UNISINOS - UNIVERSIDADE DO VALE DO RIO DOS SINOS
CENTRO DE CIÊNCIAS EXATAS E TECNOLÓGICAS (C6/6) – Curso: Informática

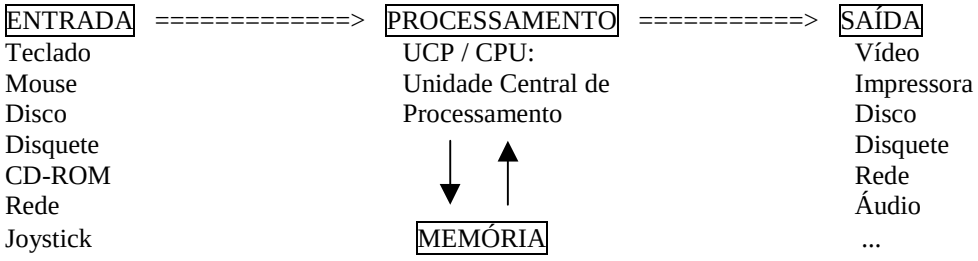
PROGRAMAÇÃO I – AULA 01

Disciplina: Linguagem de Programação PASCAL
Professor responsável: ***Fernando Santos Osório***
Semestre: 2001/2
Horário: 63

E-mail: ***osorio@exatas.unisinos.br***
Web:
http://inf.unisinos.br/~osorio/prog1.html
Xerox : *Pasta 54* (Xerox do C6/6)

1. CONCEITOS BÁSICOS:

Modelo de computador => Arquitetura de Von Neuman



Dispositivos Padrões: Entrada = Teclado
Saída = Vídeo
Processamento = Microprocessador INTEL Pentium
Memória Principal = RAM

1.1. ALGORITMO:

Seqüência de procedimentos que são executados *seqüencialmente* com o objetivo de resolver um problema especificado (conhecido).

Exemplo de algoritmo: "Como fazer um bolo"

Pegar Farinha	1 xícara	
Pegar Ovos	½ dúzia	ENTRADA
Pegar Leite	100 ml	
Pegar Fermento	1 colher pequena	
Misturar a farinha ao leite		
Bater os ovos		PROCESSAMENTO
Misturar os demais ingredientes		
Colocar no forno durante 20 minutos		
Retirar, Esfriar e comer		SAÍDA

Linguagem usada: Português (linguagem de alto nível)
Elementos manipulados: ingredientes (ovos, leite, farinha, ...)

1.2. COMANDOS:

Comandos ou Instruções = Conjunto de palavras-chave de uma linguagem de programação que tem por finalidade dizer ao computador como ele deve executar uma tarefa. Indica como armazenar e manipular as informações (obtidas através dos dispositivos de entrada, guardadas na memória, manipuladas pela CPU, e enviadas para os dispositivos de saída).

- No exemplo de algoritmo descrito acima (receita de bolo), os comandos eram palavras-chaves representados por *verbos* da língua portuguesa.
- Nos computadores os comandos são usualmente palavras originária da língua inglesa. Exemplo: write, read, do, ...

1.3. PROGRAMA:

Conjunto de comandos/instruções organizados em uma seqüência segundo uma lógica bem definida (algoritmo) e que servem para realizar uma tarefa. Exemplo de programa na linguagem PASCAL:

```
PROGRAM Calcula_Media_Simples;
VAR
  Valor1, Valor2: Integer;
  Media: Real;
BEGIN
  readln (Valor1);
  readln (Valor2);
  media := (Valor1 + Valor2 ) /  2;
  writeln (Media);
  readln;
END.
```

- Para fazer um programa teremos que armazenar informações na memória do computador, pois assim como na receita de bolo onde usamos elementos como ovos e farinha, no computador teremos que usar dados contidos em sua memória.
- Note a semelhança da estrutura do programa acima em relação a receita de bolo: no "VAR" vamos definir quais ingredientes (dados) vamos precisar e de que tipo eles são (farinha = pó, leite = líquido # Valor1 e Valor2 = números inteiros, Media = número real); após o "BEGIN" vem os comandos de processamento dos dados (entrada => processamento => saída), terminando pelo "END."

1.4 MEMÓRIA:

O computador armazena as informações necessárias a execução de um programa na sua memória, sendo que todas as informações devem ser lidas da entrada e armazenadas na memória, para que possam ser processadas, obtendo assim uma saída final.

