Introdução

Para ser um bom programador de computadores, independentemente da linguagem de programação, o estudante tem que adquirir uma boa base em lógica de programação, em especial, lógica algorítmica.

O conteúdo de lógica de programação é oferecido nos cursos de: tecnologia de com ênfase em informática, engenharias, ciência da computação, física, matemática, geologia, estatística, enfim a qualquer curso que vá utilizar o computador como uma ferramenta para otimizar seus processos.

Um dos objetivos deste material é permitir que o estudante desenvolva, gradual e progressivamente algoritmos eficientes para as suas necessidades acadêmicas. O outro objetivo é mostrar ao aluno que um bom entendimento da linguagem algorítmica propiciará um bom recurso para escolher uma linguagem de programação que poderá ser utilizada na sua vida profissional.

Neste material foi utilizada uma linguagem(sintaxe) simples, clara e objetiva para aproximar os termos (jargões) da informática à nossa linguagem coloquial. Para isso foram introduzidos alguns conceitos iniciais com intuito de preparar o aluno no uso simbologia e os termos adequados ao tema .

O material foi estruturado na seguinte ordem: uma abordagem simples e direta dos conceitos, exemplos, uma carga de exercícios resolvidos e uma outra carga de exercícios propostos.

Para um bom entendimento dos conceitos é importante que o aluno tenha paciência e perseverança, pois os resultados só virão após algum tempo de dedicação ao assunto. É recomendável que o aluno só avance para o próximo item quando as dúvidas forem totalmente sanadas. Lembro, também que a consulta a outras fontes faz parte de um melhor aprendizado.

Aproveite bem o material.
Prof. Msc. Carlos Alberto Bezerra e Silva.

"A arte de programar consiste na arte de organizar e dominar a complexidade.(Dijkstra)"

Fundamentos

Algoritmo

–Conjunto pré-determinado de regras ou métodos bem definidos com o propósito de resolver problemas com um número finitos de procedimentos.

–Definição rigorosa de uma série de operações a serem efetuadas para obtermos um resultado desejável em um número definido de etapas.

–Uma seqüência lógica de instruções passo a passo destinados à soluções de um determinado problema.

Programa

–Conjunto de instruções codificadas, em linguagem específica para computadores(máquinas), que controlarão o computador(máquina) no percurso da execução de uma operação ou de uma série de operações desejáveis para a resolução de um problema definido.

Exemplo:

Uma I.E.S. deseja calcular a média dos seus alunos utilizando como critérios: peso 2 para a primeira avaliação e peso 3 para a segunda avaliação.

Linguagem de programação

–Linguagem utilizada para transcrever o algoritmo em programa, ou seja, um código inteligível e executável por máquina(computador).

Exemplo:

Pascal, Algol, Basic, C, C++, Fortran, Java, Delphi, Visual Basic, Cobol etc.

Linguagem Algorítmica

–É uma Pseudolinguagem que é utilizada para o ensino nas linguagens de programação.

Simbologia e suas funcionalidades.

–A simbologia que será descrita nos próximos tópicos será utilizada nos algoritmos e em algumas linguagens de programação.

- Os dois pontos :

–Utilizado para declararmos variáveis.

Exemplo:

```
Nota:Numérica.
Salario:Numérica
Nome:Caracteres.
```

- O ponto e vírgula ;

–Utilizado para finalizarmos uma instrução(sentença).

Exemplo:

```
Escreva(x);
Leia(y);
```

- A vírgula ,

–Utilizada para separarmos variáveis e constantes.

Exemplo

```
Nome, Idade;
Salario, HoraExtra;
```

- O Ponto .

–É utilizado para finalizar o algoritmo.

Exemplo

```
Início
...
Fim.
```

- Também é utilizado para separar a parte inteira da parte decimal de um número.

Exemplo

```
7.5,  324.56
```

As aspas simples(apóstrofo) ' '

–Utilizada para separarmos conteúdos literais.

