



PONTIFÍCIA UNIVERSIDADE CATÓLICA DE MINAS GERAIS
Instituto de Ciências Exatas e Informática
2ª Avaliação

Curso : Engenharia de Software
Disciplina : Algoritmos e Estruturas de Dados II
Professora : Eveline Alonso Veloso
Valor : 20 pontos
Data : 15/05/2024

Leia atentamente as instruções abaixo antes de iniciar a resolução da prova.

Instruções:

- O celular (e qualquer outro dispositivo eletrônico pessoal) deve ficar desligado durante toda a prova.
- A prova é individual e sem consulta.
- Não é permitido utilizar nenhuma estrutura de dados/método da linguagem. Utilize apenas o que foi fornecido pela professora.
- Leia atentamente cada enunciado. Respostas que não corresponderem ao que foi solicitado ou que não seguirem o especificado no enunciado da questão não serão consideradas.

Nome: Luca fernari Azalim Nota: 160

1) (2 pontos) O método de ordenação *quicksort* foi aplicado na ordenação crescente de um vetor de inteiros que apresentava, inicialmente, o conteúdo a seguir: 22; 12; 23; 11; 75; 81; 35; 50. Nesse caso, foram feitas:

- a) 16 comparações e 4 trocas efetivas entre os elementos
- b) 16 comparações e 5 trocas efetivas entre os elementos
- c) 17 comparações e 4 trocas efetivas entre os elementos
- d) 17 comparações e 5 trocas efetivas entre os elementos
- e) 18 comparações e 5 trocas efetivas entre os elementos

2) (2 pontos) Associe a estrutura de dados mais adequada (Fila, Pilha, Lista encadeada) para resolver cada problema que se segue:

a) O comando “desfazer” (*undo*) de um editor de texto precisa ser implementado. Sempre que acionado, a última ação do usuário é desfeita.

Pilha

b) Uma companhia aérea deseja automatizar o controle de reservas de seus voos. Essa companhia aérea realiza diversos voos e cada um deles tem determinado número de lugares disponíveis. Os passageiros realizam a reserva de seus lugares no voo por telefone, fornecendo seus dados de identificação.

Lista encadeada

c) Vários usuários enviam documentos para serem impressos em uma única impressora compartilhada. Deve-se controlar a ordem das impressões de forma que o usuário que

envia um documento primeiro para impressão deve ser o primeiro a ter sua solicitação atendida.

Fila ✓

- d) O histórico de sites visitados pelos usuários de um navegador tem que ser organizado de forma que o site mais recentemente visitado fique no topo da estrutura de dados.

Pilha ✓

- 3) (2 pontos) Considere o tipo abstrato de dados Pilha e suas típicas operações básicas: `empilhar(item)` e `desempilhar()`.

Suponha a existência de três pilhas inicialmente vazias: p_1 , p_2 e p_3 .

Suponha ainda que foram armazenados, em p_1 , os seguintes itens, nessa ordem: 11, 80, 32, 72, 39 e 26. Sendo assim, as somas dos valores dos itens armazenados em p_1 , p_2 e p_3 são, respectivamente: 260, 0 e 0.

Nesse contexto, analise a execução do seguinte pseudocódigo:

