A3 - Estrutura de Dados

Jogo De Classificação Por Bolas Coloridas
&
Simulação de uma Fila de Impressão com Prioridades

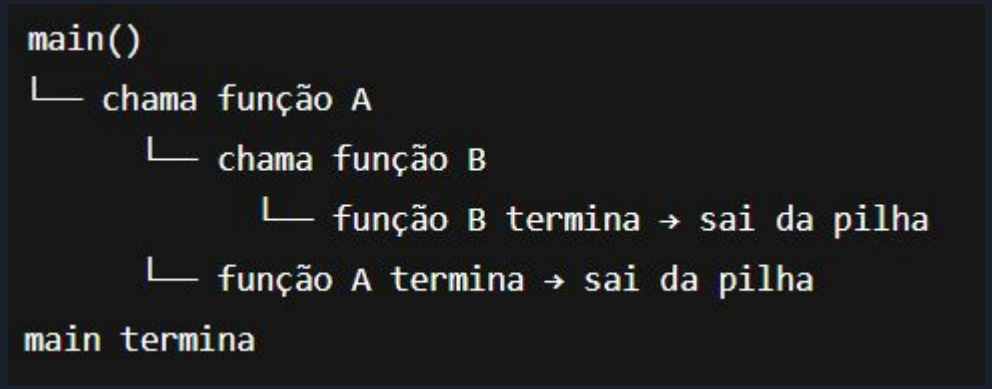
Henrique Cavalcanti Rocha - 12722117519
Marcos Vinicius Lima Ribeiro - 12723116626
Pedro Martins Caires - 12722124034
Rafael Rodrigues Figueiredo - 12722130532

O que é uma Pilha?

A pilha é uma estrutura de dados linear que segue o princípio LIFO (Last In, First Out), onde o último elemento inserido é o primeiro a ser removido. Os principais métodos são push (inserir), pop (remover) e peek (acessar o topo).

Funciona como uma pilha de pratos: só o item do topo pode ser manipulado.

É amplamente usada em chamadas de funções, verificação de sintaxe e avaliação de expressões, sendo uma das primeiras estruturas ensinadas em algoritmos pela sua simplicidade e utilidade.



Métodos da Pilha de Dados

Este trabalho apresenta o desenvolvimento do **StackColor**, um jogo computacional baseado na estrutura de dados do tipo pilha, que organiza bolas coloridas em sequência, onde cada bola contém um ponteiro para o próximo item da pilha.

O jogo simula sete pilhas, cada uma com capacidade para até sete bolas coloridas, sendo que uma pilha começa vazia para permitir a reorganização das demais, seguindo o princípio **LIFO**.

Estrutura de Dados utilizada: **Pilha**
Métodos:

- **stack(Ball novo);**
- **unstack();**
- **isEmpty();**
- **isFull();**
- **showTop();**
- **showStack();**
- **clear();**
- **winnerStack();**
- **getColorIndex(int position)**

Q.02 - Jogo De Classificação Por Bolas Coloridas

Q.02 - Classes Principais

O código está organizado no pacote ***pilha*** e é composto por quatro arquivos principais:

- ***GameBall.java***

Responsável por **exibir** o **menu principal** do jogo.

- ***PlayingGame.java***

Responsável por **executar** o **fluxo do jogo**, permitindo que o jogador **mova bolas** entre **pilhas**, mostrar o **estado atual** das **pilhas** a cada rodada, **validações de entrada** e **tratamento de exceções**.

Q.02 - Classes Principais

- ***MethodsGame.java***

Responsável por **inicializar** o **estado do jogo**, garantir a **distribuição** válida de **bolas** com cores limitadas, e implementar a **lógica** de **movimentação de bolas** entre **pilhas**.

- ***Pilha.java***

A classe ***Pilha*** implementa a **estrutura de dados pilha de bolas**, com limite de **7 elementos**.

