UNINASSAU
Campus Graças

Estrutura de Dados

PROFESSOR:

Adilson da Silva

CURSO (2024.1)

CURRÍCULO DO DOCENTE

Prof. Adilson da Silva

UNINASSAU
Campus Graças

Graduado em Ciência da Computação pela Universidade Católica de Pernambuco. Pós-graduado em Tecnologia da Informação pela UFPE, Pós-graduado em Desenvolvimento Gerencial - Gestão por competência pela Faculdade Santa Maria, Pós-graduado em Metodologias Ativas da Educação e Mestre em Engenharia de Software pelo C.E.S.A.R.EDU. Atualmente é professor do Grupo SER Educacional, é Analista de Sistemas da Empresa Municipal de Informática da Prefeitura de Recife (EMPREL) e Gerente do Setor de Arrecadação da Prefeitura da Cidade de Recife. Trabalhou no Laboratório de Inovações Acadêmicas do Grupo Ser Educacional.

20 anos de experiência em manipulação de Banco de dados

https://lattes.cnpq.br/7221002795193400

adilson.silva@sereducacional.com

@adilsondasilva.professor

Ementa da Disciplina

UNINASSAU
Campus Graças

Algoritmos de Ordenação. Recursividade. Estrutura de dados básicas: matrizes, listas encadeadas, pilhas, filas e árvores. Complexidade dos algoritmos.

Conteúdo Programático

UNINASSAU
Campus Graças

UNIDADE I

- CONCEITOS DE ESTRUTURA DE DADOS
- CONCEITOS AVANÇADOS DE PROGRAMAÇÃO - RECURSÃO; PONTEIROS; ALOCAÇÃO DINÂMICA;
- FUNDAMENTAÇÃO MATEMÁTICA; INDUÇÃO MATEMÁTICA; PRINCÍPIOS DE ANÁLISE DE ALGORITMOS; IMPLEMENTAÇÃO E ANÁLISE EMPÍRICA; CRESCIMENTO DE FUNÇÕES; NOTAÇÃO O
- FUNÇÕES RECURSIVAS
- ESTRUTURAS DE DADOS – MATRIZES E LISTAS ENCADEADAS.

UNIDADE II

- LISTAS SIMPLESMENTE ENCADEADAS
- LISTAS DUPLAMENTE ENCADEADAS,
- LISTAS CIRCULARES E MULTILISTAS.

UNIDADE III

- LISTAS LINEARES RESTRITAS: PILHAS, FILAS.
- ARVORES BINÁRIAS
- ARVORES BINÁRIAS DE BUSCA,
- ARVORES BALANCEADAS TIPO AVL

UNIDADE IV

- BUSCA BINÁRIA
- CLASSIFICAÇÃO INTERNA DE DADOS - CLASSIFICAÇÃO POR SELEÇÃO; MÉTODO DA
- BOLHA; CLASSIFICAÇÃO POR INSERÇÃO,

UNINASSAU UNAMA UNG UNINORTE UNESC UNIFAEL UNIZ
Grupo Ser Educacional

Referências Sugeridas (Física e Digital)

UNINASSAU
Campus Graças

Livros

1."Algoritmos e Estruturas de Dados" por Nivio Ziviani Este é um livro clássico que aborda de maneira abrangente algoritmos e estruturas de dados, apresentando conceitos teóricos e exemplos práticos.

2."Estruturas de Dados e Algoritmos em Java" por Loiane Groner Este livro explora estruturas de dados e algoritmos usando a linguagem de programação Java. Ele inclui exemplos e exercícios práticos.

Cursos On-line

1.Estruturas de Dados – Boson Treinamentos - https://www.youtube.com/watch?v=QRPbHdm05dk
2.Udemy: "Estruturas de Dados e Algoritmos" Um curso que abrange conceitos fundamentais de estruturas de dados e algoritmos, com exemplos práticos em várias linguagens.

Sobre Avaliações

UNINASSAU
Campus Graças

1ª Avaliação – Conteúdo ministrado.

(Postagem das notas em até 5 dias úteis)

2ª Avaliação – Todo conteúdo do semestre

(Postagem das notas em até 5 dias úteis)

2ª Chamada – Todo conteúdo do semestre

(Postagem das notas em até 48 horas)

Final – Todo conteúdo do semestre

(Postagem das notas em até 48 horas)

Sobre Avaliações

No decorrer de cada período letivo são desenvolvidas 02 (duas) avaliações por disciplina, para efeito do cálculo da média parcial. A média parcial é calculada pela média aritmética das duas avaliações efetuadas. O aluno que alcançar a média parcial maior ou igual a 7,0 (sete) é considerado aprovado.