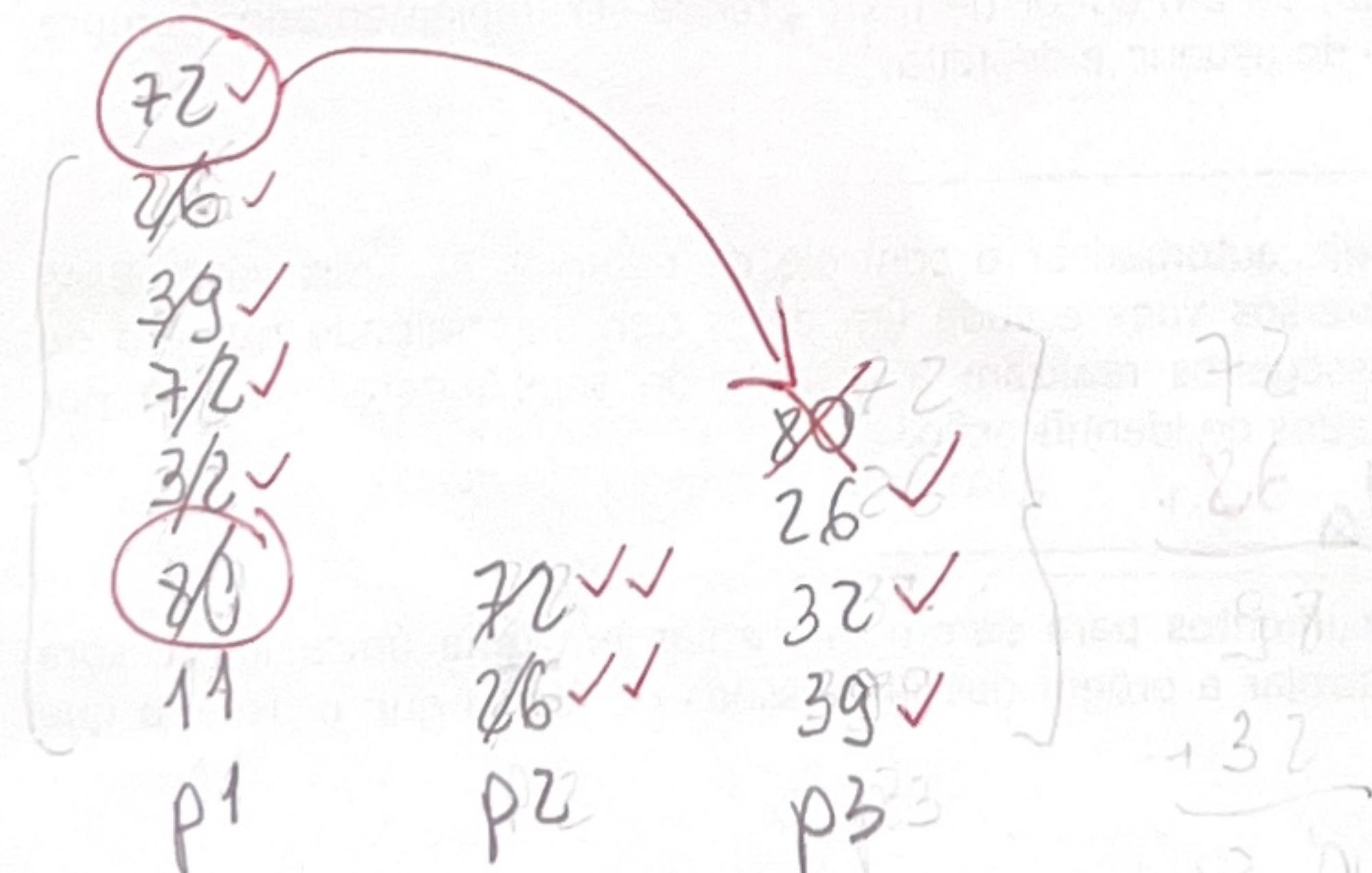
```
X p2.empilhar(p1.desempilhar());
X p3.empilhar(p1.desempilhar());
X p2.empilhar(p1.desempilhar());
X p3.empilhar(p1.desempilhar());
X p1.empilhar(p2.desempilhar());
X p3.empilhar(p2.desempilhar());
X p3.empilhar(p1.desempilhar());
```

Após a execução do pseudocódigo acima, as somas dos valores dos itens armazenados em cada uma das pilhas p_1 e p_3 serão, respectivamente:

$$p_1 = \cancel{11} \cancel{91}$$

$$p_3 = \cancel{177} \cancel{165}$$

Double checking...



$$\begin{array}{r} 106 \\ + 32 \\ \hline 138 \end{array} \quad \begin{array}{r} 138 \\ + 39 \\ \hline 177 \end{array}$$

72		
76		
39	80	
72	26	
32	32	
80	72	39
11	26	39
p_1	p_2	p_3

(11)

(177)

- 4) (2 pontos) O código abaixo, implementado na classe Pilha<E>, deveria realizar a cópia exata de uma pilha (this). O código, no entanto, possui um erro de implementação.

