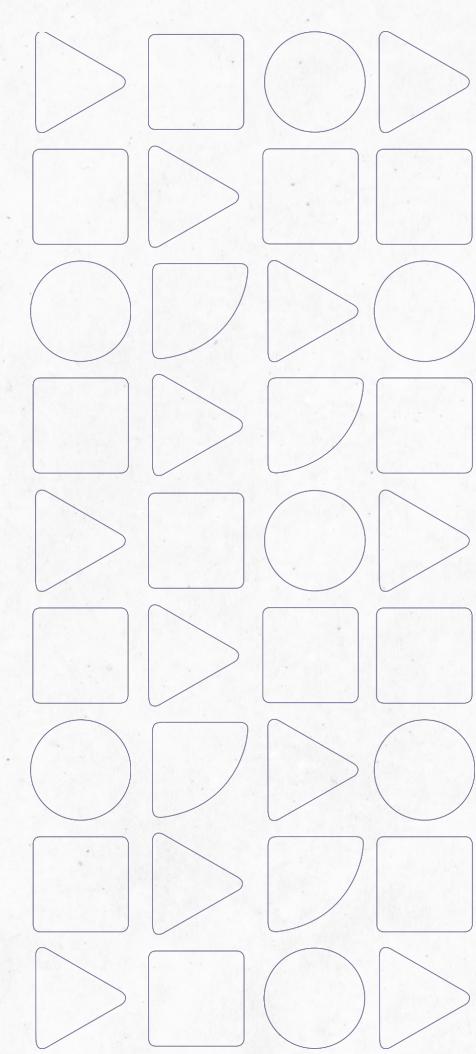


Vetores e matrizes em Java

Disciplina: Desenvolvimento de Aplicações



Conteúdos:

Vetores e matrizes em Java.

Habilidade(s):

Bloco 1

Como você está se sentindo hoje?



Arrays

Arrays são estruturas de dados que **armazenam múltiplos elementos do mesmo tipo**.

É uma maneira eficaz de lidar com múltiplos valores relacionados em uma única variável.

Características dos arrays



Armazenam valores em posições sequenciais.



São utilizados para armazenar coleções de dados.



São úteis para trabalhar com grandes volumes de informações.

Declaração e inicialização de arrays

Declaração

```
java  
tipo[] nomeDoArray;
```

Declaramos o tipo e o nome do *array*.

Inicialização

```
java  
int[] numeros = new int[5];
```

Inicializamos o *array* com um tamanho fixo.

Indexação em *arrays*

Elementos de um *array* são acessados por meio de índices.

Os índices começam em 0.

Exemplo: ‘numeros[0]’ acessa o primeiro elemento do *array*.

java

```
numeros[0] = 10; // Atribuindo o valor 10 ao primeiro elemento
```

Exemplo de indexação em *arrays*

```
java
```

```
int[] numeros = new int[3];
numeros[0] = 5;
numeros[1] = 10;
numeros[2] = 15;
```

Importância dos arrays

- Manipular grupos de dados de maneira organizada;
- Facilitar operações em conjunto de informações;
- Acessar elementos específicos de forma eficiente.

Vamos praticar?

Peguem uma folha para realizar a atividade.

Atividade



15 minutos

Criem um mapa mental sobre *array* e como utilizá-lo.
Lembrem-se de detalhá-lo para lhe auxiliar no momento de
criar um código.



Bloco 2

Fofoca do bem

Contem para a turma, em forma de fofoca, um pouco sobre *arrays*.





Vamos praticar?

Dividam-se em grupos para realizar a atividade.

Primeiro momento

30 minutos

Criem um pequeno programa utilizando *arrays*.

Podem utilizar os mapas mentais já construídos para ajudar na atividade.

Segundo momento

10 minutos

Mostrem o código para a turma.



Bloco 3

Matrizes

Matrizes são estruturas de dados que **permitem armazenar e acessar informações em formato de tabela**. Podem ser unidimensionais (*arrays*) ou multidimensionais.