Exemplo

```
'Segunda', 'Azul', 'Josineura'
```

Chaves { }

- Utilizado para aumentar a clareza do algoritmo, devem aparecer entre chaves { }. Os comentários não serão processados pelo algoritmo.

Exemplo

```
{ Este programa foi criado no dia 10/03/2008 por Carlos Alberto}
     Var  Cód,                {Código do curso}
          Nota,               {Nota do Aluno}
          Mat:Numérico,            {Matrícula do aluno}
```

Dois pontos igual :=

- Utilizado para atribuirmos um valor a uma variável

Exemplo:

```
          Idade:=32; Sexo:='M'; Peso:=90.
          Media=(A1+A2)/5;
```

Constantes =

- As constantes armazenam valores fixos, que não se alteram durante a execução do algoritmo. Os conjuntos podem assumir valores: numéricos, lógicos e literais, são representadas por identificadores e recebem seus valores por meio do símbolo da igualdade.

Identificador_de_constante=constante

Exemplo:

```
          Ano=2008; Filhos=VERDADEIRO; Bolsa_Estudo=Falso.
```

Identificadores

- É formado por uma letra ou uma letra seguida de outras ou dígitos. Não permite-se o uso de espaços ou qualquer outro caractere que não seja letra ou dígito.

Exemplo

```
          A, X15, Matricula.
          X-Y, A:B, 5B são inválidos.
```

Palavras reservadas

- São identificadores que só podem ser usados com um sentido prefixado na linguagem. Também são conhecidos como palavras chaves. As palavras abaixo serão consideradas reservadas neste material.

E	VETOR	INÍCIO	CASO
CONST	DIV	FAÇA	ATÉ
SE NÃO	FIM	ARQUIVO	PARA
FUNÇÃO	IR PARA	SE	ALOQUE
ETIQUETA	MOD	NULO	NÃO
OU	PROCEDIEMNTO	PROGRAMA	DE
REGISTRO	REPITA	CONJUNTO	ENTÃO
TIPO	VAR	ENQUANTO	COM
FECHAR	ABRIR	ASSINALAR	NIL

Constantes numéricas.

- São formadas por seqüências de dígitos que podem ou não ser precedidos por um sinal positivo(+) ou negativo(-). Eles também podem ser seguidos por um ponto(.) decimal e outras seqüência de dígitos

Exemplo

25, +3254, 3.1415, 2.78, -0.5.

.

Constantes lógicas

- São duas e representa-se pelas palavras VERDADEIRO e FALSO. Também são chamadas por booleans.

Constantes literais

- As constantes literais são representadas pelos caracteres(letras, dígitos ou símbolos) colocados entre apóstrofos**(‘’)**

Exemplo

‘Televisão’, ‘********’, ‘RA9620’, ‘07/03/2008’, ‘H2O’.

Variáveis

–São representadas por identificadores e podem ser numéricas, lógicas e literais. Ao contrário das constantes as variáveis armazenam valores que podem ser alteradas durante a execução do algoritmo.

– Associam um endereço de memória a um nome.

Utilizando a linguagem algorítmica

Os algoritmos devem ser iniciados pela palavra reservadas **Algoritmo** seguida de um título, este de ponto e vírgula(;) para indicar o final da linha. Logo abaixo devemos adicionar as **Declarações** das **constantes** e das **variáveis** e na seqüência vem o **bloco** de **instruções** ou **comandos.**
Exemplo

```
Algoritmo Título_do_algoritmo;
Declarações
Bloco.
```

O bloco de instruções é formado por comandos entre as palavras reservadas **Início** e **Fim**

Exemplo

```
Algoritmo Título_do_algoritmo;
Declarações
Início
        Comandos;
Fim.
```

As variáveis utilizadas nos algoritmos são declaradas, isto é, são identificadas com um nome e indicam os tipos de dados que elas armazenam. As declarações são usadas para validar o uso de qualquer identificador que não seja pré-definido, indicando-se as características da variável que será representada. Todas as variáveis declaradas num bloco, devem ser incluídas numa única declaração da forma:

Exemplo

```
Var   Lista_de_identificadores: Tipo;
      Lista_de_identificadores: Tipo;
      ...
```

Onde:

Lista_de_identificadores Representa o Identificador.
Tipo Tipo de variavel.

