Curso Superior de Tecnologia em Gestão da Tecnologia da Informação

Disciplina:
Introdução a Programação

Apresentação:
Professor / Alunos

Curso Superior de Tecnologia em Gestão da Tecnologia da Informação

- **PLANO DE ENSINO**
  - Apresentação
  - Explanação

Computadores e suas aplicações!

- Computador:
  - Conceito;
  - Aplicações; e
  - Modelos.
- As gerações dos computadores

Primeira Geração (1946-1954)
A primeira geração dos computadores é marcada pela utilização de **válvulas**. A válvula é um tubo de vidro, similar a uma lâmpada fechada sem ar em seu interior, ou seja, um ambiente fechado a vácuo, e contendo eletrodos, cuja finalidade é controlar o fluxo de elétrons. As válvulas aqueciam bastante e costumavam queimar com facilidade.

As gerações dos computadores

Primeira Geração (1946-1954)

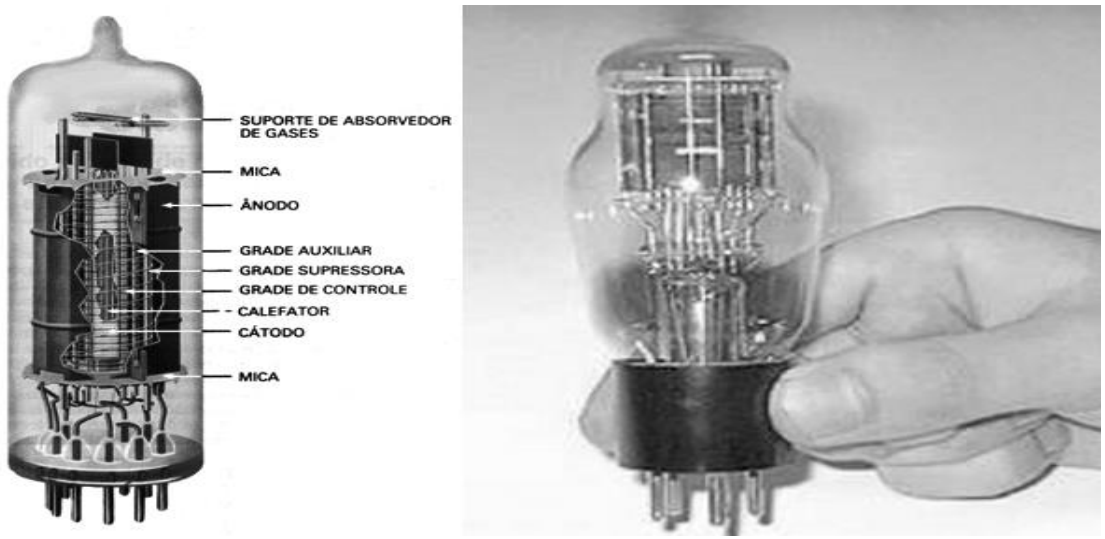


Figura 1. As válvulas eram do tamanho de uma lâmpada.
Fonte: http://producao.virtual.ufpb.br/books/camyle/introducao-a-computacao-livro/livro/livro.chunked/ch01s02.html

As gerações dos computadores

Primeira Geração (1946-1954)

Figura 2. ENIAC, representante da primeira geração dos computadores..
Fonte: http://producao.virtual.ufpb.br/books/camyle/introducao-a-computacao-livro/livro/livro.chunked/ch01s02.html

As gerações dos computadores

Segunda Geração (1955-1964)

A segunda geração de computadores foi marcada pela substituição da válvula pelo **transistor**.

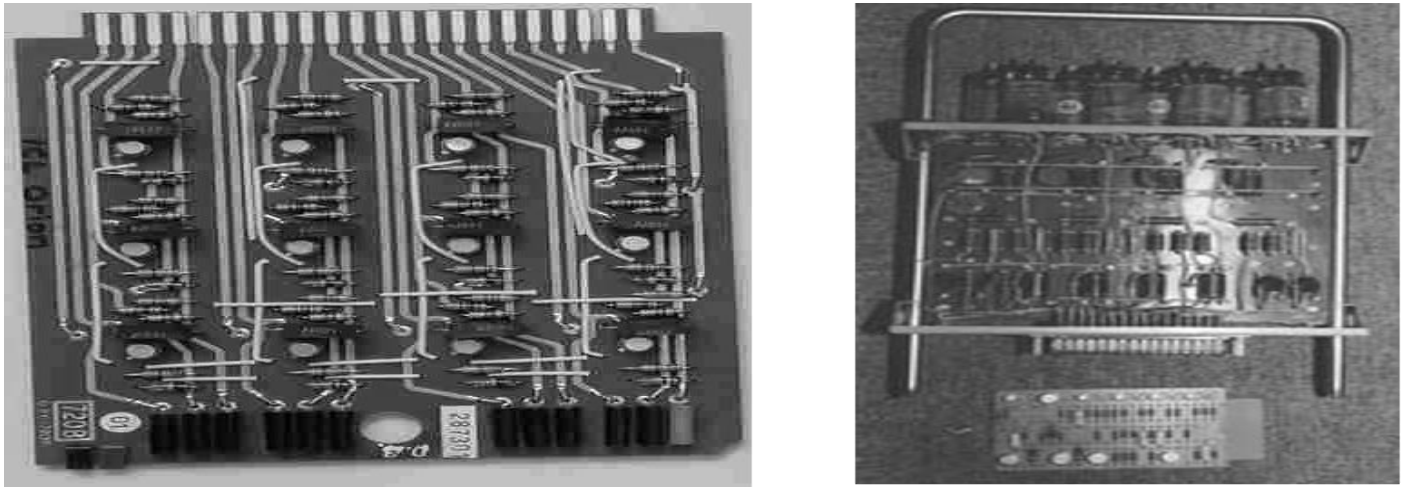


Figura 3. Circuito com vários transistores (esquerda). Comparação do circuito com válvulas (canto superior-direito) com um circuito composto de transistores (inferior-direito).
Fonte: http://producao.virtual.ufpb.br/books/camyle/introducao-a-computacao-livro/livro/livro.chunked/ch01s02.html

As gerações dos computadores

Segunda Geração (1955-1964)