A memória eqüivale a um conjunto de recepientes onde estão guardados os elementos que vamos manipular, estes recepientes são padronizados no computador, são os chamados *bytes*. Os bytes estão organizados de maneira ordenada, sendo todos numerados, e recebendo assim endereços que os identificam. Exemplo: o byte 234 poderá armazenar a idade de um funcionário de uma empresa, enquanto que os bytes 527 e 528 poderiam conter os dados referentes ao seu salário.

Vamos considerar a memória do computador como sendo uma coleção de gavetas numeradas onde poderemos armazenar ~~ingredientes~~, quero dizer, dados... ;^)

Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
Byte 6	Byte 7	Byte 8	Byte 9	Byte 10
Byte 11	Byte 12	Byte 13	Byte 14	Byte 15
Byte 16	Byte 17	Byte 18	Byte 19	Byte ...

- Dentro de cada gaveta vamos guardar informações como: a idade de uma pessoa, o seu salário, o seu nome, o seu peso, etc.
- Ao usar uma linguagem de programação vamos poder colocar etiquetas nestas gavetas, e assim vamos criar a idéia de *variáveis* e *constantes* em uma linguagem de programação (já pensou como seria difícil adivinhar em qual das gavetas numeradas acima que contem a informação referente ao sexo da pessoa ?). Nossa memória pode ficar assim com as etiquetas...

Idade (1)	Salário (2 e 3)		Peso (4)	PI Π (5)
Nome	Nome	Nome	Nome	Bytes 6 à 10
Byte 11	Byte 12	Byte 13	Byte 14	Byte 15
Byte 16	Byte 17	Byte 18	Byte 19	Byte ...

- Assim como os elementos (ingredientes) da receita possuem um tipo associado (líquido, pó, sólido, ...) e também onde temos uma unidade de medida associada a estes (ml, kg, xícara, colher, ...); as variáveis e constantes também vão possuir um tipo associado e capacidade de armazenamento relativas ao tipo escolhido. Exemplo:

Leite = líquido, capacidade do recepiente (0 à 1000 ml = 1 litro) <=== Receita
Idade = número inteiro (não fracionário), entre 0 e 150 <=== Computador

* Características dos dados:

- Em relação ao valor armazenado:

Variável = Valor que pode ser alterado durante a execução do programa. Exemplo: salário, média final do aluno, etc.

Constante = Valor que não é modificado durante toda a execução do programa. Ele é armazenado na memória para se usado em cálculos e processamentos, mas não é modificado. Exemplo: o valor do PI (Π), a nota mínima necessária para ser aprovado no semestre, etc.

- Em relação a grandeza do valor:

Os valores podem ter diferentes grandezas associadas a eles: valor inteiro pequeno (Idade, Peso), valor real pequeno ou grande (salário), valor real muito grande (dívida externa brasileira), dado não numérico (nome de uma pessoa), e assim por diante. Em uma linguagem de programação estruturada (Pascal) é importante ESPECIFICAR o TIPO DAS VARIÁVEIS para que estas sejam COMPATÍVEIS com os DADOS USADOS.

Você já pensou em armazenar leite em um saco de papel comum? A farinha pode ser, mas o leite não... e colocar 1 quilo de açucar em uma colher de sopa? Também não é possível, por isso nós vamos ver que em Pascal também não é correto armazenar o salário de uma pessoa em uma variável do tipo BYTE ou INTEGER.

- Em relação ao tipo de seu conteúdo:

  Vamos trabalhar basicamente com três tipos de dados principais...
  NUMÉRICOS, **CARACTERES** e **LÓGICOS**

Os *NUMÉRICOS* podem armazenar números, sendo subdivido em numéricos do **tipo INTEIRO** e numéricos do **tipo REAL**. Os inteiros são números não fracionários (sem casas após a vírgula) e os REAIS são números com uma parte fracionária (podem ter uma mantissa e um expoente, ou seja, podem ter casas após a vírgula). Os REAIS também são chamados de números de *ponto flutuante*.

Os CARACTERES podem armazenar letras (‘a’..‘z’; ‘A’..‘Z’), números (‘0’..’9’) ou outros caracteres especiais (‘@’,’#’, ‘$’, ‘%’, ‘*’, ‘\’, ...). Eles podem aparecer isolados (**tipo char**), ou agrupados formando palavras/frases (**tipo string**).

