Programação Orientada a Objetos

Educação a Distância
Cruzeiro do Sul Educacional
Campus Virtual

Material Teórico

Introdução a Vetores

Responsável pelo Conteúdo:
Prof. Esp. Alexander Gobbato Albuquerque

Revisão Textual:
Profa. Ms. Magnólia Gonçalves Mangolini

UNIDADE Introdução a Vetores

- Introdução a vetores

Objetivo de APRENDIZADO

Nesta unidade iremos enriquecer nosso conhecimento com novos conceitos.

Nós já estudamos os tipos de variáveis e como criá-las, e vimos também que nem sempre uma variável atenderá nossas necessidades.

Para aprimoramos nosso desenvolvimento, aprenderemos o conceito de vetores e como utilizá-los juntamente com os objetos e também passagem de parâmetros.

Hoje veremos alguns assuntos introdutórios, na nossa disciplina. Aproveito para apresentar-lhes alguns conceitos que utilizaremos na estrutura de todas as nossas unidades.

Para obter um bom aproveitamento nesta unidade, vamos conferir sua estrutura:

- Conteúdo Teórico - neste link você encontrará o material principal de estudos na forma de texto escrito.
- Atividade de Sistematização - os exercícios disponibilizados são de autocorreção; nessa atividade, você vai praticar o que aprendeu na disciplina e identificar os pontos em que precisa prestar mais atenção, ou pedir esclarecimentos a seu tutor. As notas atribuídas aos exercícios serão parte de sua média final na disciplina.
- Atividade de Aprofundamento - é uma atividade dissertativa.
- Material Complementar e Referências - nestes links você poderá ampliar seus conhecimentos.
- Videoaula - neste link serão apresentadas algumas ferramentas na prática e a resolução de alguns exercícios também de forma prática.

Lembramos a você da importância de realizar todas as atividades propostas dentro do prazo estabelecido para cada Unidade, dessa forma, você evitará que o conteúdo se acumule. Evitará ainda problemas ao final do semestre.

Uma última recomendação: caso tenha problemas para acessar algum item da disciplina, ou dúvidas com relação ao conteúdo, entre em contato com seu professor tutor através do botão *Mensagens*.

Contextualização

Já estudamos, nos capítulos anteriores, o encapsulamento de informações e a reutilização de códigos. Vimos também como criar os métodos, usar encapsulamento e chamar os métodos criados através de objetos.

Nessa unidade veremos os conceitos de vetores, e com o conceito aprendido poderemos criar estruturas complexas para atender a demanda de desenvolvimento.

Vamos aprender mais sobre vetores!!!

Introdução a vetores

Antes de começar a falar de arrays, devemos lembrar o que é uma variável e para que ela serve.

Nas unidades anteriores, vimos que variável é a representação simbólica dos elementos, e que ficam armazenadas em memória. Cada variável corresponde a uma posição, cujo conteúdo pode se alterado ao longo do tempo durante a execução de um programa. No entanto, a variável só pode ser armazena por um único tipo.

Também vimos que as variáveis podem ser de três tipos: numéricas, alfanuméricas e lógicas, e que para declarar uma variável precisamos definir o seu nome e que tipo de dados será armazenado nela.

Antes de definir vetor, imaginemos a seguinte situação:

Precisamos criar um programa que armazene as seguintes notas de um aluno: 8.0, 10.0, 9.0, 10.0, 8.5, 10.0 e que calcule a média final.

A solução é bem simples. Definiremos uma variável, alocamos um espaço na memória do computador para armazenar uma e somente uma constante por vez, seja ela literal, numérica ou lógica. Quando atribuímos um valor à variável sobrescrevemos seu conteúdo.

Em algumas rotinas é necessário a manipulação de várias variáveis ao mesmo tempo, por exemplo, imagine que o programa deverá controlar o nome de 100 pessoas, ao invés de criar 100 variáveis é possível à criação de apenas uma, que é definida como array.

Mas como criar um array? É muito simples: especificamos o nome do array e o número de posições da memória que queremos alocar. Cada posição de memória pode armazenar um valor.

Os arrays são criados para que possam armazenar várias informações do mesmo tipo ou da mesma classe; com um vetor também é possível o armazenamento de vários objetos. Essas diversas informações que foram armazenadas em array são de fácil manipulação. As informações ficam disponíveis em forma de tabelas e podem ser acessadas por meio do índices.

Nas unidades anteriores vimos que para se chamar um programa em Java, devemos criar um programa principal com o método main, e como parâmetro do método devemos criar um string args[], isso nada mais é do que um array de Strings.

Os arrays estão presentes em todas as linguagens de programação, em Java, os arrays são objetos que permitem armazenar uma lista de itens relacionados.

