**LÓGICA DE PROGRAMAÇÃO**

17/10 - Soluções estruturadas para a resolução de problemas: **NARRATIVA, FLUXOGRAMA e PSEUDOCÓDIGO.**

Na programação, para resolvermos um problema, é necessário dividir o problema em pequenos passos, para que no final como um todo, possamos executar determinada tarefa (isso se chama **operacionalização de problemas**). É como uma receita de bolo. Para fazermos o bolo, precisamos seguir uma série de passos até o resultado final.

**Narrativa:** descrever detalhadamente os passos para realizar uma tarefa.

**Fluxograma:** divisão dos passos representada por ícones (setas, retângulos, etc).

Um exemplo de como operacionalizar um problema na programação é o de realizar a média aritmética entre dois números. Para isso, o computador precisa:

Ler o primeiro número **(entrada de dados)**

Ler o segundo número **(entrada de dados)**

Somar os dois números **(processamento de dados)**

Dividir os dois números por 2 **(processamento de dados)**

Mostrar o resultado obtido **(saída de dados)**

18/10 – **VARIÁVEL**

A variável é um espaço na memória do computador destinado a guardar dados.

Tipos de variáveis:

**Inteiro =** define apenas números inteiros.

**Real =** define números com vírgula.

**Caracter =** define apenas uma letra.

**Cadeia =** define frases (string).

19/10 – **VARIÁVEIS CONSTANTES**

As variáveis constantes são aquelas que não podem ter o seu valor alterado, Para definir uma variável constante, basta digitar antes da variável a função “const”.

**ATRIBUIÇÃO DE VARIÁVEIS**

Quando atribuímos um valor a uma variável, isso significa que vamos guardar algum dado dentro dela, ou seja, na memória do computador. Essa atribuição é feita usando o sinal de igual (=).

21/10 – **OPERADORES MATEMÁTICOS**

São todos aqueles que usamos para realizar cálculos matemáticos. Nas linguagens de programação, temos que:

Adição: +

Subtração: -

Multiplicação: \*

Divisão: /

Igual: ==

Assim como na expressão numérica, o computador resolve por uma ordem já definida (PEMDAS). Para definir algo para calcular primeiro na expressão numérica, usamos os parênteses. ()

Exemplo:

9 \* 4 + 2 = 38, pois o computador resolve primeiro 9 \* 4 e depois adiciona 2.

9 \* (4 + 2) = 54, pois o computador resolve primeiro 4 + 2 e multiplica o valor obtido com 9.

**COMANDOS DE ENTRADA E SAÍDA DE DADOS**

A entrada de dados ocorre quando queremos que o usuário interaja escrevendo algum dado, e isso ficará salvo em uma variável. A saída ocorre quando queremos mostrar alguma informação para o usuário.

Para ENTRADA, usamos LEIA

Para SAÍDA, usamos ESCREVA

22/10 – **ESTRUTURAS DE SELEÇÃO (SE, SENÃO, ESCOLHA, CASO)**

**SE SENAO SE -** As estruturas de seleção são utilizadas para verificarmos algo em nosso código. Com essas estruturas de seleção, nós podemos estabelecer condições e fazer escolhas.

Operadores de condições:

>: maior

<: menor

>=: maior ou igual

<=: menor ou igual

==: igual

e (n == 18 e n>10)

ou (n<7 ou n>11)

Exemplo:

se (idade >= 18){

escreva(“Você é maior de idade!”)

}senão{

Escreva(“Você é menor de idade!”)

}

O **“se”** é usado para determinar uma condição. Se essa condição for verdadeira, algo é executado. Há a possibilidade de criarmos outra condição paralela a essa, e nesse caso, usamos o “**senão se”.** Se essa condição for verdadeira, outro bloco de códigos é executado. Se ambas as condições forem falsas, outra coisa será executada. É nessas horas que usamos o **“senão”.**

24/10 - **ESCOLHA-CASO**

Essa estrutura de seleção é muito usada em menus, reduz a complexidade do problemas e cria desvios condicionais.

25/10 – **ESTRUTURAS DE REPETIÇÃO**

É uma estrutura que nos permite executar mais de uma vez o mesmo comando ou um conjunto de comandos. Há três principais estruturas de repetição: **enquanto, faca enquanto e para.**

**Enquanto** = a função do comando enquanto é executar uma lista de comandos enquanto uma determinada condição for verdadeira.

**Faca enquanto =** é muito semelhante a função enquanto. A diferença é que ela executa o código antes de fazer a verificação.

**Para** = é usado para controlar a repetição de algo. A sintaxe do laço “para” é:

para (inteiro i = 0; i < 10; i++)

Isso indica que eu quero que algo seja repetido até 10 vezes.