

Fatec Rubens Lara **Estrutura de Dados**

Exercícios de revisão (1º. Bimestre letivo)

Prof. Chiara

1. O que podemos entender por um tipo abstrato de dados? E uma estrutura de dados?

- TAD é uma especificação de um conjunto de dados e operações que podem ser executadas sobre esses dados. Em linguagens orientadas a objeto a implementação é feita usando classes e a especificação usando interfaces.
- As estruturas de dados utiliza dados de forma conjunta, levando em consideração a eficiência para buscas, volume de dados e a complexidade, pode realizar operações como inserir, excluir, buscar, localizar e ordenar elementos.

2. O que são listas lineares?

- Lista linear é uma estrutura de dados na qual elementos de um mesmo tipo de dado estão organizados de maneira sequencial.

3. O que são listas lineares restritas? Cite 2 tipos e respectivas definições.

- São listas onde existe alguma restrição na forma com que são atualizadas ou acessadas. A limitação imposta à lista determina uma ordenação dos itens diferente da ordem natural dos valores dos dados.

PILHA é uma lista em que as operações de inserção e remoção são feitas na mesma extremidade da lista, conhecido como topo da pilha.

FILA é uma lista na qual as inserções são feitas em uma extremidade chamada "cauda" ou "fundo", e as remoções são feitas na outra extremidade, chamada "cabeça" ou "frente".

4. Considere os protótipos, definidos abaixo, para uma estrutura de dados do tipo fila, implementada em alocação sequencial:

```
def insere(self, x): //insere um novo elemento na fila
def remove(self): //remove um elemento da fila, retornando este elemento removido
def primeiro(self): //retorna o valor do primeiro elemento da fila.
```

Mostre a situação de uma fila, inicialmente vazia após cada uma das seguintes operações:

1. insere(5) 5	2. insere(8) 5 8	3. insere(2); 5 8 2
4. insere(primeiro()); 5 8 2 [...]	5. remove() 5 8 2	6. insere(remove()); 5 8 none
7. remove(); 5 8	8. insere(1) 5 8 1	9. remove(); 5 8
10. insere(remove()) 5 none	11. insere(primeiro) 5 none 5	12. insere(remove()) 5 none none
13. insere(3) 5 none none 3	14. insere(remove()) 5 none none none	15. insere(3); 5 none none none 3

5. Considere os protótipos, definidos abaixo, para uma estrutura de dados do tipo Pilha, implementada em alocação sequencial:

```
a) def empilha(self,x): - empilha um novo elemento
b) def desempilha(self): - desempilha um elemento e retornando o elemento desempilhado
c) def topo(self): - retorna o valor do elemento do topo da pilha
```

Mostre a situação de uma pilha, inicialmente vazia, após a execução de cada uma das operações:

1. empilha(1) 1	2. empilha(desempilha()) none	3. desempilha()
4. empilha(5) 5	5. empilha(9) 5 9	6. desempilha() 5
7. empilha(Topo()) 5 5	8. desempilha() 5	9. empilha(desempilha()) none
10. empilha(4) none 4	11. empilha(3) none 4 3	12. empilha(8); none 4 3 8

6. Considerando uma pilha de números inteiros e de tamanho definido pelo usuário, construa uma aplicação que:

a) preenche a pilha com números aleatórios na faixa entre 10 e 90

b) exibir a pilha

b) transfere todos os elementos da pilha para uma fila.

c) exibir a fila

```
import random

pilha = []
fila = []

menu = ( '''
    \nSelecione uma opção:

    1 - preencher com numeros aleatorios.
    2 - exibir a pilha
    3 - transferir da pilha para a fila
```

```

    4 - exibe fila

    Opção:''')

opc = 4
while opc != 9:
    print(menu)
    opc = int(input())
    if opc == 1:
        for i in range(5):
            x = random.randint(10,90)
            pilha.append(x)

    elif opc == 2:
        print('\n ', pilha)

    elif opc == 3:
        fila = pilha

    elif opc == 4:
        print ('\n ', fila)

    elif opc == 9:
        print('Programa finalizado!')

    else:
        print('Opção inválida.')
print()

```

7. Considerando uma fila de números inteiros e tamanho definido pelo usuário, construa uma aplicação que:
- preenche a fila com números aleatórios compreendidos entre 35 e 75
 - exibir a fila
 - transfere todos os elementos da fila para a pilha
 - exibir a pilha

```

import random

pilha = []
fila = []

menu = ('''
    \nSelecione uma opção:

    1 - preencher com numeros aleatorios.
    2 - exibir a fila
    3 - transferir da fila para a pilha
    4 - exibe fila
    9 - Finalizar programa.

    Opção:''')

opc = 4

```

```

while opc != 9:
    print(menu)
    opc = int(input())
    if opc == 1:
        for i in range(5):
            x = random.randint(35,75)
            fila.append(x)

    elif opc == 2:
        print('\n ', fila)

    elif opc == 3:
        pilha = fila

    elif opc == 4:
        print ('\n ', pilha)

    elif opc == 9:
        print('Programa finalizado!')

    else:
        print('Opção inválida.')
print()

```

8. Considerando os dados de alunos: matrícula e nome, construa uma aplicação que permita cadastrar e exibir, em ordem de nome, todos os alunos e as suas notas (ano, semestre, nota1, nota2).

```

dic_cadastro = {}
dic_nome = {}

menu = ('''
    \nSelecione uma opção:

    1 - Cadastrar
    2 - Listar
    9 - Finalizar Programa
''')

opc = 5
while opc != 9:
    print(menu)
    opc = int(input())
    if opc == 1:
        nomegeral = str(input("\nDigite sua matrícula: "))
        if dic_cadastro.get(nomegeral):
            print("Aluno já cadastrado.")
        else:
            nome = str(input("\nDigite seu nome: "))
            matricula = str(input("Digite seu número de matrícula: "))
            ano = str(input("Digite o ano que o aluno estuda: "))
            semestre = str(input("Digite o semestre que o aluno
estuda: "))

```



```
opc = int(input())
if opc == 1:
    temp = int(input('Digite um número: '))
    binario = bin(temp)
    print('\n', binario[2::])
elif opc == 9:
    print("Programa finalizado!")
```

A estrutura do caso pode ser pilha e fila, pois conforme se faz cada etapa da divisão, será acrescentado o número ao final da fila apenas, operação presente nos dois tipos de estruturas.