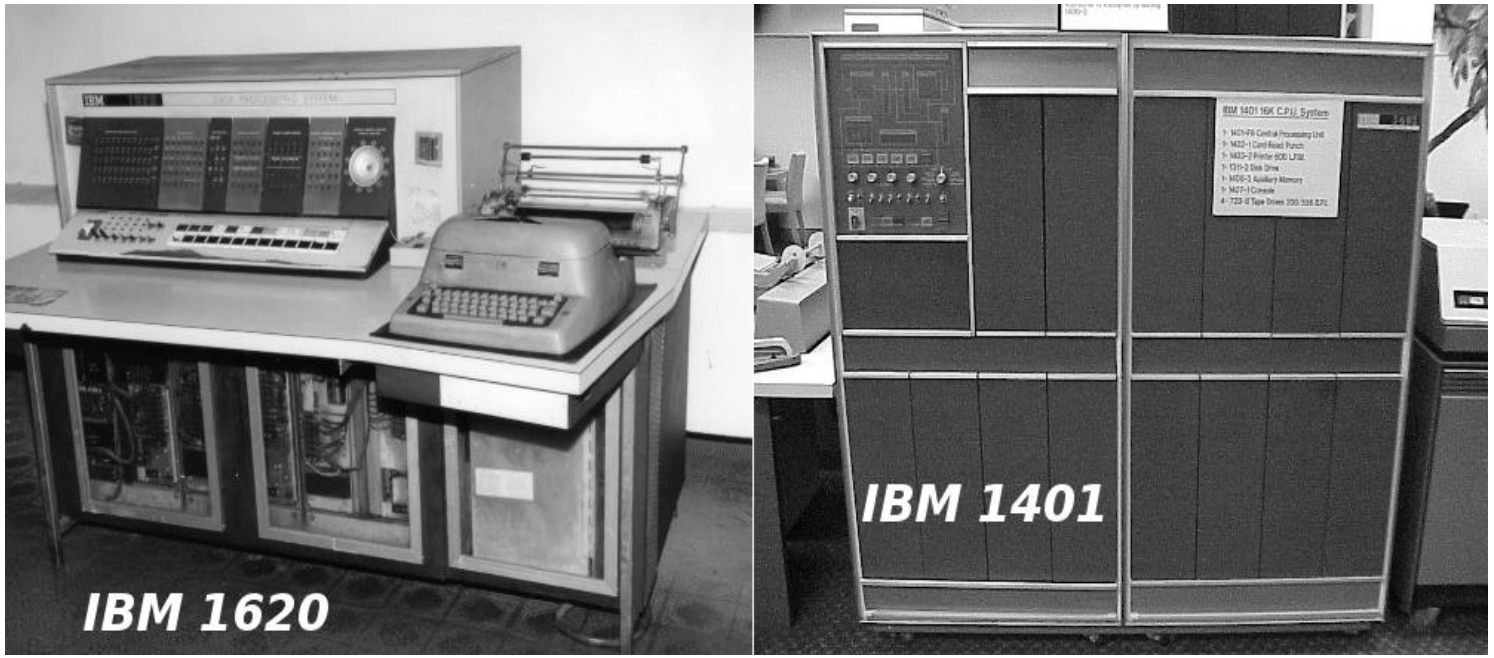


Figura 4. Computadores IBM da segunda geração. Fonte: http://producao.virtual.ufpb.br/books/camyle/introducao-a-computacao-livro/livro/livro.chunked/ch01s02.html

As gerações dos computadores

Terceira Geração (1964-1977)

A terceira geração de computadores é marcada pela utilização dos **circuitos integrados**, feitos de silício. Também conhecidos como **microchips**, eles eram construídos integrando um grande número de transistores, o que possibilitou a construção de equipamentos menores e mais baratos.

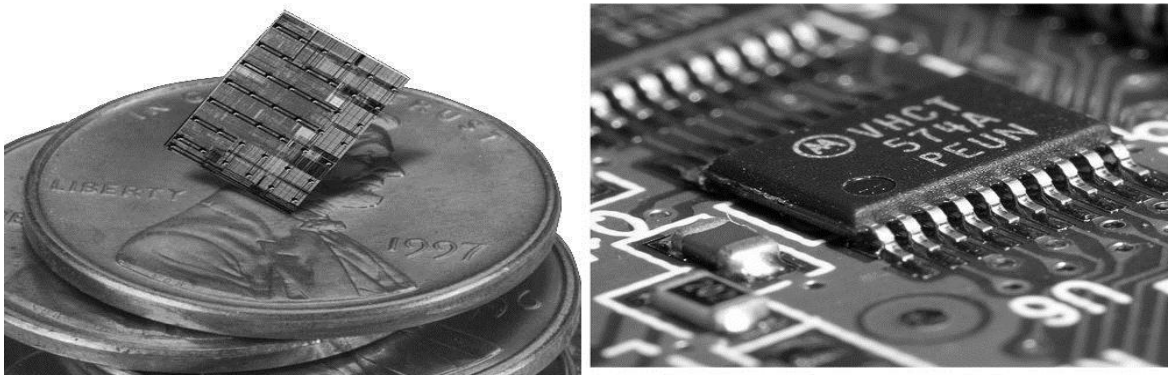


Figura 5. Comparação do tamanho do circuito integrado com uma moeda (esquerda) e um chip (direita).
Fonte: http://producao.virtual.ufpb.br/books/camyle/introducao-a-computacao-livro/livro/livro.chunked/ch01s02.html

As gerações dos computadores

Terceira Geração (1964-1977)

Figura 6. Computador Apple I.
Fonte: http://producao.virtual.ufpb.br/books/camyle/introducao-a-computacao-livro/livro/livro.chunked/ch01s02.html

As gerações dos computadores

Quarta Geração (1977-1991)

Os computadores da quarta geração são reconhecidos pelo surgimento dos processadores — unidade central de processamento. Os sistemas operacionais como MS-DOS, UNIX, Apple's Macintosh foram construídos.

Figura 7. Computador pessoal da quarta geração.
Fonte: http://producao.virtual.ufpb.br/books/camyle/introducao-a-computacao-livro/livro/livro.chunked/ch01s02.html

As gerações dos computadores

Quinta Geração (1991 — dias atuais)

Os computadores da quinta geração usam processadores com milhões de transistores. Nesta geração surgiram as arquiteturas de 64 bits, os processadores que utilizam tecnologias RISC e CISC, discos rígidos e diversos dispositivos com grande capacidade de armazenamento.

Figura 8. Computador da quinta geração. Fonte: http://producao.virtual.ufpb.br/books/camyle/introducao-a-computacao-livro/livro/livro.chunked/ch01s02.html

Computadores e suas aplicações!

- Linguagem de programação:
  - Conceitos;
  - Aplicações; e
  - Tipos.

Linguagem de programação

Os computadores chegaram aos diversos níveis das organizações:

Nestes contexto:

• Que linguagem entendem?

• Que produtos podem usar?

Todos os computadores digitais, são sistemas em tudo semelhantes, no que diz respeito ao princípio de funcionamento.

• Neste é possível encontrar três subsistemas:

• hardware, software e peopleware.

Linguagem de programação

Uma máquina precisa de programação para oferecer facilidades:

- Criação de arquivos
- Segurança
- Comunicação
- Resolução de conflitos
- Tradução de linguagens

Linguagem de programação

O que é uma **Linguagem de Programação?**

• um conjunto de termos (vocabulário) e de regras (sintaxe) que permitem a formulação de instruções a serem executadas num computador.

Linguagem de programação

O que é uma **Linguagem de Programação?**

• um conjunto de termos (vocabulário) e de regras (sintaxe) que permitem a formulação de instruções a serem executadas num computador.

O computador só entende uma linguagem conhecida como código binário ou código máquina, consistente em zeros e uns.

Linguagem de programação

As linguagens mais próximas à arquitetura hardware se denominam **linguagens de baixo nível**

As que se encontram mais próximas aos programadores e usuários se denominam **linguagens de alto nível.**

Linguagem de programação

Você conhece alguma linguagem de programação?

Precisamos entender como construir um programa de computador.

