Fundamentos de Java
6-1: Matrizes
Atividades Práticas

Objetivos da Lição:

- Escrever um array unidimensional em um programa Java usando tipos de dados primitivos
- Escrever um array unidimensional em um programa Java usando tipos de referência (Objeto)
- Escrever um array bidimensional em um programa Java usando tipos de dados primitivos
- Escrever um array bidimensional em um programa Java usando tipos de referência (Objeto)
- Declarar um array, inicializar um array e atravessar o array
- Descrever a inicialização do array
- Diferenciar o método String length() e o valor do comprimento do array
- Reescrever um programa Java para armazenar números inteiros em um array, realizar um cálculo matemático e exibir o resultado
- Usar uma sintaxe alternativa de declaração de array

Vocabulário:

Identifique a palavra do vocabulário para cada definição a seguir.

	O ato de processar por meio de um array
	Uma estrutura que armazena vários valores do mesmo tipo de dados
	Um array bidimensional.
	Um número inteiro que identifica o local de um valor em um array
	A capacidade de passar dados para a função principal e acessá-los como um elemento de um array.
	Um procedimento lógico de computação que, se aplicado corretamente, garante a solução de um problema
	Um array de arrays, semelhante a uma tabela, matriz ou planilha.
	Um loop positivo dentro de um loop positivo
	Um objeto nomeado usado para armazenar mais de um valor.

Tente/solucione:

1. Declare uma pontuação do nome de array de uma dimensão do tipo int que possa conter 9 valores.

2. Declare um array bidimensional chamado price do tipo flutuante que tenha 10 linhas e 3 colunas.

3. Declare e inicialize um array bidimensional denominado matrix do tipo long que tenha 4 linhas e 3 colunas para ter todos os valores definidos como 5.

4. Declare e inicialize um array de byte unidimensional denominado values de tamanho 10 para que todas as entradas contenham 1.

5. Sem digitar o código, determine a saída do seguinte programa.

```
int num[] = {7,7,6,6,5,5,4,4};
for(int i = 0; i < 8; i = i + 2)
        System.out.print(num[i]);
```

6. Sem digitar o código, determine a saída do seguinte programa.

```
int[][] num = {{3,3,3},{2,2,2}};
int[] array = {4,3,2};
for(int i = 0; i < 3; i++){
        num[1][i] = num[0][i]+array[i];
}

for(int i = 0; i < 2; i++){
        for(int j = 0; j < 3; j++){
                System.out.print(num[i][j]);
        }
        System.out.println();
}
```

7. Em determinada aula, há 5 testes, cada um valendo 100 pontos. Crie um programa que considere as pontuações dos 5 testes do usuário, armazene as pontuações em um array e, em seguida, calcule a média dos alunos.8.

8. Na aula de Álgebra, aprendemos sobre matrizes. Aprendemos a somar, subtrair e multiplicar matrizes 2x2 e 3x3. Veja abaixo alguns exemplos da aula de Álgebra com as respostas:

$$\begin{pmatrix}3 & 4\\5 & 6\end{pmatrix}+\begin{pmatrix}1 & 0\\-2 & 3\end{pmatrix}=\begin{pmatrix}4 & 4\\3 & 9\end{pmatrix}$$

$$\begin{pmatrix}3 & 4\\5 & 6\end{pmatrix}-\begin{pmatrix}1 & 0\\-2 & 3\end{pmatrix}=\begin{pmatrix}2 & 4\\7 & 3\end{pmatrix}$$

$$\begin{pmatrix}3 & 4\\5 & 6\end{pmatrix}*\begin{pmatrix}1 & 0\\-2 & 3\end{pmatrix}=\begin{pmatrix}-5 & 12\\-7 & 18\end{pmatrix}$$

É praticamente óbvio como somar. Somamos a primeira posição na primeira matriz com a primeira posição na segunda matriz. Continuamos com as posições correspondentes para obtermos a resposta. A subtração segue o mesmo método de posições. A multiplicação de matrizes parece confusa, pois não segue o método de posições usado na soma e na subtração.