Arrays Unidimensionais

Os arrays de uma dimensão são aqueles que possuem um único índice para acessar os elementos. Eles podem ser declarados da seguinte forma:

TIPO-DE-DADO NOME-DO-ARRAY[] = NEW TIPO-DO-DADO[QTDE].

Em que:

Tipo-de-dado: Qualquer tipo de variável ou de classe;

Nome-do-array: Qualquer nome válido vale o mesmo conceito para a criação dos nomes de variáveis.

Exemplos:

Criar um array de nome Cidades contendo 50 elementos do tipo String e seu índice varia de 0 a 49.

```
String Cidades[] new String[50];
```

Criar um array de nome mes contendo 12 elementos do tipo int e seu índice varia de 0 a 11.

```
int mes[] new int[12];
```

Para se atribuir valores a um vetor (array), devemos colocar o índice desejado entre os colchetes, como demonstrado abaixo:

```
Cidades[0] = "São Paulo";
Cidades[10] = "Rio de Janeiro";
mes[0] = 1;
mes[11] = 12;
```

O exemplo que será demonstrado a seguir utiliza um array para armazenar um conjunto de argumentos do tipo inteiro, informado pelo usuário na linha de execução:

```
class Exemplo{
    public static void main(String args[]){
        int i, total=0;
        int N[]=new int[10];

        if(args.length>0){

            try{

                for(i=0;i<args.length;i++){
                    N[i] = Integer.parseInt(args[i]);
                    total = total + N[i];
                }

                System.out.println("O Total de números digitados na ordem inversa é:")

                for(i=args.length;i<=0;i--){
                    System.out.println(N[i] + " ");
                }
            }catch(NumberFormatException E){
                System.out.println("Os argumentos devem ser números inteiros");
            }
        }else{
            System.out.println("Digite pelo menos um número");
        }
    }
}
```

Exemplo do livro Java 2 - Ensino Didático (pag. 142)

Os vetores (arrays) podem ser criados e inicializados ao mesmo tempo, para isso, deve-se utilizar o operador new para instanciar um novo objeto. Os arrays criados entre chaves e separados por vírgula devem ter o mesmo tipo que a variável. Veja a sintaxe abaixo:

TIPO-DE-DADO NOME-DO-ARRAY[] = { valores separados por vírgula }

O exemplo abaixo mostra como usar, e explica também a função String.valueOf() para manipulação do conteúdo de um array de caracteres.

```
1  public class Exemplo {
2      public static void main(String args[]){
3          String nomes="";
4
5          char CaracterArray[] = ('a','b','c','d','e','f','g');
6
7          System.out.println("Mostrando o array " + String.valueOf(CaracterArray));
8
9          System.out.println("Quant. de elementos: " + CaracterArray.length);
10
11         System.out.println("1° ao 3° caracter " + String.valueOf(CaracterArray,0,3));
12
13         String StringArray[] = ("Aprendendo","a","utilizar","array");
14
15         for(int =0;i<StringArray.length;i++){
16             nomes = nomes + StringArray[i] + " ";
17         )
18
19         System.out.println("Mostrando array de " + nomes);
20
21         System.out.println("Quant. de elementos do array: " + StringArray.length)
22
23         System.out.println("Mostrando o 1° elemento: " + StringArray[0]);
24
25         System.out.println("Mostrando o último elemento do array: " + StringArray[StringArray.length - 1]);
26     )
27 )
```

Linha 5: criando array de caracteres.

Linha 9: mostrando a quantidade de elementos do array.

Linha 15: armazenando informações no array.

Informação

Alguns pontos devem ser levados em consideração quando se trabalhar com vetores (arrays) de caracteres e strings: nos array de caracteres são utilizados ‘ (apóstrofos) e já nos arrays de string são utilizados “ (aspas).

A função String.valueOf() pode ser usada para apresentar os elementos do array de caracteres ou um trecho dele. A função String.valueOf() não funciona para um conjunto de arrays do tipo string, apenas do tipo char(conjunto de caracteres).

Arrays Bidimensionais

Os arrays bidimensionais ou arrays multidimensionais permite a criação de vetores com mais de um índice. Essa característica possibilita que os valores sejam armazenados na forma de matrizes de qualquer dimensão.

TIPO-DO-DADO NOME-DO-ARRAY[][] = NEW TIPO-DO-DADO[<índice>][<índice>].