Bem vindos a Lógica de programação (Algoritmos).

Linguagem de programação
Evolução

http://migre.me/dHK4w

http://www.naosalvo.com.br/8-reportagens-que-provam-que-a-tecnologia-passa-rapido/?fb_action_ids=525731890782129&fb_action_types=og.likes&fb_source=other_multiline&action_object_map=%7B%2252573189078212 9%22%3A460829943978718%7D&action_type_map=%7B%2252573189078212 9%22%3A%22og.likes%22%7D&action_ref_map=%5B%5D

1. Introdução à Lógica de Programação

1.1 - Noções de lógica

O que é lógica?

- A lógica trata da correção do pensamento. Como filosofia, ela procura saber por que pensamos assim e não de outro jeito.

- Com arte ou técnica, ela nos ensina a usar corretamente as leis do pensamento.

1.1 - Noções de lógica

- Exemplos:

- A gaveta está fechada.
  A chave está na gaveta.
  Preciso primeiro abrir a gaveta, para depois pegar a chave.

- Maria é mais velha que João.
  João é mais velho que Paulo.
  Portanto, Maria é mais velha que Paulo.

- Uso da lógica no dia a dia

1.2 - Algoritmizando a lógica

- O objetivo fundamental da aplicação da lógica voltada para programação de computadores é a construção de algoritmos.

O que é algoritmo?

É uma seqüência de passos que visam atingir um objetivo bem definido.

Apesar de achar este nome estranho algoritmos são comuns em nosso dia a dia, como por exemplo uma receita de bolo. Nela está descrita uma série de ingredientes necessários, uma seqüência de diversos passos – ações – a serem cumpridos para que se consiga fazer determinado tipo de bolo.

1.2 - Algoritmizando a lógica

Ação é um acontecimento que a partir de um estado inicial, após um período de tempo finito, produz um estado final previsível e bem definido em que:

Estado é a situação atual de um determinado objeto.

Portanto podemos redefinir algoritmo como:

Algoritmo é a descrição de um conjunto de ações que, obedecidas, resultam numa sucessão finita de passos, atingindo o objetivo.

1.2 - Algoritmizando a lógica

Em geral um algoritmo destina-se a resolver um problema:

Fixa um padrão de comportamento a ser seguido, uma norma de execução a ser trilhada, com vistas a alcançar, como resultado final, a solução de um problema.

Padrão de comportamento:

Imagine a seguinte seqüência de números: 1,6,11,16,21,26...

Para determinar o sétimo elemento da série, precisamos descobrir qual a sua regra de formação, isto é o seu padrão de comportamento.

Introdução à Lógica de Programação

Fundamentação complementar leitura capítulo 1 Forbellone e Eberspacher. Lógica de programação – A construção de Algoritmos e Estruturas de Dados.

Exercícios:

1. Descreva todos os passos (ações) que você executou desde o momento que saio de casa ou trabalho etc... Até o momento de chegada na faculdade.

   Após 20 minutos socialização do exercício.

Introdução à Lógica de Programação

Exercícios:

2. Um senhor está numa das margens de um rio com uma raposa, uma dúzia de galinhas e um saco de milho. Ele pretende atravessar o rio com suas cargas num barco que só comporta ele e uma de suas cargas.

   Descreva o algoritmo (todas as ações) para que este senhor consiga realizar tal proeza sem perder nenhuma de suas cargas.

   Após 20 minutos socialização do exercício.

1.3 – Tipos Primitivos

Para entendermos os tipos primitivos, voltemos a um conceito muito importante a Informação:

Informação é a matéria-prima que faz com que seja necessária a existência dos computadores, pois eles são capazes de manipular e armazenar um grande volume de dados.

Tipos primitivos é a classificação da Informação de forma que possa ser interpretada por uma máquina (computador).

1.3 – Tipos Primitivos

Inteiro: Toda e qualquer informação numérica que pertença ao conjunto dos números inteiros relativos (negativa, nula ou positiva).

Exemplos: Ele tem 15 irmãos.
A temperatura desta noite será de −2 graus.

Real: Toda e qualquer informação numérica que pertença ao conjunto dos números reais (negativa, nula ou positiva).

Exemplos: Ela tem 1,73 metros de altura.
O saldo bancário dele é R$ 3.500,00.

1.3 – Tipos Primitivos

Caractere: Toda e qualquer informação composta por um conjunto de caracteres alfabéticos (a...z), alfanuméricos (0...9) ou caracteres especiais (@#$%*!~^...).

Exemplos: Constava na prova: "Use somente caneta!".
O parque esta repleto de placas: "Não pise na grama".

Lógico: Toda e qualquer informação que pode apenas assumir duas situações.

Exemplos: A porta pode estar aberta ou fechada.
A lâmpada pode estar acesa ou apagada.

1.3 – Tipos Primitivos

Exercícios:

1. Determine qual o tipo primitivo de informação presente nas sentenças abaixo:

a) A placa "PARE" tinha dois furos de bala.
b) Maria subiu cinco degraus para pegar uma maça boa.
c) João levou uma hora e meia para chegar na faculdade.
d) Paulo pintou em sua camisa: "Preserve o meio ambiente!" , e ficou devendo R$ 15,50 ao vendedor de tintas.
e) Ana recebeu sua 8ª medalha por ter alcançado a marca de de 56,3 segundos nos 100 metros rasos.

Após 10 minutos socialização do exercício.

1.4 – Informações Constantes

Uma informação é constante quando não sofre nenhuma variação no decorrer do tempo.

Obs: Para diferenciar as informações constantes de tipo caractere de outros tipos iremos delimitá-las por aspas (“”).
Em sistemas de informação o tipo lógico sempre será Falso ou Verdadeiro.

Exercício: Identifique na sala pelo menos uma informação constante.

1.5 – Informações Variáveis

Uma informação é variável quando sofre alterações no decorrer do tempo.

Exercício: Identifique na sala pelo menos três informação variáveis.

1.6 – Formação de identificadores

Um identificador é declarado para representar uma determinada informação constante ou variável.

Regra de formação:

1. Devem começar por caractere alfabético.
2. Podem ser seguidos de mais caracteres alfabéticos ou numéricos.
3. Não é permitido o uso de caracteres especiais com exceção do *underline*.

Sintaxe:

<Nome do identificador> : <Tipo primitivo>;

Exemplo: Nome:Caractere;
Idade:Inteiro;

1.6 – Formação de identificadores

Exercícios:

1. Identifique os identificadores válidos:
   a) (x) b) U2 c) AH! d) "Aluno"

   e) #55 f) Km/L g) Nota_Aluno h) B54

2. Supondo que as variáveis NA, NM, NMAT, SX sejam utilizadas para armazenar a nota do aluno, o nome do aluno, o número da matrícula e o sexo, declare-as corretamente.

Após 10 minutos socialização dos exercícios.

