_empty(pilha):
        return pilha[-1]
    else:
        print("A pilha está vazia!")
        return None

def is_empty(pilha):
    return len(pilha) == 0
```

INSTITUTO FEDERAL
Baiano

Pilha (Stack)

```
push(pilha, 'A')
push(pilha, 'B')
push(pilha, 'C')
print("Pilha:", pilha)

removido = pop(pilha)
print("Removido:", removido)
print("Pilha após remoção:", pilha)

topo = peek(pilha)
print("Topo da pilha:", topo)

print("Está vazia?", is_empty(pilha))
```

```
Pilha: deque(['A', 'B', 'C'])
Removido: C
Pilha após remoção: deque(['A', 'B'])
Topo da pilha: B
Está vazia? False
```

```
from collections import deque

pilha = deque()

def push(pilha, elemento):
    pilha.append(elemento)

def pop(pilha):
    if not is_empty(pilha):
        return pilha.pop()
    else:
        print("A pilha está vazia!")
        return None

def peek(pilha):
    if not is_empty(pilha):
        return pilha[-1]
    else:
        print("A pilha está vazia!")
        return None

def is_empty(pilha):
    return len(pilha) == 0
```

INSTITUTO FEDERAL
Baiano

Filas x Pilhas

INSTITUTO FEDERAL
Baiano

Característica	Lista (list)	Fila(queue)	Pilha(stack)
Tipo de estrutura	Estrutura genérica de dados	Estrutura de dados FIFO (First In, First Out)	Estrutura de dados LIFO (Last In, First Out)
Inserção (entrada)	Qualquer posição	No final	No topo (final)
Remoção (saída)	Qualquer posição	No início	No topo (final)
Principais métodos	append(), insert(), pop(), remove()	enqueue(), dequeue(), front(), is_empty()	push(), pop(), peek(), is_empty()
Exemplo de uso prático	Armazenar dados diversos, listas de objetos	Fila de impressão, fila de atendimento	Pilha de chamadas de função, desfazer/refazer
Analogia do mundo real	Lista de compras (ordem livre)	Fila de banco (quem chega primeiro é atendido primeiro)	Pilha de pratos (quem é colocado por último sai primeiro)

Exercícios

INSTITUTO FEDERAL
Baiano

1. Implemente uma pilha utilizando uma lista chamada pilha. Implemente as funções:

   ```
   def push(x):  # adiciona um elemento no topo
   def pop():    # remove o topo e retorna o valor removido
   def peek():   # retorna o topo sem remover
   def isEmpty(): # verifica se está vazia
   ```

Teste as funcionalidades implementadas.

Exercícios

INSTITUTO FEDERAL
Baiano

2. Crie uma fila com o deque e implemente as funções:

```
def enqueue(x):    # adiciona no final da fila
def dequeue():     # remove o primeiro elemento
def front():       # consulta o primeiro elemento sem remover
def isEmpty():     # verifica se está vazia
```

Teste as funcionalidades implementadas.

Exercícios

INSTITUTO FEDERAL
Baiano

3. Implemente uma fila de atendimento bancário:

   Cada cliente é identificado por um número;

   Use enqueue() e dequeue() para gerenciar a fila;

   Mostre a cada iteração qual cliente está sendo atendido e quantos ainda aguardam.

   Dica: use time.sleep(1) para simular o tempo de atendimento.

Exercícios

INSTITUTO FEDERAL
Baiano

4. Escreva uma função reverter(texto) que use uma pilha para inverter a string passada como parâmetro.

   Exemplo:

   Entrada: "python"

   Saída: "nohtyp"

Dúvidas

INSTITUTO FEDERAL
Baiano

PROGRAMAÇÃO I

Curso Técnico Subsequente em Informática

Lucas Sampaio Leite