Os LÓGICOS podem armazenar valores lógicos, ou seja, verdadeiro ou falso (True/False). São as chamadas variáveis do tipo Booleano (originárias da lógica de Boole – Valores Binários).

- Em relação ao seu uso:

- Uma variável (posição da memória do computador = “gaveta”), pode armazenar apenas um valor por vez, portanto uma variável só pode armazenar um dado num certo instante de tempo. Se colocarmos um outro dado dentro de uma posição de memória que já possuía alguma informação, a informação anterior vai ser perdida, sendo guardada apenas a nova informação.
- Ao desligar o computador os dados contidos na memória principal da máquina serão apagados, devendo portanto serem salvos em algum lugar (escritos no disco, por exemplo). Ao terminar a execução de um programa, os dados relativos aquele programa também serão apagados da memória principal.

2. LINGUAGEM PASCAL:

A Linguagem PASCAL foi criada por Niklaus Wirth na Suiça entre os anos de 1968/1970. Esta linguagem é uma linguagem de alto nível, bem estruturada e desenvolvida com fins didáticos para o ensino de programação. Atualmente o Pascal é a linguagem mais usada nas Universidades para o ensino de programação.

O software que vamos usar é o Ambiente Integrado de Desenvolvimento de Programas TURBO PASCAL 7.0 que foi criado pela Borland, sendo também conhecido como Borland Pascal. O Turbo Pascal foi criado em 1983 (versão 1.0 para CP/M) e depois surgiram novas versões para PC-DOS, Windows3.11 e Windows95/Windows98. A Borland mudou recentemente de nome, chamando-se agora de Inprise, e tendo como principal produto a Linguagem Delphi, que nada mais é do que uma extensão da linguagem Pascal (inclui conceitos de orientação à objetos e construção de interfaces gráficas através de uma programação visual e interativa).

2.1. Notação utilizada:

Vamos usar a seguinte notação para representar nossos programas em Pascal:

- Textos entre '<' e '>' indicam que o conteúdo deste item não é um comando/dado verdadeiro da linguagem, onde podemos substituir o texto por um comando/dado/valor conforme a indicação fornecida. Exemplo:
  <comando>; Pode ser interpretado como: writeln; ou readln; ou media:=(a+b)/2; ou qualquer outro comando da linguagem...
  <valor> + <valor> Indica que devemos escrever valores ali: 5 + 5 ou 14.2 + 16.76 ou ...
  <texto> Indica que devemos escrever um texto ali: ' Hello World' ou '...'

- Textos entre '[' e ']' indicam que o conteúdo deste item é opcional, podendo ou não ser escrito. Exemplo:
  <comando>[;] Significa que o comando pode ou não ser seguido de um ponto-e-vírgula.
  IF <expressão> THEN <comando> [ELSE <comando>] ; O "else <comando>" é opcional.

2.2. Tipos de Variáveis:

TIPOS NUMÉRICOS DO PASCAL

° NÚMEROS INTEIROS:

TIPO INTEIRO	Tamanho em Bytes	Intervalo de valores representados
INTEGER	**2**	**-32768 .. +32767**
WORD	2	0 .. 65535 (só positivos)
BYTE	1	0 .. 255 (só positivos)
SHORTINT	1	-128 .. +127
LONGINT	4	-2.147.483.648 .. +2.147.483.647

° NÚMEROS FRACIONÁRIOS:

TIPO Ponto-Flutuante	Bytes	Dígitos	Intervalo de Valores
REAL	6	11-12	2.9E-39 .. 1.7E38
SINGLE	4	7-8	1.5E-45 .. 3.4E38
DOUBLE	8	15-16	5.0E-324 .. 1.7E308
EXTENDED	10	19-20	3.4E-4932 .. 1.1E4932
COMP	8	19-20	-9.2E18 .. +9.2E18

Notação científica: $1.0 \times 10^0 = 1 = 1.0E0 = 0.1 \times 10^1 = 0.01 \times 10^2 = 0.001 \times 10^3$
$1.0 \times 10^1 = 10 = 1.0E1 = 100 \times 10^{-1} = 1000 \times 10^{-2}$
$1.0 \times 10^2 = 100 = 1.0E2$
$1.0 \times 10^3 = 1000 = 1.0E3$
Outros exemplos... $1.2 \times 10^3 = 1200 \; / \; 5E\text{-}3 = 5.0 \times 10^{-3} = 0.005$

TIPOS COM CARACTERES: Char, String

TIPOS LÓGICOS: Boolean

2.3. Declaração de Variáveis e Constantes:

A declaração de uma variável/constante consiste em reservar um espaço na memória do micro para armazenar um certo tipo de informações, associando a este espaço um *tipo de dados* e uma *identificação* (rótulo, etiqueta, ou mais usualmente, nome da variável/constante).