1.7 – Operadores Aritméticos

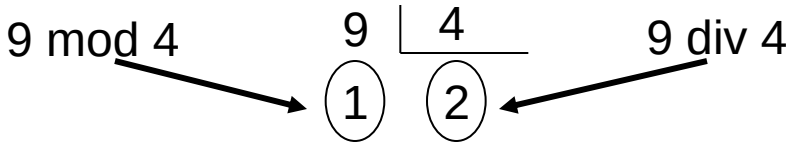
Chamamos de operadores aritméticos o conjunto de símbolos que representa as operações básicas da matemática:

+ (adição)	- (subtração)
* (multiplicação)	/ (divisão)
** (potenciação)	//(radiciação)

Exemplos: 2 + 2, xpto / 5, x**2, ((nota * 0.7) / 2)...

Usaremos outras operações matemáticas não convencionais cujos operadores são: mod (resto da divisão)
div (quociente da divisão inteira)

Exemplo:



1.8 – Funções matemáicas

Além das operações básicas, podemos usar nas expressões aritméticas algumas funções da matemática:

Sem(x)	Seno de x
Cos(x)	Coseno de x
Tg(x)	Tangente de x
Arctg(x)	Arco cujo a tangente é x
Arccos(x)	Arco cujo o seno é x
Arcsen(x)	Arco cujo o seno é x
Int(x)	A parte inteira de um número fracionário
Rnd(x)	Valor randômico a x

1.9 – Operadores relacionais

São utilizados para realizar operações entre dois valores de mesmo tipo primitivo. Os valores podem ser representados por constantes, variáveis ou expressões aritméticas.

Os operadores aritméticos são:

= (igual a)	<> (diferente de)
> (maior que)	>= (maior igual a)
< (menor que)	<= (menor igual a)

O resultado obtido de uma relação é sempre um valor lógico.

1.10 – Operadores lógicos

São utilizados para a formação de novas proposições a partir de outras já conhecidas.

Os operadores lógicos são:

Símbolo	Função
e	conjunção
ou	Disjunção (não exclusiva)
xou	Disjunção (exclusiva)
não	negação

1.10.1 – Tabela verdade

É conjunto de todas as possibilidades combinatórias entre os valores de diversas variáveis lógicas, as quais se encontram em apenas duas situações e um conjunto de operadores lógicos.

A	B	A e B
F	F	F
F	V	F
V	F	F
V	V	V

A	B	A ou B
F	F	F
F	V	V
V	F	V
V	V	V

A	B	A xou B
F	F	F
F	V	V
V	F	V
V	V	F

A	não A
F	V
V	F

Exemplo: Se chover e relampejar, Maria fica com medo.
Quando Maria fica com medo?

1.11 – Comando de atribuição

Permite fornecer um valor a uma certa variável, onde o tipo dessa informação deve ser compatível com o tipo da variável.

O comando de atribuição possui a seguinte sintaxe:

<Nome do identificador> := <Valor a ser atribuído>;

Exemplo: Nome := "Maria";
Idade := 40;

Obs.: O valor a ser atribuído pode ser: Uma informação vareável ou constante, uma expressão aritmética ou lógica.

1.12 – Comandos de entrada e saída

Fazendo uma analogia ao nosso dia a dia como por exemplo a respiração:

Inspiramos o ar que compõe a atmosfera, realizando dessa forma uma ***entrada***, agora o ar é ***processado*** pelo organismo e liberado na seqüência que é a ***saída***.

Da mesma forma funciona o computador, só que no lugar do ar o processamento se dá com informações.

1.12 – Comandos de entrada

Comando utilizado para carregar as informações.

Sintaxe:

<Comando de leitura> <(Variável)>;

Exemplo: Leia(Nome);
Leia(idade);

Obs: A leitura é realizada somente para informações vaiáveis.

1.13 – Comandos de saída

Comando utilizado para mostrar as informações (resultados).

Sintaxe:

<Comando de Saída> <(Variável, Constante, expressão)>;

Exemplo: Escreva("Nome do aluno: " , Nome);
Escreva("Sua idade é: ", idade, "Média = ", (n1 + n2) / 2);

Obs: A saída é realizada para informações vaiáveis e constantes.

1.14 – Blocos

- Um bloco é um conjunto de ações com uma função definida.
- Um algoritmo pode ser visto como um bloco.
- É utilizado para definir os limites nas quais as variáveis declaradas em seu interior são conhecidas.

Sintaxe: Para delimitar um bloco, utilizamos os delimitadores: Inicio e Fim.

<Inicio> <Ações> < Fim>;

Exemplo: Inicio
Escreva("Digite o nome do Aluno: ");
Leia(Nome);
fim.

1.15 – Metodologia para resolução de problemas

- Algoritmos:
- Seqüência de passos para a solução de um problema.
- Método *top-down* ou refinamento sucessivo.
- Fases para a solução de um problema independente da área:
  - Leitura e interpretação
  - Resolução do problema
  - Apresentação do resultado ou da solução

1.15 – Metodologia para resolução de problemas

- Proposta de etapas para a solução de um algoritmo, apresenta 9 passos:

1. Ler o problema até entender;
2. Identificar o que está sendo fornecido (entrada);
3. Identificar o que está sendo pedido (saída);
4. Criar uma base de dados com possíveis resultados;
5. Descobrir as operações necessárias para transformar as entradas em saídas (processamento);

1.15 – Metodologia para resolução de problemas

6. Descrever as operações necessárias para transformar as entradas em saídas (algoritmo);
7. Executar o que foi descrito no passo 6 teste-de-mesa;
8. Comparar os resultados obtidos com os esperados passo 4. Voltando ao passo 1 caso não sejam satisfeitos;
9. Revisar a partir do passo 5 buscando uma otimização

1.15 – Metodologia para resolução de problemas
Exemplo

COMO PROPOSTO

COMO ESPECIFICADO

COMO PROJETADO

COMO IMPLEMENTADO

COMO INSTALADO

O QUE O USUÁRIO QUERIA

1.15 – Metodologia para resolução de problemas Aplicação

- Com base na metodologia proposta, vamos aplicar os passos no seguinte problema:

- Elaborar um algoritmo que entre com o valor de duas notas e apresente como resultado a média das notas informadas.

1.15 – Metodologia para resolução de problemas Aplicação

- Passo 1 – Trata-se de um problema de cálculo de média;
- Passo 2 – O valora das duas notas;
- Passo 3 – O resultado é a média das notas;
- Passo 4 – Médias: 8, 9... Ou seja valores de 0 a 10;
- Passo 5 – Soma e divisão, somar as duas notas e dividir por dois;

1.15 – Metodologia para resolução de problemas Aplicação

- Passo 6:
  - Entrar com a primeira nota;
  - Entrar com a segunda nota;
  - Calcular a média;
  - Mostrar o resultado;
- Passo 7:

Primeira nota	Segunda nota	Cálculo	Resultado
8	9	((8 + 9) / 2)	8,5

1.15 – Metodologia para resolução de problemas Aplicação

- Passo 8 – O resultado obtido atende o passo 4 solução ok;
- Passo 9 – Não há necessidade de otimização a solução do problema proposto está ok.

1.15 – Metodologia para resolução de problemas Exercícios

- Faça um algoritmo que entre com o ano de nascimento de uma pessoa e mostre como resultado a sua idade.

- Faça um algoritmo que entre com um valor qualquer e mostre como resultado o valor acrescido de mais 10%.