tmp copia

```
Pilha<E> copiar() {
    Pilha<E> tmp = new Pilha<E>(); //pilha temporária
    Pilha<E> copia = new Pilha<E>(); //pilha a ser retornada

    1. while(! vazia()) {
        2.     E item = desempilhar();
        3.     copia.empilhar(item);
        4.     tmp.empilhar(item);
    }
    5. while(! tmp.vazia())
        6.     empilhar(tmp.desempilhar());
    return copia;
}
```

Analise o código e indique a alternativa abaixo que identifica corretamente o erro de implementação apresentado por ele.

- (a) A pilha retornada não apresenta uma cópia exata da pilha original, pois seus elementos estão em ordem inversa a dos elementos da pilha original.
- (b) O erro apresentado pelo código ocorre devido ao fato da pilha original ter ficado vazia.
- (c) Os comandos 5 e 6 deveriam considerar a pilha copia, onde lê-se pilha tmp.
- (d) Os comandos 5 e 6 deveriam considerar a pilha copia, onde referencia-se a pilha this.
- (e) O erro apresentado pelo código ocorre devido ao fato dos elementos da pilha original terem sido invertidos em relação à ordem que ocupavam originalmente nessa pilha.

Respostas:

40

1	4
X C	A ✓

- 5) (6 pontos) Implemente, em Java, na classe `Lista<E>`, especificada abaixo e desenvolvida durante as aulas teóricas da disciplina, o método `public E localizar(E procurado)`, capaz de localizar na lista encadeada o item correspondente àquele que foi passado como parâmetro para esse método e retorná-lo. Essa correspondência deve basear-se no(s) critério(s) empregado(s) na implementação do método `equals` do item. Se a lista encadeada estiver vazia, ou o item correspondente àquele que foi passado como parâmetro para esse método não for localizado na lista encadeada, esse método deve lançar uma exceção.

Utilize **obrigatoriamente** as classes especificadas abaixo:

```
public class Lista<E> {  
  
    private Celula<E> primeiro;  
    private Celula<E> ultimo;  
    private int tamanho;  
}
```

```
public class Celula<T> {  
  
    private final T item;  
    private Celula<T> proximo;  
  
    public T getItem();  
    public Celula<T> getProximo();  
    public void setProximo(Celula<T> proximo);  
}
```

- 6) (6 pontos) Implemente, em Java, um programa que receba como entrada uma sequência de letras 'A', 'B' e 'C'; armazene-a em uma cadeia de caracteres (= *String*); e determine se essa *string* de entrada é da forma *xCx*, em que *x* é uma *string* consistindo somente das letras 'A' e 'B'. Por exemplo, a *string* "ABCAB" é da forma *xCx*, em que *x* = "AB"; enquanto a *string* "ABCBA" não é da forma *xCx*.

Utilize **obrigatoriamente** a estrutura de dados `Fila<E>`, especificada abaixo, para a resolução do problema proposto.

```
public class Fila<E> {  
  
    public Fila();  
    public boolean vazia();  
    public void enfileirar(E item);  
    public E desenfileirar();  
    public E consultarPrimeiro();  
}
```

Luca Ferrari Azalim

Questão 5:

public E localizar (E procurado) {

Celula <E> cel = this.primeiro;

for (int i = 0; i < this.tamanho; i++) {

6,0 cel = cel.getProximo();

if (Objects.equals(cel.getitem(), procurado)) {

return cel.getItem();

}

}

throw new NoSuchElementException();

}

Questão 6:

```
public boolean checar(String str) {  
    fila<char> fila = new fila<>();  
    for (int i = 0; i < str.length(); i++) {  
        if (str.charAt(i) == 'c') {  
            break;  
        }  
        fila.enfileirar(str.charAt(i));  
    }  
    for (int i = (str.length() / 2 + 1); i < str.length(); i++) {  
        if (!fila.desenfileirar().equals(str.charAt(i))) {  
            return false;  
        }  
    }  
    return (fila.tamanho() * 2 + 1) == str.length();  
}
```