Em Pascal declaramos as constantes da seguinte forma:

```
CONST
   <nome_da_constante> = <valor_da_constante> ;
```

Exemplos:

```
CONST
   PI = 3.1415926;
   VERSAO_TP = 7.0;
   WINDOWS = 95;
   PROF_PROG1 = 'Fernando Osório';
```

Em Pascal declaramos as variáveis da seguinte forma:

```
VAR
   <nome_da_variável> : <tipo_da_variável> ;
```

Exemplos:

```
VAR
   Idade: INTEGER;
   Salario: REAL;
   Sexo: CHAR;          { Pode ser  'M' ou 'F' }
   Nome: STRING;
   Qtde_de_Unidade_Vendidas: LONGINT;
    Dia_do_mes: BYTE;
```

Podemos declarar as variáveis sempre imediatamente antes do BEGIN/END. Os programas Pascal tem usualmente uma aparência como a deste exemplo que segue abaixo:

```
PROGRAM exemplo;
CONST                                              { Declaração de constantes }
   PI = 3.1415926;
VAR                                                { Declaração de Variáveis }
   Raio :  REAL;
   Area_da_Circunferencia: REAL;
BEGIN                                              { Comandos do Programa }
   write('Forneça o raio: ');
   readln (Raio);
   Area_da_Circunferencia :=  PI * ( Raio * Raio );          { Area = PI * R^2 = PI * R * R }
   writeln('Área = ',Area_da_Circunferencia);
   readln;
END.
```

2.4. Nomes de Variáveis e Constantes:

Existem algumas regras importantes em relação a escolhas dos nomes à serem dados para as variáveis, constantes e também para o nome do programa (indicado logo após "program"). As regras são as seguintes:

- Começar **sempre** os nomes com uma letra;
- Os nomes começam por uma letra seguida de uma seqüência de letras ('a'..'z', 'A'..'Z'), números ('0'..'9') ou caracteres do tipo sublinhado ('_' = Underscore);
- Um nome de variável pode conter no máximo 63 caracteres;
- Nomes com caracteres em minúsculas ou maiúsculas são considerados iguais independentemente dos caracteres estarem em minúsculas ou maiúsculas. Exemplo: Nome é igual à noME
- **NÃO é permitido** colocar caracteres de **espaço em branco** (' ') dentro de nomes de variáveis/constantes ou nomes de programas! Exemplo: VAR Média Final:Real; <= ERRADO!
- **NÃO é permitido** colocar **acentos e caracteres especiais** em nomes de variáveis, constantes ou programas. Exemplo: VAR Preço-em-R$: Real; <=== ERRADO!
- **NÃO é permitido** colocar nomes que sejam iguais aos comandos (palavras-chave) da linguagem Pascal. Exemplo: VAR End: Integer; <=== ERRADO!

2.5 Usando Variáveis e Constantes: Atribuições e Expressões:

- Atribuições: Atribuir um valor à uma variável, significa colocar um valor na memória que está identificada com o nome especificado. Ao fazer uma atribuição, copiamos um valor, ou o resultado de alguma operação, para dentro da variável, alterando o seu conteúdo.

- Em Pascal a atribuição é feita da seguinte forma:

```
<variável> := <valor>;          <=== Colocar um valor na variável
<variável> := <variável>;       <=== Copiar o conteúdo de uma variável em outra
<variável> := <expressão>;      <=== Colocar o resultado de uma expressão (de um cálculo
                                     matemático, por exemplo) dentro de uma variável
```

Exemplos de atribuições:

(1) Atribuir		(2) Trocar		(3) Somar/Subtr. 1
Media := 8.2;	||	Temporaria := A;	||	A:=A+1;
Media := Media_Final;	||	A:= B;	||	Total:=Total+1;
Media := (Nota1 + Nota2) / 2	||	B:= Temporaria;	||	Aulas:=Aulas-1;

- Expressões: Uma expressão contêm uma ou mais operações aritméticas (ou lógicas) que vão transformar os dados. Uma expressão contêm operadores e operandos, como por exemplo, no cálculo da média entre 2 valores dado pela expressão: "Media := (Valor1 + Valor2)/2; ", temos Valor1 e Valor2 que são os operandos, e '+' e '/' que são os operadores. O resultado final da expressão é calculado e armazenado na variável Media.