1.15 – Metodologia para resolução de problemas Exercícios

- O valor de determinado artigo adquirido em uma loja, pode ser valor a vista, com um desconto de 8%, ou em 3 vezes acrescido de 15% no valor. Faça um algoritmo que tendo como entrada o valor do artigo, mostre como resultado o valor a vista e o valor das parcelas a prazo.

2 – Estruturas de Seleção

- Uma estrutura de seleção permite a escolha de um grupo de ações e estruturas a ser executado quando determinadas condições , representadas por expressões lógicas, são ou não satisfeitas.

- Seleção Simples:

Sintaxe: Se (<Condição>) entao
inicio
<ações>;
fim;

Seleção Simples

- <Condição> é uma expressão lógica que quando inspecionada, pode gerar um resultado falso ou verdadeiro.

  Se a condição for verdadeira o bloco de código é executado, caso contrário encerra o comando.

- Seleção Simples:

Sintaxe:

```
Se (<Condição>) entao
    inicio
        <ações>;
    fim;
```

Seleção Composta

Sintaxe:

```
Se (<Condição>) entao
    inicio
        <ações>;
    fim
  Se não
        inicio
            <ações>;
        fim;
```

Seleção Encadeada

Sintaxe:

```
Se (<Condição1>) entao
   inicio
        <ações>;
   fim
  Se não
        Se (<Condição2>) entao
           inicio
                <ações> ;
            fim
        Se não
              Se (<Condição3>) entao
                 inicio
                      <ações>;
                fim
              Se não  <ação B>;
```

Seleção Composta

Elabore um algoritmo que permita entrar com o conceito final da disciplina de lógica de programação. O conceito pode ser:

- [OT] – Ótimo
- [B] – Bom
- [S] – Satisfatório
- [I] – Insatisfatório

Emitir a mensagem “Aluno aprovado!” se o conceito for OT, B ou S e a mensagem “Aluno Reprovado!” caso o conceito seja I.

Seleção Encadeada

Elabore um algoritmo que entre com a idade de um nadador, mostre como resultado a categoria que ele pertence de acordo com a tabela a seguir:

Infantil A – 5 a 7 anos

Infantil B – 8 a 10 anos

Juvenil A – 11 a 13 anos

Juvenil B – 14 a 17 anos

Sênior maiores de 18 anos.

Seleção Encadeada

Construa um algoritmo que tendo como dados de entrada o preço de um produto e um código de origem, emita como resultado o preço seguido de sua procedência. Conforme tabela abaixo:

Código de origem:

1 – Sul

2 – Norte

3 – leste

4 – oeste

5 ou 6 – nordeste

7, 8 ou 9 – sudeste

10 até 20 – cetro-oeste

25 até 50 – noroeste

Produtos que não tem o código de origem na tabela acima serão considerados importados.

Seleção Encadeada

Elabore um programa que determine o grau de obesidade de uma pessoa, sendo fornecido o seu peso e a sua altura. O grau de obesidade é determinado pelo índice da massa corpórea

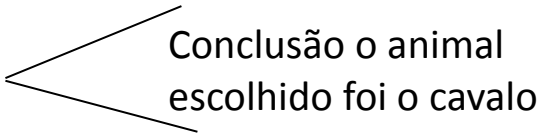
(Massa = Peso / (Altura * Altura)) conforme tabela abaixo:

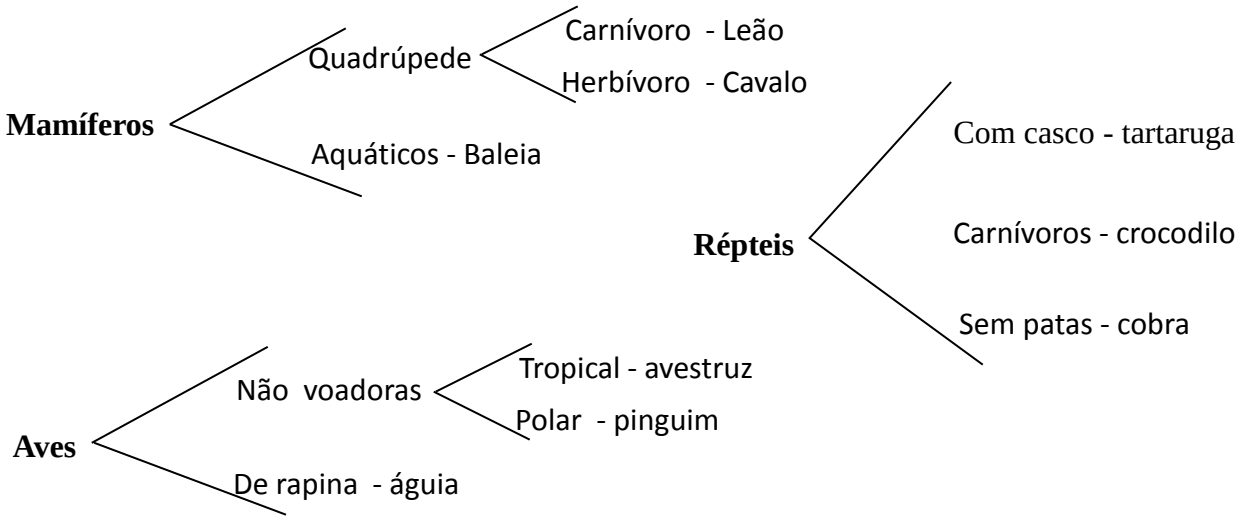
Massa Corpórea	Grau de Obesidade
< 26	Normal
>= 26 e < 30	Obeso
< =30	Obeso Mórbido

Seleção Encadeada

•Construa um algoritmo que seja capaz de identificar qual dentre os animais seguintes, foi escolhido, através de perguntas e respostas.
Animais Possíveis: Leão, cavalo, baleia, avestruz, pinguim, águia, tartaruga, crocodilo, cobra.

Exemplo: É mamífero? Sim
É quadrúpede? Sim
É Carnívoro? Não
É herbívoro? Sim





Seleção Encadeada

A empresa XPTO Empreiteira Ltda pretende conceder um aumento de salários aos seus colaboradores. Sabendo que existem 3 faixas de salários e o aumento será concedido em reais conforme tabela abaixo:

Faixa	Valor Salário em R$	Aumento em R$
1	1.000,00	193,56
2	1.001,00 até 2.000,00	174,34
3	Maior que 2.000,00	127,89

Faça um Algoritmo que entre com o salário do colaborador e apresente como resultado o percentual de aumento para faixa salarial correspondente.

Seleção Caso for

Faça um programa que calcule as quatro operações matemáticas elementares. O programa deve solicitar ao usuário que informe dois números reais. Em seguida, deve solicitar que informe a operação a ser realizada, de acordo com as seguintes regras:

- Se usuário informar o caractere '+', então o programa deve somar os dois números;
- Se usuário informar o caractere '-', o programa deve subtrair o segundo número do primeiro.
- Se usuário informar o caractere '/', o programa deve dividir o primeiro número pelo segundo.
- Se usuário informar o caractere '*', o programa deve multiplicar os dois números.

Seleção Composta

- Elabore um algoritmo para validar uma senha de acesso, sabendo que a senha correta é ‘asdfg’. Emitir a mensagem: acesso permitido se a senha estiver correta ou acesso negado caso contrário.