Tipos pré-definidos da LINGUAGEM ALGORÍTMICA:

NUMÉRICO, LÓGICO, LITERAL

A utilização do tipo NUMÉRICO se dará quando quisermos associar o conteúdo da variável a números Inteiros ou Reais.

A utilização do tipo LÓGICO se dará quando quisermos associar o conteúdo da variável a uma situação VERDADEIRA OU FALSO.

A utilização do tipo LITERAL se dará quando quisermos associar o conteúdo da variável a uma cadeia de caracteres.

Exemplo

```
Var   SALARIO, X5: Numérico;
      CONTADOR, X5: Numérico;
      TESTE, FLAG: Lógico;
      NOME, FONE: Literal;
```

Declaração de tipos:

Na linguagem algorítmica utilizada neste material o aluno poderá criar novos tipos de dados além daqueles considerados pré-definidos. Um novo tipo é criado através de uma definição que determina um conjunto de valores e os associa a um identificador este conjunto. A declaração de tipo precede a declaração de variáveis.

```
TIPO  Identificador_de_tipo= descrição;
      Identificador_de_tipo= descrição;
      Identificador_de_tipo= descrição;
      ...
```

Onde TIPO é a palavra chave que inicia a declaração de tipos.
Neste material também será possível criar um tipo formado por elementos consecutivos de um tipo pré-definido.

```
TIPO  Identificador_de_tipo= Intervalo;
      Identificador_de_tipo= Intervalo;
      Identificador_de_tipo= Intervalo;
      ...
```

Onde TIPO é a palavra chave e Intervalo representa o tamanho do identificador

```
Var   Lista_de_identificadores: Tipo;
      Lista_de_identificadores: Tipo;
      ...
```

Exemplo

```
TIPO  Meses = 1..12;
Var   Mês : Meses;
```

Neste caso a variável Mês é do tipo Meses assumindo valores na faixa de 1..12.

Expressões Aritméticas:

-O sinal da multiplicação é um asterisco: *
-O sinal da divisão é uma barra: /
-O quociente inteiro de uma divisão com operandos inteiros será calculado com o operador: **DIV**
-O resto de uma divisão inteira será calculado pelo operador **MOD**.
-A exponenciação será indicada pelo circunflexo: **^**

-A subtração é representada por **-**

Exemplos

9 **/** 2 é igual a 4.5 9 **DIV** 2 é igual a 4 9 **MOD** 2 é igual a 1

Quanto a **prioridade** observe a tabela a seguinte.

Prioridade	Operadores
1ª	*** / DIV MOD**
2ª	**+ -**

Exemplos

X + Y; TOTAL / N; A * B + C; (NOTA1 * 2 + NOTA2 *3) / 5.

Funções numéricas pré-definidas:

Nome	***Resultado da Função***	***Tipo***
SQRT(x)	Raiz quadrada de x	Real ou inteiro
SQR(x)	Quadrado de x	Real ou inteiro
ABS(x)	Valor absoluto de x	Real ou inteiro
INT(x)	Retorna a parte inteira de x	Inteiro

Expressões lógicas:

- As expressões lógicas comparam valores utilizando um dos operadores

=	Igual	**< >**	Diferente
<=	Menor ou igual	**<**	Menor
>=	Maior ou igual	**>**	Maior

O resultado das expressões lógicas serão sempre um valor lógico VERDADEIRO ou FALSO.

Obs. A expressão A + B = C será verdadeira ou falsa dependendo do conteúdo do lado esquerdo da expressão A +B podendo ser igual ou diferente a C.