Exemplos de expressões:

```
Salario_Bruto:=Total_de_Horas_Trabalhadas * Valor_por_Hora;
Descontos:=Desconto_INSS + Desconto_IR_na_Fonte + Desconto_Faltas_ao_Servico;
Salario:=Salario_Bruto - Descontos;
Preco_em_Reais :=  (Preco_em_Dolar * 2) * Fator_de_Especulacao + Lucro_Anual / 12;
```

- Operadores usados em expressões numéricas (Operadores Aritméticos):

  ** Operadores usados com números inteiros: +, - , *, /, Div, Mod*

  Soma: <inteiro> **+** <inteiro> = <inteiro> Exemplo: Res_Int := VInt1 + VInt2;
  Subtração: <inteiro> **-** <inteiro> = <inteiro> Exemplo: Res_Int := VInt1 - VInt2;
  Multiplicação: <inteiro> ***** <inteiro> = <inteiro> Exemplo: Res_Int := VInt1 * VInt2;
  Divisão: <inteiro> **/** <inteiro> = <real> Exemplo: Res_Real := VInt1 / VInt2;

  Divisão Inteira:<inteiro> **DIV** <inteiro> = <inteiro> Exemplo: Res_Int := VInt1 DIV Vint2;
  Resto da Divisão Inteira: <inteiro> **MOD** <inteiro> = <inteiro> (É a sobra da divisão inteira)

  ** Operadores usados com números reais: +, -, *, /*

  Soma: <real> **+** <real> = <real> Exemplo: Res_Real := VReal1 + VReal2;
  Subtração: <real> **-** <real> = <real> Exemplo: Res_Real := VReal1 - VReal2;
  Multiplicação: <real> ***** <real> = <real> Exemplo: Res_Real := VReal1 * VReal2;
  Divisão: <real> **/** <real> = <real> Exemplo: Res_Real := VReal1 / VReal2;

  ** Operadores (funções) de conversão entre inteiros e reais:*

  - Transformar um Inteiro em Real:
    Variavel_Real := Variavel_Inteira;

  - Transformar um Real em Inteiro:
    Variavel_Inteira := TRUNC (Variavel_Real); {Ignora parte fracionária}
    Variavel_Inteira := ROUND (Variavel_Real); {Arredondamento}

  - Transformar um Real em Real (quebrar um Real):
    Variavel_Real := INT (Variavel_Real); {Guarda só a parte inteira}
    Variavel_Real := FRAC (Variavel_Real); {Guarda só a parte fracionária}
    Variavel_Real := ABS (Variavel_Real); {Ignora o sinal, guarda só a parte absoluta do número => positivo}

  Sendo que assumimos as seguintes variáveis para os exemplos acima:

```
VAR
  VInt1, VInt2: INTEGER;          VReal1,VReal2:REAL;
  Res_Int:INTEGER;                Res_Real:REAL;
  Variavel_Inteira:INTEGER;       Variavel_Real:REAL;
```

- Observação sobre os inteiros: as variáveis inteiras são *números ordinais*, ou seja, podem ser ordenados um após o outro, quando conhecemos um número podemos determinar o seguinte, o que não é possível de se fazer com os números reais. Os dados do tipo ordinal podem usar algumas funções especiais para saber obter o número seguinte (valor+1) e o número anterior (valor-1). As funções de manipulação de ordinais do Pascal são as seguintes:

  Inc (x) = Soma um a variável X. Dec(X) = Diminui uma unidade da variável X.
  Succ(x)= Retorna o sucessor de X (para um valor numérico ordinal isso equivale ao valor +1)
  Pred(x)= Retorna o predecessor de X (para um valor numérico isso equivale ao valor –1)