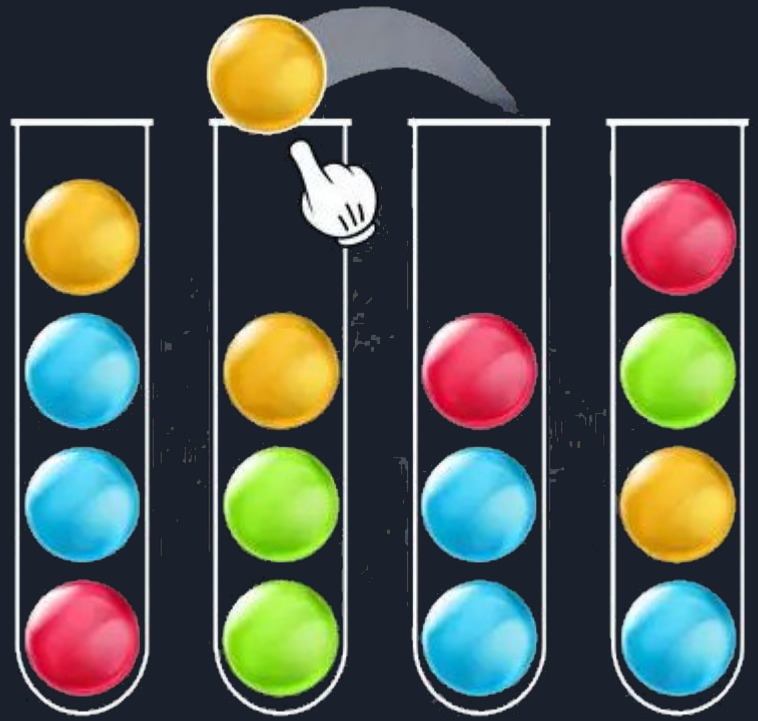
Q.02 - Benefícios Educacionais

O jogo ***StackColor*** oferece diversos **benefícios educacionais** no ensino de **estruturas de dados**, simulando o funcionamento de **pilhas** por meio de uma mecânica **lúdica** e **interativa**, e permite que os alunos compreendam, na prática, o princípio **LIFO**.

A manipulação de bolas coloridas ajuda a visualizar como os dados são:

- Armazenados
- Acessados
- Removidos

Reforça o entendimento da **estrutura linear** e de suas **operações básicas**, conectando a teoria à prática, promovendo uma aprendizagem mais **significativa** e **acessível**.



Q.05 - Simulação de uma Fila de Impressão com Prioridades

Q.05 - Simulação de uma Fila de Impressão com Prioridades

Este trabalho tem como objetivo criar um código para simular uma ***Fila de Impressão com Prioridades***, utilizando o paradigma de programação orientada a objetos em **Java**.

Estrutura de Dados utilizada: **Fila**

Métodos:

- **insertHeap(Document newDocument);**
- **insert(Document newDocument);**
- **remove();**
- **showFirst();**
- **showLast();**
- **isEmpty();**
- **size();**
- **toString()**

Q.05 - Classes Principais

O código está organizado no pacote ***fila*** e é composto por quatro arquivos principais:

- ***ProgramPrint.java***

Responsável pela **criação** de **documentos pré-definidos**, com **título**, **conteúdo** e **nível de prioridade**, **inserção** desses **documentos** em uma **fila de prioridade (*WaitingLine*)**, e da **exibição** de um **menu interativo**.

- ***WaitingLine.java***

A classe ***WaitingLine*** implementa a **lógica** da **fila de documentos**.

Q.05 - Classes Principais

- ***WaitingLineException.java***

A classe ***WaitingLineException*** é uma **exceção personalizada** para tratar **erros** relacionados à **fila de documentos**.

- ***ShowMenu.java***

Responsável por **exibir** o **menu principal**, as **funcionalidades do menu**, e **limpar o console**.

O que é uma Fila Heap?

A **Fila Heap**, ou **fila de prioridade**, é uma **estrutura de dados** que **organiza** os **elementos** conforme sua **prioridade**, e não pela **ordem de chegada**.
Usando uma estrutura de **árvore** armazenada em **vetor**, ela garante que o **item mais importante** fique sempre no **topo**.
Existem **dois** tipos: ***Max Heap* (maior valor no topo)** e ***Min Heap* (menor valor no topo)**, e ambos **reorganizam automaticamente** os **elementos** a cada **inserção** ou **remoção**.
É muito **eficiente** e **útil** em sistemas que exigem **decisões rápidas** e **ordenadas**.

```
public void insertHeap(Document newDocument) throws WaitingLineException{
        WaitingLine filaAux = new WaitingLine();  //Aqui eu instancio a fila auxiliar para o corte da estrutura de dados

        if (start == null) {
                start = newDocument;
                end = newDocument;
        } else {
                boolean insert = false;
                Document temp = this.start;
                while(temp!=null) {
                        if(!insert && newDocument.getPriority().getPriorityInt() < temp.getPriority().getPriorityInt()) {
                                filaAux.insert(newDocument);
                                insert = true;
                        }
                        filaAux.insert(temp);
                        temp = temp.getNext();
                }
                if(!insert) {
                        filaAux.insert(newDocument);
                }

                this.start = filaAux.showFirst();
                this.end = filaAux.showLast();
        }

}
```

Diferenças entre Pilha e Fila Heap

Pilha e ***Fila Heap*** são **estruturas de dados** com **propósitos diferentes**.
A ***Pilha*** segue o princípio **LIFO (Last In, First Out)** e é usada para situações como **desfazer ações** ou **chamadas de função**.
Seus principais métodos são ***push*** **(inserção)** e ***pop*** **(remoção do topo)**.

Já a ***Fila Heap*** organiza elementos por **prioridade**, não pela ordem de chegada, seguindo o modelo de uma **árvore binária completa**.
Os principais métodos são ***insert*** **(inserir mantendo a ordem)** e ***remove/extract*** **(retirar o elemento com maior ou menor prioridade)**.

Enquanto a ***Pilha*** é **linear** e **rápida (tempo constante)**, a ***Fila Heap*** é **hierárquica** e **mais complexa**, com operações em **tempo logarítmico**.