Exemplo

X<>Y	Nome=’Jeguinaldo’
A * B – C<0	Ano=2008

Exercício. Com as declarações abaixo preencha a tabela com V de verdadeiro e **F** de falso.

```
Var  X,Y:Numérico;
     Nome,Sexo:Literal;
```

X	*Y*	*Nome*	*Sexo*	*X+Y>=SQRT(Y)*	*Nome<>'Ordicléia'*	*Sexo='F'*
4,0	20	'joão'	'M'			
6,0	40	'Mario'	'M'			
1,5	7	'Ordicléia'	'F'			
6,0	2	'Ana'	'F'			
5,5	7	'Jeguinaldo'	'M'			

Conjunção/Disjunção optativa OU

- Duas ou mais expressões unidas pelo operador OU formam uma sentença verdadeira se pelo menos uma delas for verdadeira.

Exemplo

```
5 >= 5 OU 3 < 8    é VERDADEIRA
7 = 7 OU 1< 10          é VERDADEIRA
3 <= 1 OU 5 = 8    é FALSO
```

Advérbio de negação NÃO

- Retorna o valor lógico oposto ao da expressão

Exemplo

```
Não(10 > 8) é FALSO
Não(3 = 5)  é VERDADEIRA
```

Tabela de prioridade das operações

Prioridade	*Operadores*
1	NOT
2	* , / , DIV, MOD , AND
3	+ , - , OR
4	= , < > , < , > , >= , <=

Pode-se usar diversos níveis de parentêses para dar uma ordem de execução às expressões distintas da tabela acima.

Comandos de entrada e saída.

- O comando de **entrada(input)** que será utilizada na nossa linguagem algorítmica tem a seguinte sintaxe:

Leia(Lista_de_identificadores);

Onde,
Leia: Lê o valor da variável ou variáveis de uma unidade de entrada de dados como teclado por exemplo.
Lista_de_identificadores: Nome das variáveis.

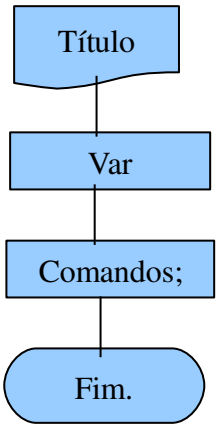
O comando de **saída(output)** da nossa linguagem algorítmica tem a seguinte forma:

Escreva(Identificador e/ou constantes e/ou expressões)

Onde,
Escreva: Escreve o valor da variável ou variáveis de uma unidade de saída de dados como monitor por exemplo
Lista_de_identificadores: Nome das variáveis.
Constantes: Constantes literais entre apóstrofos**(' ')**
Expressões: Expressões entre vírgulas**(,)**

Estrutura seqüencial

- Os algoritmos que escreveremos terão que iniciar com a palavra reservada ALGORITMO seguido de um título e um ponto e virgula(;). Em seguida temos a área de declarações que compreende as instruções que se referem as variáveis e as constantes que serão utilizadas no algoritmo. É importante lembrar que as tais variáveis e as constantes não podem ser declaradas em outro lugar do algoritmo. Após as declarações vem o bloco de instruções que devem iniciar com as palavras INÍCIO seguido das instruções do algoritmo finalizados com ponto e vírgula(;) finalizando com a palavra FIM

Linguagem algorítmica	***Fluxograma***
Algoritmo título; Declarações **Var** Identificadores: Tipo; **Início** Comando 1; Comando 2; ... Coamndo n;	

Entre as palavras **Início** e **Fim** devem aparecer as instruções para que os dados sejam lidos através do dispositivo de entrada(teclado), processados com as devidas instruções, atribuições, expressões etc e em seguida exibidos no dispositivo de saída(monitor).

Exemplo. Algoritmo que lê dois números e calcula a média aritmética entre eles.