O conhecimento sobre matrizes é essencial para trabalhar com dados estruturados e complexos.

Tipos de matrizes

Matrizes unidimensionais

Armazenam elementos em uma única linha e permitem o acesso aos elementos através de índices.

Matrizes multidimensionais

Armazenam elementos em linhas e colunas e permitem o acesso através de índices em cada dimensão.

Declaração e inicialização de matriz unidimensional

Declaração

```
java
```

```
tipo[] nomeDaMatriz;
```

Inicialização

```
java
```

```
int[] numeros = new int[5];
```

É o mesmo conceito de *arrays*, mas com uma dimensão adicional. Matrizes unidimensionais são indexadas da mesma forma que *arrays*.

Declaração e inicialização de matriz multidimensional

Declaração

```
java
```

```
tipo[][] nomeDaMatriz;
```

Inicialização

```
java
```

```
int[][] matriz = new int[3][3];
```

Declaramos a matriz com duas dimensões e especificamos o tamanho delas.



Acessando um dado na matriz multidimensional

```
java
```

```
matriz[0][0] = 5; // Atribuindo valor ao primeiro elemento
```

Usamos índices para acessar elementos em cada dimensão.

Exemplo de matriz multidimensional

java

```
int[][] matriz = {  
    {1, 2, 3},  
    {4, 5, 6},  
    {7, 8, 9}  
};
```

Importância das matrizes

- Para representar dados em formato tabular;
- Úteis para armazenar informações em grades;
- Utilizadas em jogos, imagens, cálculos matriciais etc.

Vamos praticar?

Peguem uma folha para realizar a atividade.

Atividade

15 minutos

Criem um mapa mental sobre matrizes e como utilizá-las.
Lembrem-se de detalhá-lo para lhe auxiliar no momento de
criar um código.



Bloco 4

O que vem à sua mente quando ouve falar sobre matrizes?



Vamos praticar?

Dividam-se em grupos para realizar a atividade.

Primeiro momento

30 minutos

Criem um pequeno programa utilizando matrizes. Podem utilizar os mapas mentais já construídos para ajudar na atividade.

Segundo momento

10 minutos

Mostrem o código para a turma.



Bloco 5

Oi, falso!

Contem para a turma, como se estivessem falando para um amigo que faltou
à aula, um pouco sobre *arrays* e *matrizes*.



Vamos praticar?

Dividam-se em grupos para realizar a atividade.

Primeiro momento

30 minutos

Criem um pequeno programa utilizando *arrays* e *matrizes*.

Podem utilizar os mapas mentais já construídos para ajudar na atividade.

Segundo momento

10 minutos

Mostrem o código para a turma.



Bloco 6



Fale uma palavra que te faça
lembrar de *array* ou matriz.

Arrays

VANTAGENS

Armazenamento eficiente

Acesso direto

Simplicidade

Iteração simples

VS

DESVANTAGENS

Tamanho fixo

Ineficiente para inserções e
remoções

Espaço desperdiçado

Complexidade de código

Mas...

quando usar *arrays*?



Quando usar *arrays*

- Quando o tamanho é conhecido e fixo;
- Para armazenar e processar coleções simples de dados;
- Em situações que requerem acesso rápido aos elementos.

Vamos praticar?

Atividade

10 minutos

Pesquise, no celular, um *software* que utilize *arrays* no seu código e veja como ele funciona. Em seguida, fale o que você encontrou para a turma.



Discussões e reflexões



Fechamento

Vamos discutir?

O que eu não posso esquecer?

O que eu desejo aprofundar?

Anote três tópicos diferentes que você aprendeu na aula.

Anote três tópicos da aula para você pesquisar quando chegar em casa.



Referências Bibliográficas

PROZ EDUCAÇÃO. *Apostila de Desenvolvimento de Aplicações*. 2023.