O aluno que não alcançar a média parcial faz em exame final onde precisa alcançar média final maior ou igual a 5,0. São aplicadas avaliações dos tipos: provas teóricas, provas práticas, seminários, trabalhos individuais ou em grupo e outras atividades em classe e extraclasse. O exame final é, obrigatoriamente, prova escrita.

1ª Avaliação: (7,0) + (3,0) OU (8,0) + (2,0) OU 10,0.
2ª Avaliação: 10,0

Obs: Caso proponha alguma atividade junto a Coordenação, especificar qual o tipo de atividade e critérios que irá atribuir a pontuação.

(Exemplos: Seminário, resenha, resumo, fichamento bibliográfico, artigo, pôster científico, relatório, enquete, pesquisa de campo com relatório pós pesquisa, mesa redonda, visitas técnicas, teatro, etc.)

Objetivos

UNINASSAU
Campus Graças

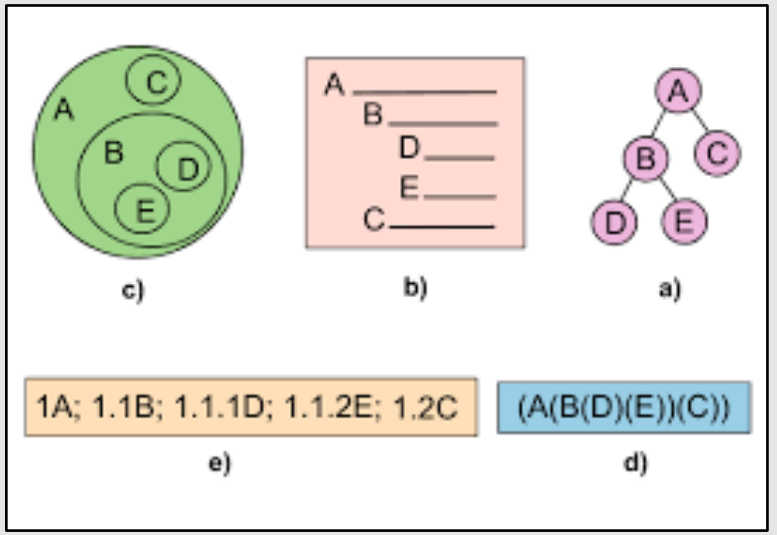
- Revisão de conceitos básicos de programação Java
  - Array e Listas
    - Arrays em Java
      - Declaração, inicalização, acesso e modificação
    - Listas: vantagens sobre arrays, ArrayList em Java.
    - Operações básicas em listas, como adição, remoção e busca.

Variáveis e tipos de Dados – em Java

UNINASSAU
Campus Graças

- Em Java, você declara variáveis para armazenar dados. Os tipos de dados básicos incluem inteiros (int), números de ponto flutuante (float e double), caracteres (char) e booleanos (boolean).
- **Exemplo:**

```
int idade = 30;
double preco = 19.99;
char genero = 'F';
boolean estaChovendo = false;
```