Algoritmo	Codificação na linguagem Pascal
Algoritmo Media;	**Program** Media;
Var A1,A2,Med:Numérica;	**Var** A1,A2,Med:**Real**;
Início	**Begin**
Escreva('Digite o primeiro número');	**Write**('Digite o primeiro número');
Leia(A1);	**Read**(A1);
Escreva('Digite o segundo número');	**Write**('Digite o segundo número');
Leia(A21);	**Read**(A21);
Med:=(A1+A2)/2;	Med:=(A1+A2)/2;
Escreva('A média entre os número é',Med);	**Write**('A média entre os número é',Med);
Fim.	**End.**

Exercícios resolvidos

01 – Escreva um algoritmo para calcular e exibir a média de três números quiasquer.

```
Algoritmo Media_de_três_Num;

Var   X, Y, Z, Media:Numérico;

Início
      Escreva(‘Digite três números’);
      Leia(X, Y, Z);
      Escreva(‘A média é:’+Media);
Fim.
```

02 – Escreva um algoritmo que realize o cálculo do salário de um funcionário. O salário é baseado no número de salário mínimos, Leia o salário mínimo vigente e o número de salários recebido.

```
Algoritmo Salarios;

Var   Sal_Min, Num_Sal, Salario:Numérico;

Início
      Escreva(‘Digite o valor do salário mínimo ➔ R$ ’);
      Leia(Sal_Min);
      Escreva(‘Digite o número de salários é ➔ ’);
      Leia(Num_Sal);
      Salário:=Sal_Min*Num_Sal;
       Escreva(‘O salário do funcionário é ➔ ’, Salario);
Fim.
```

03 – Escreva um algoritmo que calcule o preço de uma mercadoria dados o seu peso por Quilo.

```
Algoritmo Preço_Produto;

Var   Preço, Preço_Quilo, Peso:Numérico;

Início
      Escreva('Digite o preço por quilo ➔ R$ ');
      Leia(Preço_Quilo);
      Escreva('Digite o peso do produto ➔ ');
      Leia(Peso);   Preço:=Peso * Preço_Quilo;
      Salário:=Sal_Min*Num_Sal;
       Escreva('O preço do produto  é ➔ ', Preço);
Fim.
```

04 – Imagine um carro de tanque cheio. Escreva um algoritmo para calcular o consumo médio de combustível do carro. Leia a capacidade máxima do tanque quantos Km são percorridos usando todo o combustível.

```
Algoritmo consumo_médio;

Var   consumo, km, capacidade:Numérico;

Início
      Escreva('Qual a capacidade do tanque em Litros ➔ ');
      Leia(capacidade);
      Escreva('Quantidade de quilômetros  percorridos ➔ ');
      Leia(km);     consumo:=km / capacidade;
      Escreva('O consumo foi de', consumo, 'km por litro !');
Fim.
```

Estrutura condicional.

- As estruturas condicionais alteram a execução do algoritmo impondo condições para que determinadas linhas sejam ou não executadas. A estrutura condicional mais comum é o **"Se"**, que pode aparecer sob duas formas.

Sem desvio.

```
      Se Condição Então
          Comando;
```

O comando só será executado se a condição for verdadeira. A condição é uma expressão lógica.
A comando pode ser simples ou composto. Considere composto aquele formado por diversos comandos delimitados pelas palavras Início e Fim.

```
Se Condição Então
        Início
          Comando;
        Comando;
        Comando;
        Fim;
```

Estrutura com desvio.

```
Se Condição Então
        Comando 1
Se não
        Comando 2;
```

Nessa estrutura o algoritmo executará o comando 1 se a condição for verdadeira e executara o comando 2 se a mesma for falsa. Os comandos 1 e 2 também podem ser compostos.