- Construa um algoritmo que verifique se um fornecido pelo usuário é par ou impar.

Repetição com teste no início

- Estrutura de controle de fluxo lógico que permite executar diversas vezes um mesmo trecho do algoritmo, porém, sempre verificando antes de cada execução se é permitido executar o mesmo trecho.

- Para realizar a repetição com teste no início, utilizamos a estrutura *Enquanto*, que permite que um bloco ou ação seja repetida enquanto uma determinada condição for verdadeira.

Repetição com teste no início *Enquanto*

Sintaxe:

```
Enquanto (<condição>) faça
 Início
        <ação 1>;
        <ação 2>;
        <ação n>;
        <validação da condição>;
 fim;
```

Repetição com teste no início *Enquanto*

Onde:

<condição> → pode ser uma informação variável, constante ou uma expressão

<ação → um comando

<validação da condição> → validar a condição que permite a execução do enquanto.

Exercícios com Repetição *Enquanto*

- Construa um algoritmo que, dado um conjunto de valores inteiros e positivos, determine qual o menor e o maior valor do conjunto. O final do conjunto é conhecido através do valor 0 (zero) que não deve ser considerado.
- Elabore um algoritmo que, dado a idade e o sexo de um grupo de pessoas, apresente no final: a média de idade das pessoas; o percentual de pessoas do sexo masculino e feminino; a maior idade masculina e a menor idade feminina. Para finalizar o sexo deverá ser igual a N.

Exercícios com Repetição *Enquanto*

Em uma eleição presidencial, existem 4 candidatos. Os votos são informados através de código. Os dados para a escrutinagem obedecem à seguinte codificação:

- 1,2,3,4 ➔ voto para os respectivos candidatos
- 5 ➔ voto nulo
- 6 ➔ voto em branco.
- 0 ➔ finalizar

Elabore um algoritmo que calcule e escreva:

a) total de votos para cada candidato;

b) total de votos nulos;

c) total de votos em branco;

d) Percentual de votos de cada candidato em relação ao total de votos validos.

Atividade: Aplique todos os 9 passos para a solução do problema proposto.
A empresa XPTO Ltda pretende conceder um aumento de salários aos seus colaboradores. Sabendo que existem 3 faixas de salários e o aumento será concedido em reais conforme tabela abaixo:

Faça um algoritmo que entre com o salário do colaborador e apresente como resultado o percentual de aumento para faixa salarial correspondente.

Obs. Para finalizar informar zero para o salário.

Faixa	Valor Salário em R$	Aumento em R$
1	1.000,00	193,56
2	1.001,00 até 2.000,00	174,34
3	Maior que 2.000,00	127,89

Repetição com variável de controle *Para*

- A estrutura de repetição *Para* repete a execução de uma ação ou bloco de ações um número definido de vezes, pois ela possui limites fixos.

Sintaxe:

```
para V := vi Até vf Faça
inicio
        <ação 1>;
        <ação 2>;
        <ação n>;
fim;
```

Repetição com variável de controle *Para*

Onde:

- v -> V é a variável de controle
- vi -> é o valor inicial de V
- vf -> é o valor final de V, ou seja, o valor até o qual ela pode chgar
- <ação> -> um comando

Repetição com variável de controle *Para*

- Construa um algoritmo que liste os números impares no intervalo de 1 a 100.

- Elabore um algoritmo que calcule o valor de H, sendo que ele é determinado pela série:
  - H = 1/1 + 3/2 + 5/3 + 7/4 + ... + 99/50.

Repetição com variável de controle *Para*

- Construa um algoritmo que liste a tabuada de 8.

- Elabore um algoritmo para calcular a tabuada informada pelo usuário, deve ser informado a tabuada, intervalo inicial e final.

  exemplo: tabuada = 25

  intervalo inicial = 1

  intervalo final = 100

  obs. Para finalizar digitar a tabuada 0(zero).

Exercícios com Repetição *Para*

- A conversão de graus Faherenheit para centígrados é obtida pela formula C = 5/9(F-32). Faça um algoritmo que calcule e escreva uma tabela de graus centígrados em função de graus Faherenheit que variam de 50 a 150 de um em um.

Repetição com variável de controle *Para*

• Elabore um algoritmo que determine o valor de S: Onde:
S = 1/1 – 2/4 + 3/9 ... – 10/100

• Elabore um programa que mostre como resultado a soma dos dez primeiros elementos da seguinte série:
2/500 - 5/450 + 2/400 - 5/350 +...

• Faça um programa que calcule o valor de y, sendo que ele é determinado pela série:
y = 1/1 + 3/2 - 5/3 -7/4 +9/5 + 11/6... 99/50.

Repetição com teste no final

- Estrutura de controle de fluxo lógico que permite executar diversas vezes um mesmo trecho do algoritmo. Verificando no final do bloco se é permitido executar o mesmo trecho.

- Para realizar a repetição com teste no final, utilizamos a estrutura *Repita*, que permite que um bloco ou ação seja repetida até que uma determinada condição seja verdadeira.

Repetição com teste no Final *Repita*

Sintaxe:

```
Repita
        <ação 1>;
        <ação 2>;
        <ação n>;
Ate <condição>;
```

Obs.: A execução das ações acontecerá pelo menos umas vez, isto porque a condição é validada somente no final.

Repetição com teste no Final *Repita*

Escreva um algoritmo para validar uma senha. A senha correta é ‘asdfg’.

Quebra de laço

```
I := 1;
while true do
begin
   Inc(I);
   if I < 10000000 then
     Continue;
   ShowMessage('Chegamos a dez milhões');
   Break;
end;
```

Quebra de laço

Repeat

O laço repeat executa instruções até que uma condição seja verdadeira.

```
I := 1;
repeat
   S := InputBox('Acesso', 'Digite a senha', '');
   Inc(I);
   If I > 3 then
      Halt;
until S = `asdfg';
```

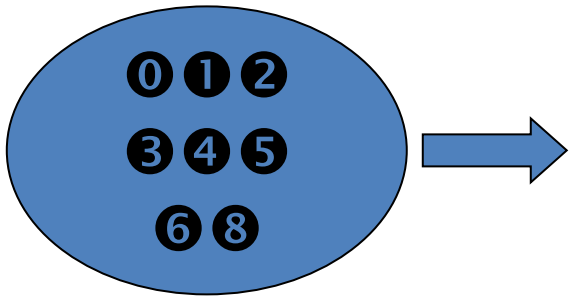
Estrutura de dados

- A quantidade de tipos de informação estipulados (tipos primitivos) não é suficiente para representar toda e qualquer informação que possa surgir.

- Construiremos novos tipos, denominados "tipos construídos" a partir da composição de tipos primitivos. Esses novos tipos tem um formato denominado ESTRURA DE DADOS, que define como os tipos primitivos estão organizados.

Variáveis compostas homogêneas

- Quando uma determinada estrutura de dados for composta de variáveis com o mesmo tipo primitivo, temos um conjunto homogêneo de dados.



Temos um conjunto de dados do tipo primitivo inteiro, ou seja, temos um conjunto homogêneo de dados.