EXERCÍCIOS – AULA 01

1. Quais das variáveis abaixo possuem nomes válidos para a Linguagem Pascal?

	Válido	Inválido
1.1. Salario_Real_Apos_Total_de_DeduCOES_E_ACRESCIMOS	( )	( )
1.2. Total_em_Dolares	( )	( )
1.3. Real	( )	( )
1.4. Nota GrauA	( )	( )
1.5. @Home	( )	( )
1.6. Web@Home	( )	( )
1.7. Web_At_Home	( )	( )
1.8. Salário1999	( )	( )
1.9. Opção_Inicial	( )	( )
1.10. Fisrt_Option	( )	( )
1.11. Last-Option	( )	( )
1.12. 4Ever	( )	( )

2. Qual o tipo de variável que você declararia para armazenar os seguintes dados:

2.1. Idade: _____
2.2. Total_mensal_de_Faltas_ao_Servico: _____
2.3. Media_Final_na_Disciplina: _____
2.4. Velocidade_do_Veiculo_Multado: _____
2.5. Valor_da_Conta_Telefonica: _____
2.6. Producao_Anual_de_Carros: _____
2.7. Distancia_Percorrida: _____

3. Determinar o resultado das seguintes expressões e trechos de programas dados abaixo:

```
VAR
   A, B, C, Total_Laranjas, Suco1, Suco2, Suco3, Suco4 : INTEGER;
   Nota1, Nota2, Media, X, Y, Z, W : REAL;
```

3.1.
```
Nota1 := 5.0;                                   => Media = ?
Nota2 := 8.0;
Media := Nota1*2 + Nota2 *3 /  5;
```

3.2.
```
A := 6.0;                                       =>  A = ?
B := 4.0;                                           B = ?
C := 2.0;                                           C = ?
A := B+C / 2;
C := A;
B := A;
```

3.3.
```
Total_Laranjas := 9;                            => Suco1, Suco2, Suco3 = ?
Suco1 := Total_Laranjas DIV 2;                     Sobrou laranjas para o Suco3 ?
Suco2 := Suco1;                                    Sobrou laranjas para o Suco4 ?
Suco3 := Total_Laranjas MOD 2;
Suco4 := Total_Laranjas – Suco1 – Suco2 – Suco3;
```

3.4. Qual a atribuição que pode ser feita para adicionar uma unidade à variável Total_Alunos?
Total_Alunos:= _____

3.5. Supondo que o CPMF é igual a 0.5% do valor pago, calcule o valor real gasto ao emitir um cheque:
Valor_Real_Gasto := Valor_Pago + _____

3.6. Qual o valor final armazenado na variável X após a execução dos seguintes comandos:
```
A :=  10;
B :=   A DIV 3;
Y :=  1.6;
X :=  A *  (A MOD B) * (A DIV B) * ROUND(Y) + FRAC(Y);
X := X+1;
```

3.7. Qual o valor final armazenado na variável X após a execução dos seguintes comandos:
```
A := 2.7;
B :=  3.2;
X := A + SQRT(B) / 2 * X;
```

3.8. *QUEBRA-CABEÇA LÓGICO:* Dadas duas variáveis inteiras A e B, nas quais foram armazenados dois valores inteiros, escreva uma seqüência de comandos que realize a troca dos valores entre A e B (A vai receber o valor que estava contido em B e B receberá o valor que estava contido em A). Faça esta troca *SEM O USO DE NENHUMA OUTRA VARIÁVEL* , usando apenas A e B.

3.9. QUEBRA-CABEÇA LÓGICO: Dada uma variável que contém o número de dias do ano, faça uma expressão (uma única atribuição) que coloque em uma variável Fevereiro, o número correto de dias que possui este mês, sabendo que podemos estar considerando ou não um ano bissexto.
Fevereiro := _____

3.10. QUEBRA-CABEÇA LÓGICO: Dado o seguinte trecho de programa abaixo e sabendo-se que ao final de sua execução as variáveis X e Y valiam 14 e 12 respectivamente, determinar quais eram os valores iniciais de X e Y ***antes*** de executar o referido trecho do programa.

```
X := X + Y;
Z := Y / 2 + 3 * X;
X := W * 0.5 + Z + 1 – W / 2;
Y := 6 * Y;
```

Ao final destas 4 linhas X possui armazenado o valor 14 e Y possui o valor 12.
Qual o valor inicial de X e de Y ?

- Se você resolveu todos os exercícios corretamente, parabéns !
- Se você não conseguiu resolver todos os exercícios ou se você errou na solução de alguns dos problemas, atenção... você deverá se esforçar e trabalhar bastante para aprender a programar bem.