Exercícios resolvidos

5 – Faça um algoritmo que calcule a média das duas notas de um aluno e imprima a mensagem "Aprovado" se a média for maior ou igual a 6,0(seis) e se imprima "Retido" em caso contrário.

```
Algoritmo Média_Aluno;

Var   Média, nota1, Nota2:Numérico;

Início
        Escreva('Digite a nota 1 ➔ ');
        Leia(Nota1);
         Escreva('Digite a nota 2 ➔ ');
        Leia(Nota2);
        Média:=(Nota1+Nota2)/2;
        Se (Média>=6) Então
                Escreva('Aprovado !');
        Se não
                Escreva('Retido !');
Fim.
```

6 – Faça um algoritmo que imprima uma das mensagens "Número par" ou "Número Ímpar'.

```
Algoritmo Par_Impar;

Var   Número:Numérico;
```

```
Início
        Escreva('Digite um núemero ➔ ');
        Leia(Número);
        Se (Número MOD 2) Então
                Escreva('O número é par !');
        Se não
                Escreva('o número é ímpar');
Fim.
```

7 – faça um algoritmo que leia o salário de uma pessoa e calcule o imposto de renda a ser pago obedecendo a seguinte tabela.

Salário	Imposto
<500	*0*
>=500 e <800	*0,15*
>=800 e <1100	*0,3*
>1100	*0,4*

```
Algoritmo Imposto_Salário;

Var   Salário:Numérico;

Início
        Escreva('Digite o valor do salário ➔ ');
        Leia(Salário);
        Se (Salário < 500) Então
                Escreva('Isento de imposto !');
        Se não
                Se (Salário < 800) Então
                 Escreva('O imposto é de ',Salário * 0.15);
            Se não
                Se (Salário < 1100) Então
                Escreva('O imposto é de ',Salário * 0.30);
             Se não
                Escreva('O imposto é de ',Salário * 0.40);
Fim.
```

8 – Escreva um algoritmo que leia três valores numéricos e mostre o menor deles.

```
Algoritmo Menor_três;

Var   N1,N2,N3, Menor:Numérico;

Início
      Escreva('Digite o primeiro valor ➔ ');
      Leia(N1);
       Escreva('Digite o segundo valor ➔ ');
      Leia(N2);
       Escreva('Digite o terceiro valor ➔ ');
      Leia(N3);
      Se (N1 < N2) E (N1<n3) Então
            Menor:=N1
      Se não
             Se (N2<N3) Então
              Menor:=N2;
          Se não
             Menor:=N3;
           Escreva('O menor é',Menor);
Fim.
```

9 – Escreva um algoritmo que receba três valores e informe se os mesmos representam os lados de um triângulo ou não. Caso seja, classifique-os quanto aos lados. Eqüilátero(três lados iguais), isósceles(dois lados iguais) e escaleno(todos os lados distintos)

```
Algoritmo triângulo;

Var   x,y,z:Numérico;

Início
      Escreva('Digite o primeiro valor ➔ ');
      Leia(x);
       Escreva('Digite o segundo valor ➔ ');
      Leia(y);
       Escreva('Digite o terceiro valor ➔ ');
      Leia(z);
      Se (x< y + z) e (y< x + z) E (z< x +y) Então
            Se (x = y) E (x = z) Então
                  Escreva('O triângulo é eqüilátero')
            Se não
                  Se (x = y ) OU (x = z) OU (y =z )   Então
                  Escreva('O triângulo é isósceles')
          Se não
            Escreva('O triângulo é escaleno');
      Se não
            Escreva('Os valores não representam os lados de um triângulo');
Fim.
```

10 – Faça um algoritmo que ordene três números fornecidos por uma usuário.

```
Algoritmo números_crescentes;

Var   x,y,z,aux:Numérico;

Início
      Escreva('Digite o primeiro valor ➔ ');
      Leia(x);
       Escreva('Digite o segundo valor ➔ ');
      Leia(y);
       Escreva('Digite o terceiro valor ➔ ');
      Leia(z);
      Se (x > y) OU (x > z) Então
            Se (y < z ) Então
                  Início
                        aux:=x;
                        x:=y;
                        y:=aux;
                  Fim
            Se não

                  Início
                        aux:=x;
                        x:=z;
                        z:=aux;
                  Fim;

                  Se ( y > z ) Então

                      Início
                        aux:=y;
                        y:=z;
                        z:=aux;
                   Fim;
             Escreva(x , y , z );

Fim.
```