Variáveis compostas unidimensionais

- Quando uma determinada estrutura de dados for composta de uma única dimensão (Vetor).

Dados	Ana	Bruno	Carol	Dani	Edson	Fabio	Gino
Índice	1	2	3	4	5	6	7

Variáveis compostas unidimensionais

- Declaração:

Tipo <identificador> = Vetor[Li..Lf] de <tipo primitivo>;

Onde:

<identificador> -> Representa o nome da estrutura;

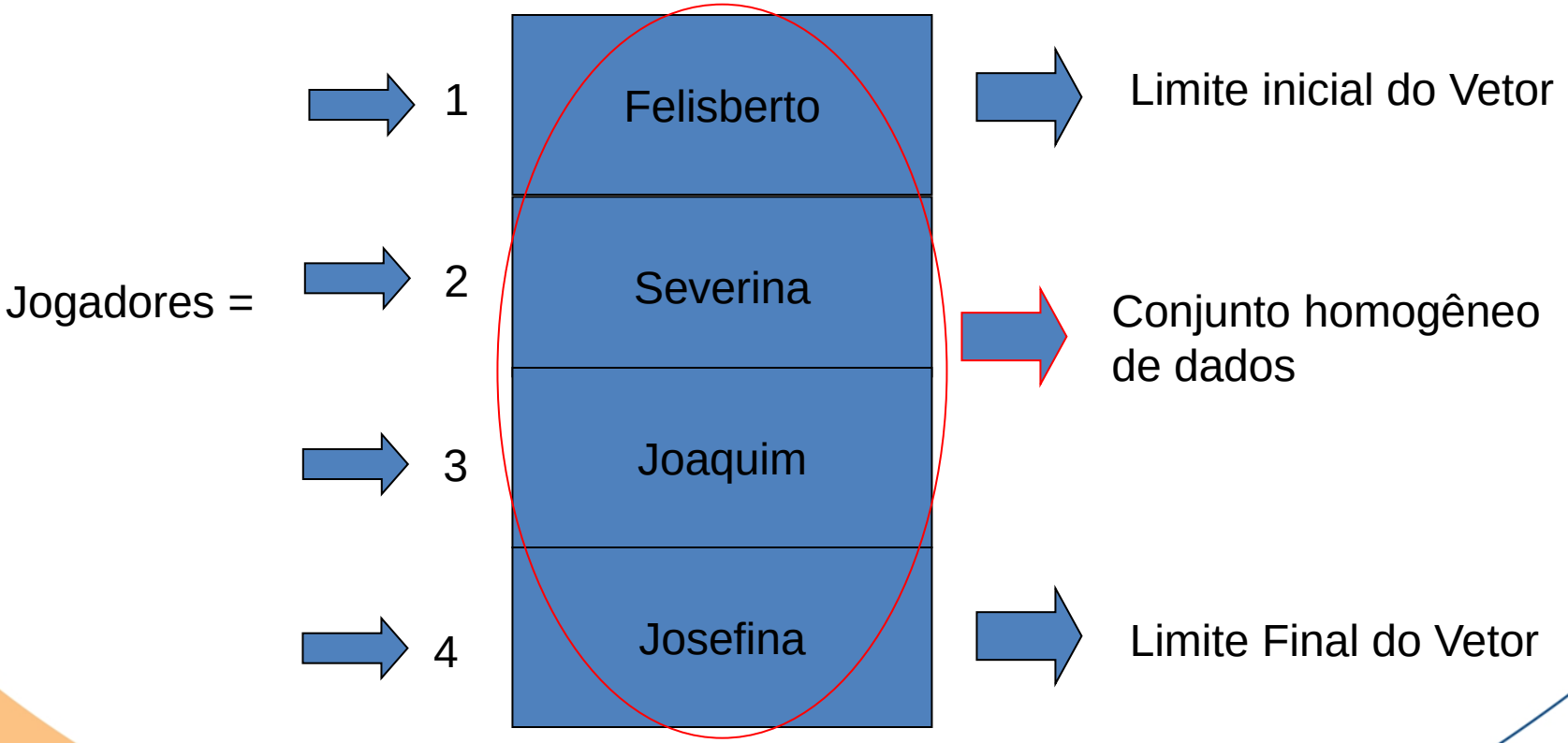
Li -> Limite inicial do vetor;

Lf -> Limite final do vetor;

<tipo primitivo> -> Representa qualquer um dos tipos primitivos;

Variáveis compostas unidimensionais

- Exemplo: Tipo Tjogador = Vetor[1..4] de caractere;
  Variável Jogadores: Tjogador;



Variáveis compostas unidimensionais

- Faça um programa que carregue um vetor de 10 posições de valores inteiros e mostre como resultado o maior e o menor valor do conjunto.

- Faça um programa que carregue dois vetores de 5 posições de valores inteiros e mostre como resultado um terceiro vetor com a soma dos valores lidos.

Variáveis compostas heterogêneas

- Quando uma determinada estrutura de dados for composta de variáveis com mais de um tipo primitivo, temos um conjunto heterogêneo de dados.

Temos um conjunto de dados com mais de um tipo primitivo, ou seja, temos um conjunto heterogêneo de dados.

Variáveis compostas heterogêneas

- Declaração:

Tipo <identificador> = Registro

<Campo1> : <Tipo de dado>;

<Campo2> : <Tipo de dado>;

<CampoN> : <Tipo de dado>;

fim;

Onde:

<identificador> -> Representa o nome da estrutura;

<Campo1> <Campo2> <CampoN> -> Representam as variáveis que compõem o registro;

<tipo de dado> -> Representa qualquer um dos tipos primitivos ou tipo definido.

Variáveis compostas heterogêneas

- Exercícios:

1) Crie uma estrutura para representar as informações de um Cheque;

2) Defina uma estrutura que represente uma passagem de ônibus interestadual;

3) Elabore uma estrutura para representar as informações de uma agenda de compromissos.

Variáveis compostas unidimensionais

A secretaria de transportes de uma determinada cidade, quer ter um controle melhor sobre os veículos que circulam na cidade, para tanto solicitou ao setor de TI uma solução para o problema, este resolveu fazer um levantamento dos veículos existentes na cidade. Foram informados para cada veiculo: A placa, ano, motor, chassi, cor e montadora.

Pede-se:

- A quantidade de veículos existente na cidade;
- A quantidade de veículos com mais de dez anos de circulação;
- A quantidade de veículos com mais de vinte anos de circulação;
- O percentual de veículos com menos de cinco anos de circulação em relação ao total de veículos;
- O percentual de veículos com motor 1.0 cujo ano seja superior 2000 e a cor seja branca.

Variáveis compostas multidimensionais

- Quando uma determinada estrutura de dados for composta por mais de uma dimensão (Matriz).