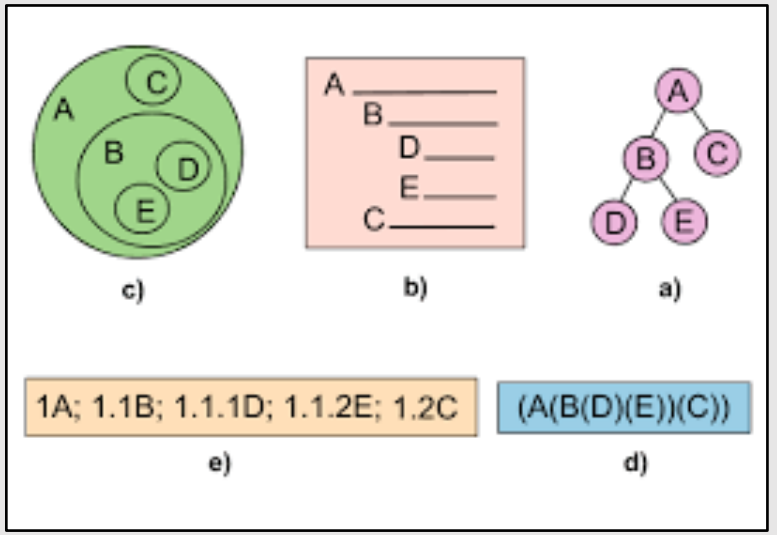
Estrutura de Controle

UNINASSAU
Campus Graças

- Java oferece estruturas de controle para tomar decisões (if-else) e repetir ações (loops como for e while).
- **Exemplo:**

```
if (idade >= 18) {
    System.out.println("Você é maior de idade.");
} else {
    System.out.println("Você é menor de idade.");
}

for (int i = 0; i < 5; i++) {
    System.out.println("Contagem: " + i);
}
```



Funções (Métodos)

UNINASSAU
Campus Graças

- Em Java, você define funções para agrupar um conjunto de instruções em um bloco reutilizável.
- Exemplo:

```
Public static int somar(int a, int b) {
    return a + b;
}
```

Classes e Objetos

UNINASSAU
Campus Graças

- Java é uma linguagem orientada a objetos. Você define classes para criar objetos, que são instâncias dessas classes.
- Exemplo:
  - **class Pessoa {**
  - **String nome;**
  - **int idade;**
  - **}**

  - **Pessoa pessoa1 = new Pessoa();**
  - **pessoa1.nome = "Alice";**
  - **pessoa1.idade = 25;**

Arrays

UNINASSAU
Campus Graças

Arrays são estruturas que permitem armazenar múltiplos valores do mesmo tipo em uma única variável.

```
int[] numeros = {1, 2, 3, 4, 5};
```

Entrada e Saída de Dados

UNINASSAU
Campus Graças

- Você pode usar a classe Scanner para receber entrada do usuário e a classe System.out.println para exibir saída.

- Exemplo:

```
import java.util.Scanner;


Scanner scanner = new Scanner(System.in);
System.out.print("Digite seu nome: ");
String nome = scanner.nextLine();
System.out.println("Olá, " + nome + "!");
```

Manipulação de Strings

UNINASSAU
Campus Graças

- Java oferece muitas funções para manipular strings, como concatenação, busca, substituição, entre outras.

- **Exemplo:**

```
String frase = "Isso é uma string.";
int tamanho = frase.length();
String maiusculas = frase.toUpperCase();
```

Tratamento de Exceções

UNINASSAU
Campus Graças

- Java permite tratar erros com blocos try-catch, evitando que o programa seja interrompido por exceções não tratadas.
- Exemplo:

```
try {
    int resultado = 10 / 0;
} catch (ArithmeticException e) {
    System.out.println("Erro: Divisão por zero!");
}
```

Arrays em Java

UNINASSAU
Campus Graças

- Os arrays em Java são estruturas de dados que permitem armazenar um conjunto de valores do mesmo tipo em uma única variável. Aqui está uma explicação de como trabalhar com arrays em Java, incluindo declaração, inicialização, acesso e modificação:

Declaração de Arrays em Java

UNINASSAU
Campus Graças

- Para declarar um array em Java, você especifica o tipo de dados que ele armazenará, seguido pelo nome do array e colchetes [].
- Você pode declarar um array de qualquer tipo de dados, como inteiros, números de ponto flutuante, caracteres ou objetos.

```
// Declaração de um array de inteiros
int[] numeros;

// Declaração de um array de Strings
String[] nomes;
```

Inicialização de Arrays

UNINASSAU
Campus Graças

- Depois de declarar um array, você precisa inicializá-lo antes de usá-lo. Existem várias maneiras de inicializar um array em Java:

- Inicialização direta: Você pode inicializar um array diretamente com seus valores usando chaves {}.

  - **int[] numeros = {1, 2, 3, 4, 5};**

Inicialização de Arrays

UNINASSAU
Campus Graças

- Inicialização com o operador "new": Você pode usar o operador new para criar um array e, em seguida, atribuir valores a ele.

```
int[] numeros = new int[5]; // Cria um array de inteiros com 5 elementos
numeros[0] = 1;
numeros[1] = 2;
numeros[2] = 3;
numeros[3] = 4;
numeros[4] = 5;
```

Acesso a Elementos de Arrays

UNINASSAU
Campus Graças

- Você acessa elementos de um array usando o nome do array seguido por um índice entre colchetes []. Os índices começam em 0 para o primeiro elemento.

```
int[] numeros = {1, 2, 3, 4, 5};

int primeiroElemento = numeros[0]; // Acessando o primeiro elemento (índice 0)

int terceiroElemento = numeros[2]; // Acessando o terceiro elemento (índice 2)
```