O exemplo abaixo demonstra o uso de arrays bidimensionais para armazenar 2 (duas) notas de 3 (três) alunos (total de 6 notas). Uma vez armazenadas, o programa solicita ao usuário o número do aluno para exibir suas notas e também a média.

```
1  import javas.swing.*;
2  public class Exemplo {
3      public static void main(String args[]){
4          float notas[][]=new float[3][2];
5          int aluno=0, nota;
6
7          while(aluno<3){
8              nota=0;
9              while(nota<2){
10                 System.out.println("Aluno " + (aluno +1) + ", digite a " + (nota+1) + "ª nota:");
11                 notas[aluno][nota] = Float.parseFloat(JOptionPane.showInputDialog("Informe a nota do aluno"));
12                 nota++;
13             }
14             aluno++;
15         }
16
17         System.out.println("Digite o número do aluno de 1-3");
18         aluno = Integer.parseInt(JOptionPane.showInputDialog("Digite o nr. do aluno:"))
19         System.out.println("Aluno: " + aluno);
20         System.out.println("Nota1: " + notas[aluno-1][0] + " Nota2: " + notas[aluno-1][1]);
21         System.out.println("Média: " + ( ((notas[aluno-1][0]) + notas[aluno-1][1])/2 );
22     }
23 }
24
25
26
```

Arrays de objetos

Do mesmo jeito que criamos array de dados primitivos (int, float, Double, string e char), é possível criar um array para armazenamento de objetos. Isso é muito útil, pois permite as mesmas operações com diversos objetos do mesmo tipo.

O exemplo abaixo utiliza um array de objetos, aproveitando das funcionalidades da classe Veículo.

```
public class Veiculo{
    private int velocidade;

    public void setVelocidade(int vVelocidade){
        velocidade = vVelocidade;
    }

    public int getVelocidade(){
        return velocidade;
    }
}


public class Exemplo {
    public static void main(String args[]){
        Veiculo veic[]=new Veiculo[300];

        for(int indice=0; indice<300;indice++){
            veic[indice] = new Veiculo();
        }

        veic[0].setVelocidade(10);
        veic[10].setVelocidade(200);
        veic[250].setVelocidade(150);

        for(int i=0;i<300;i++){
            veic[i].setVelocidade(0);
        }
    }
}
```

Passagem de arrays em métodos

Os métodos já foram vistos nos capítulos anteriores. Vimos que é possível a criação de métodos que recebem valores, manipulam esses valores e retornam um resultado.

A passagem de arrays em métodos é basicamente o mesmo das variáveis. Quando um método é invocado, um vetor qualquer é informado; esse vetor pode ser manipulado internamente pelo método e também retornar o resultado.

Public static TIPO_DE-ARRAY[] nome-do-método (tipo-do-array nome-do-array[])

O exemplo a seguir mostra o método que recebe um array do tipo inteiro, organiza seus elementos e retorna o array.

```
import javas.swing.*;
public class Exemplo {
    public static void main(String args[]){
        int numeros[] = new int[10];
        for(int i=0; i < 10; i++){
            JOptionPane.showMessageDialog(null,);
            numeros[i] = Integer.parseInt(JOptionPane.showInputDialog("Digite o numero " + (i+1)));
        }

        numeros = ordenaArray(numeros);
        mostraArray(numeros);
    }

    public static int[] ordenaArray(int arr[]){
        int x,y,aux;
        for(x=0;x<arr.length;x++){
            for(y=0;y<arr.length;y++){
                if(arr[x]<arr[y]){
                    aux=arr[y];
                    arr[y]=arr[x];
                    arr[x]=aux;
                }
            }
        }
        return arr;
    }

    public static void mostraArray(int arra[]){
        for(int i=0;i<arr.length;i++){
            System.out.println(arr[i] + " ");
        }
    }
}
```

Material Complementar

Livros Disponíveis na Biblioteca Virtual Universitária da Pearson Education.

- Aprenda Programação Orientada a Objetos em 21 dias (Anthony Sintes)
- Java Como Programar, 8ª edição (Paul Deitel e Harvey Deitel)
- Core Java – Volume 1 (Cay Horstman e Gary Cornell)

Conceitos de POO em inglês, site da SUN:

http://download-llnw.oracle.com/javase/tutorial/java/concepts/index.html

Referências

DEITEL, P.; DEITEL, H. *Java Como Programar.* 8. ed. São Paulo: Pearson Education do Brasil, 2010.

HORSTMANN, C.S.; CORNELL, G. *Core Java.* 8. ed. São Paulo: Pearson Education do Brasil, 2010, v. 1.

SINTES, Tony. *Aprenda Programação Orientada a Objetos em 21 dias.* São Paulo: Pearson Education do Brasil, 2002, v. 1.

Java 2 Ensino Didático.

Anotações


Educação a Distância
Cruzeiro do Sul Educacional
Campus Virtual

www.cruzeirodosulvirtual.com.br
Campus Liberdade
Rua Galvão Bueno, 868
CEP 01506-000
São Paulo SP Brasil
Tel: (55 11) 3385-3000

Universidade Cruzeiro do Sul

UNICID Universidade Cidade de S. Paulo

UNIFRAN Universidade de Franca

UDF Centro Universitário

Módulo Centro Universitário