Colunas

	1	2	3	4	5	6	7
1	Ana	Bruno	Carol	Dani	Edson	Fabio	Gino
2	Laura	Yan	Jose	Maria	Paulo	Joao	Ivo
3	Luiza	Rose	Davi	Pedro	Beti	luis	Artur

Dados

Linhas

Variáveis compostas multidimensionais

- Declaração:

  Tipo <identificador> =

  Vetor[Li1..Lf1, Li2..Lf2] de <tipo>;

  Onde:

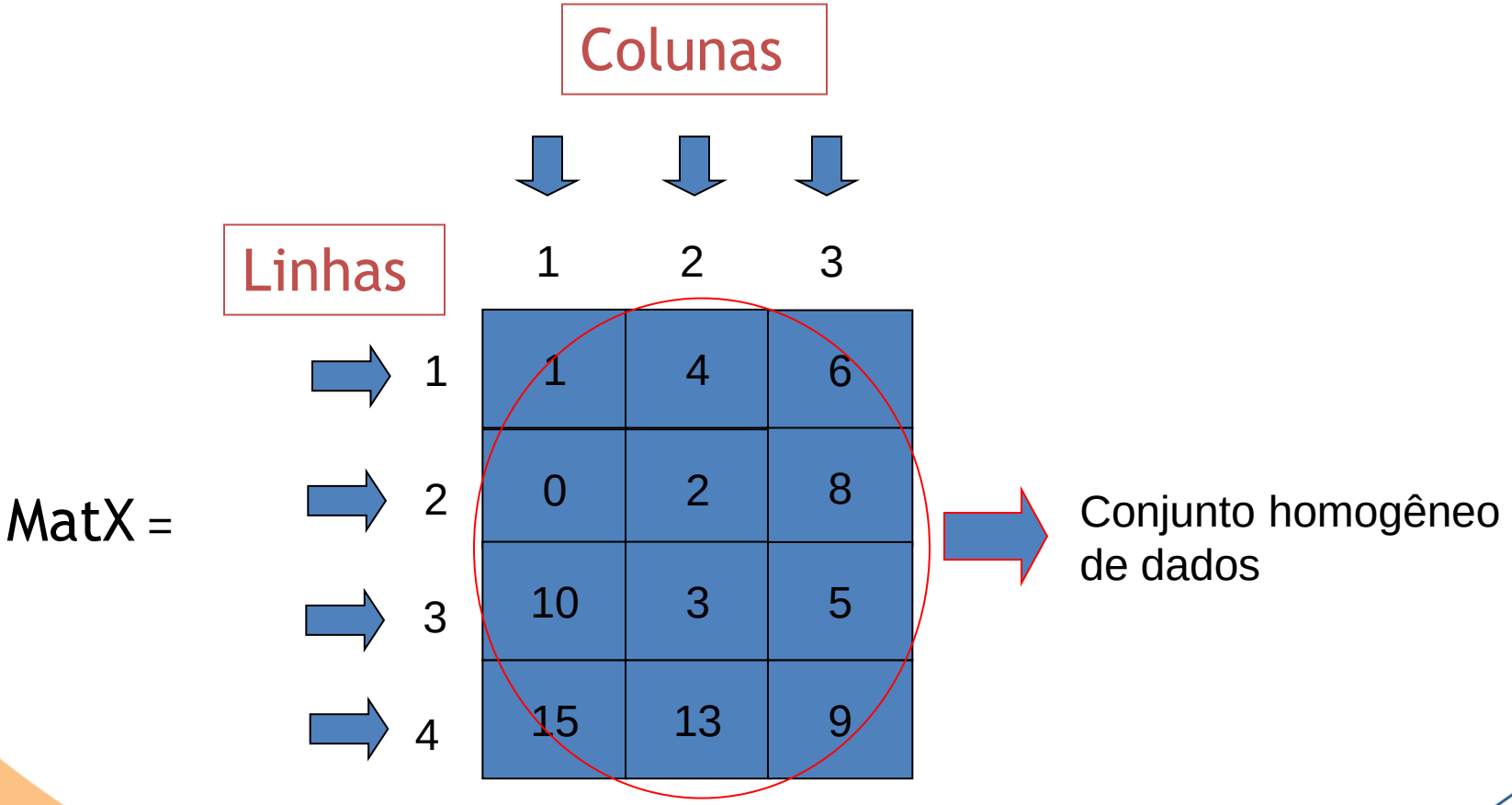
  <identificador> -> Representa o nome da estrutura;

  Li1..Lf1, Li2..Lf2 -> são os limites dos intervalos;

  <tipo> -> Representa qualquer um dos tipos primitivos ou outra estrutura;

Variáveis compostas multidimensionais

- Exemplo: Tipo TMat = Matriz[1..4, 1..3] de inteiros;
  Variável MatX: TMat;



Variáveis compostas multidimensionais

- Exercício:

  Construa um algoritmo que carregue uma matriz 4X4 de inteiros e mostre como resultado os valores da diagonal principal.

Variáveis compostas multidimensionais

- Exercício:

Crie um programa que contenha uma matriz chamada *MatrizDeReais4x2*, com 4 x 2 números reais.

Métodos para:
a) Receber os valores;
b) Retornar a multiplicação dos valores de uma dada coluna;
c) Retornar a soma dos valores de uma dada linha;
d) Retornar a soma de todos os elementos da matriz.

Variáveis compostas multidimensionais

- Exercício:

Considere uma matriz quadrada de inteiros de ordem 6. Crie um programa que carregue a matriz e resolva as seguintes questões:

a) a soma dos elementos da diagonal principal da matriz;
b) o menor valor par da matriz;
c) Mostrar os números primos da matriz;
d) uma nova matriz da mesma ordem, porém transposta.

Faça um programa que entre com um conjunto de dados contendo a idade e o grau de escolaridade de um grupo de pessoas. O grau de escolaridade deve ser: 1 – Ensino básico;
2 – Ensino Médio;
3 – Ensino Superior;
4 – Pós-graduado;
5 – Mestrado;
6 – Doutorado.

Mostrar como resultado:

a) A quantidade de pessoas para cada grau de escolaridade;
b) O percentual de pessoas para cada grau de escolaridade;
c) O grau de escolaridade com o maior e o menor numera de pessoas;
d) O percentual de pessoas que concluíram o ensino superior antes dos 24 anos de idade;
e) A média de idade das pessoas que concluíram o mestrado;
f) A menor idade com doutorado concluído.

Para efetuar o recolhimento do Imposto de Renda a Receita Federal tem o NOME, CPF, UF (RS, PR e SC) e RENDA ANUAL de cada contribuinte, durante o ano.
EX.: Nome: João da Silva CPF: 123.456.789-00 UF: PR
Renda Anual: R$10.000
Para o cálculo do imposto a pagar de cada contribuinte, considere o seguinte:

Nível de Renda Anual	Alíquota
0 a 4.000	0%
4.001 a 9.000	5,8%
9.001 a 25.000	15%
25.001 a 35.000	27,5%
acima de 35.000	30%

Sendo assim, deve-se calcular o imposto a pagar do seguinte modo:

Imposto a pagar = Renda Anual * Alíquota

Faça um programa orientado a objetos que leia as informações a serem digitadas pela Receita.
Ao final do programa será possível:

- digitar o CPF de algum contribuinte e ver seus dados e o imposto a pagar;
- saber os dados e o imposto a pagar do contribuinte que tem o maior imposto a pagar;
- saber os dados e o imposto a pagar do contribuinte do RS que tem a menor renda anual;
- saber a participação % de cada estado no total de impostos a serem recebidos pela Receita.