Modificação de Elementos de Arrays

UNINASSAU
Campus Graças

- Você pode modificar os elementos de um array atribuindo novos valores a eles usando o operador de atribuição =.

```
int[] numeros = {1, 2, 3, 4, 5};
numeros[1] = 10; // Modificando o segundo elemento (índice 1) para 10
numeros[3] = 42; // Modificando o quarto elemento (índice 3) para 42
```

Arrays em Java

UNINASSAU
Campus Graças

- É importante lembrar que, em Java, os arrays têm um tamanho fixo após a inicialização. Portanto, você não pode adicionar ou remover elementos diretamente de um array após a inicialização. Se você precisar de uma coleção flexível de elementos, pode considerar o uso de uma estrutura de dados como ArrayList, que é parte da biblioteca padrão de Java.

ArraysList em Java

UNINASSAU
Campus Graças

- As listas são estruturas de dados que, assim como os arrays, permitem armazenar coleções de elementos. No entanto, elas têm algumas vantagens em relação aos arrays tradicionais. Vamos explorar essas vantagens e falar especificamente sobre ArrayList em Java.

Vantagens das Listas sobre Arrays

UNINASSAU
Campus Graças

- **Tamanho Dinâmico:** Uma das principais vantagens das listas é que elas têm um tamanho dinâmico. Isso significa que você pode adicionar ou remover elementos facilmente sem se preocupar com o tamanho máximo do array. Em contraste, os arrays têm um tamanho fixo após a inicialização.

- **Facilidade de Uso:** As listas em Java oferecem métodos convenientes para adicionar, remover e pesquisar elementos, tornando o código mais legível e menos propenso a erros do que a manipulação manual de arrays.

Vantagens das Listas sobre Arrays

UNINASSAU
Campus Graças

- **Redimensionamento Automático:** As listas podem redimensionar-se automaticamente conforme você adiciona ou remove elementos. Isso elimina a necessidade de gerenciar manualmente o tamanho da estrutura de dados.

- **Eficiência de Memória:** As listas dinâmicas, como o ArrayList, são eficientes em termos de memória porque alocam apenas a quantidade de memória necessária para os elementos que contêm.

ArrayList em Java

UNINASSAU
Campus Graças

- ArrayList é uma classe da biblioteca padrão de Java que implementa uma lista dinâmica, permitindo o armazenamento de uma coleção de objetos.

Características ArrayList em Java

UNINASSAU
Campus Graças

- **Tamanho Dinâmico:** Um ArrayList pode crescer ou diminuir dinamicamente à medida que você adiciona ou remove elementos.

- **Tipos Genéricos:** Você pode criar um ArrayList para armazenar objetos de um tipo específico usando a sintaxe de tipos genéricos. Por exemplo, ArrayList<String> cria uma lista que armazena strings.

- **Métodos Úteis:** O ArrayList fornece uma variedade de métodos úteis para adicionar elementos (add()), remover elementos (remove()), acessar elementos (get()), verificar o tamanho (size()), entre outros.

```
import java.util.ArrayList;

public class ExemploArrayList {
    public static void main(String[] args) {
        // Criando um ArrayList de strings
        ArrayList<String> nomes = new ArrayList<>();

        // Adicionando elementos
        nomes.add("Alice");
        nomes.add("Bob");
        nomes.add("Charlie");

        // Acessando elementos
        String primeiroNome = nomes.get(0);
        System.out.println("Primeiro nome: " + primeiroNome);

        // Removendo um elemento
        nomes.remove(1);
        System.out.println("Lista após a remoção: " + nomes);

        // Tamanho da lista
        int tamanho = nomes.size();
        System.out.println("Tamanho da lista: " + tamanho);
    }
}
```

UNINASSAU
Campus Graças

Operações Básicas em listas

UNINASSAU
Campus Graças

- Em Java, você pode realizar várias operações básicas em listas, como adição, remoção e busca, usando a classe ArrayList da biblioteca padrão. Aqui estão exemplos de como realizar essas operações:

Adição de elementos

- Para adicionar elementos a uma lista em Java, você pode usar o método add(). Ele permite adicionar elementos ao final da lista.

```
import java.util.ArrayList;
public class ExemploLista {
    public static void main(String[] args) {
        // Criando uma lista de números inteiros
        ArrayList<Integer> numeros = new ArrayList<>();
        // Adicionando elementos
        numeros.add(10);
        numeros.add(20);
        numeros.add(30);
        // Imprimindo a lista
        System.out.println("Lista de números: " + numeros);
    }
}
```

UNINASSAU
Campus Graças

Remoção de elementos

UNINASSAU
Campus Graças

- Para remover elementos de uma lista, você pode usar o método remove(). Pode ser especificado o índice do elemento que você deseja remover ou o próprio objeto.

```
import java.util.ArrayList;
public class ExemploLista {
    public static void main(String[] args) {
        // Criando uma lista de nomes
        ArrayList<String> nomes = new ArrayList<>();
        // Adicionando elementos
        nomes.add("Alice");
        nomes.add("Bob");
        nomes.add("Charlie");
        // Removendo o elemento "Bob" pelo valor
        nomes.remove("Bob");
        // Removendo o elemento na posição 0 (Alice)
        nomes.remove(0);
        // Imprimindo a lista após a remoção
        System.out.println("Lista de nomes após a remoção: " + nomes);
    }
}
```

Busca de elementos

- Para buscar elementos em uma lista, você pode usar o método indexOf() para encontrar a posição de um elemento ou o método contains() para verificar se um elemento está presente na lista.

```
import java.util.ArrayList;
public class ExemploLista {
    public static void main(String[] args) {
        // Criando uma lista de frutas
        ArrayList<String> frutas = new ArrayList<>();
        // Adicionando elementos
        frutas.add("Maçã");
        frutas.add("Banana");
        frutas.add("Pera");
        // Buscando a posição da fruta "Banana"
        int posicaoBanana = frutas.indexOf("Banana");
        if (posicaoBanana != -1) {
            System.out.println("Banana encontrada na posição " + posicaoBanana);
        } else {
            System.out.println("Banana não encontrada na lista.");
        }
        // Verificando se a lista contém a fruta "Uva"
        boolean contemUva = frutas.contains("Uva");
        if (contemUva) {
            System.out.println("A lista contém Uva.");
        } else {
            System.out.println("A lista não contém Uva.");
        }
    }
}
```

No Eclipse

UNINASSAU
Campus Graças

1. Abra o Eclipse IDE.
2. Crie um novo projeto Java:
   - Vá em "File" (Arquivo) -> "New" (Novo) -> "Java Project" (Projeto Java).
   - Dê um nome ao projeto (por exemplo, "DigitarVetor").
   - Clique em "Finish" (Concluir) para criar o projeto.

No Eclipse

UNINASSAU
Campus Graças

3.Dentro do projeto, crie uma nova classe Java:

- Clique com o botão direito do mouse na pasta "src" (código-fonte) do projeto.
- Vá em "New" (Novo) -> "Class" (Classe).
- Dê um nome à classe (por exemplo, "Main").
- Marque a opção "public static void main(String[] args)".
- Clique em "Finish" (Concluir) para criar a classe.

```
import java.util.Scanner;
public class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        // Solicita ao usuário o tamanho do vetor
        System.out.print("Digite o tamanho do vetor: ");
        int tamanho = scanner.nextInt();
        // Cria um vetor com o tamanho especificado
        int[] vetor = new int[tamanho];
        // Solicita ao usuário que digite os elementos do vetor
        System.out.println("Digite os elementos do vetor:");
        for (int i = 0; i < tamanho; i++) {
            System.out.print("Elemento " + (i + 1) + ": ");
            vetor[i] = scanner.nextInt();
        }
        // Exibe o vetor digitado pelo usuário
        System.out.println("Vetor digitado:");
        for (int i = 0; i < tamanho; i++) {
            System.out.print(vetor[i] + " ");
        }
        // Fecha o scanner
        scanner.close();
    }
}
```

No Eclipse

UNINASSAU
Campus Graças

5.Para executar o programa:
   - Clique com o botão direito na classe "Main" no Package Explorer.
   - Escolha "Run As" (Executar Como) -> "Java Application" (Aplicativo Java).
6.O programa será executado no console do Eclipse. Siga as instruções na tela para digitar o tamanho do vetor e seus elementos.

UNINASSAU
Campus Graças

Obrigado

E- mail:adilson.silva@sereducacional.com

@a dilson da silva .professor