A resposta é obtida pegando a linha da primeira matriz e a coluna da segunda e multiplicando os respectivos valores e, em seguida, pegando a soma dos produtos.

A resposta acima foi obtida da seguinte forma:

3(1)+4(-2)=-53(0)+4(3)=12

5(1)+6(-2)=-75(0)+6(3)=18

Crie um programa que considere as duas matrizes e, em seguida, permita que o usuário opte por somá-las, subtraí-las ou multiplicá-las exiba a resposta. O programa exibirá o seguinte menu:

a. Enter Matrix A

b. Enter Matrix B

c. Display A + B

d. Display A - B

e. Display A * B

f. Exit

O programa deve ser executado e permitir que o usuário continue escolhendo diferentes opções até decidir sair. O programa bem gravado modulará o processo em diferentes métodos.

9. Digite o seguinte código e teste. Adicione as seguintes características.

a. Adicione um método de embaralhamento à Classe Deck. Mostre o método na classe Main para verificar se o baralho está realmente embaralhado.s. Call the method from the Main class to verify that the deck is indeed shuffled.

b. Adicione uma Classe Hand que contenha um conjunto de 5 referências de Card. Faça com que o programa veja as duas cartas da mão e exiba-as para o usuário. Informe ao usuário quantos pontos ele tem e pergunte se ele deseja outra carta. Continue permitindo que o jogador adicione cartas até que ele consiga 5 cartas ou até que o total seja maior que 21.

c. Ajuste a classe Card para permitir que os Aces contem como 11 para começar. Se a Classe Hand tiver um valor maior que 21, faça com que a Classe Hand procure Aces e reduza o valor da pontuação para 1.

d. Faça com que o programa crie uma Banca que o usuário possa jogar contra. O usuário deve tentar chegar o mais próximo de 21 sem fazer esforço para derrotar a Banca. Se a Banca tiver 16 ou mais, ela deverá parar de pegar cartas.

```
public class Main {

    public static void main(String args[]){

                Deck d = new Deck();

                d.print();

      }



}



public class Deck {

    Card[] cardArray = new Card[52];

    Deck(){ //constructor

          int suits = 4;

          int cardType = 13;

          int cardCount = 0;

           for(int i = 1; i <= suits; i++)

                 for(int j = 1; j <= cardType; j++){

                        cardArray[cardCount] = new Card(i,j);

                        cardCount++;

                 }

        }

       public void print(){

                 for(int i = 0; i < cardArray.length; i++)

                       System.out.println(cardArray[i]);

        }

}

public class Card{

                  String suit,name;
```

```
            int points;
    Card(int n1, int n2){
        suit = getSuit(n1);
        name = getName(n2);
        points = getPoints(name);
    }
    public String toString(){
            return "The " + name + " of " + suit;
    }
    public String getName(int i){
        if(i == 1)  return "Ace";
        if(i == 2)  return "Two";
        if(i == 3)  return "Three";
        if(i == 4)  return "Four";
        if(i == 5)  return "Five";
        if(i == 6)  return "Six";
        if(i == 7)  return "Seven";
        if(i == 8)  return "Eight";
        if(i == 9)  return "Nine";
        if(i == 10)  return "Ten";
        if(i == 11)  return "Jack";
        if(i == 12)  return "Queen";
        if(i == 13)  return "King";
        return "error";
    }
    public int getPoints(String n){
       if(n == "Jack" ||n == "Queen" ||n == "King"||n == "Ten")
          return 10;
       if(n == "Two")
```

```
            return 2;
        if(n == "Three")
            return 3;
        if(n == "Four")
            return 4;
        if(n == "Five")
            return 5;
        if(n == "Six")
            return 6;
        if(n == "Seven")
            return 7;
        if(n == "Eight")
            return 8;
        if(n == "Nine")
            return 9;
        if(n == "Ace")
            return 1;
        return -1;
     }
    public String getSuit(int i){
        if(i == 1)  return "Diamonds";
        if(i == 2)  return "Clubs";
        if(i == 3)  return "Spades";
        if(i == 4)  return "Hearts";
        return "error";
    }
}
```