Manual da linguagem Gomes

Linguagem de programação desenvolvida
por Dionatan Gomes e Diego Aron Gomes

Apresentamos a seguir o funcionamento da linguagem de programação Gomes. Esta linguagem tem por objetivo a implementação de programas básicos, utilizando-se de dados do tipo double. Algumas considerações importantes:

- Todas as linhas devem ser encerradas com ";".
- Linhas com // serão consideradas comentários, não influenciando no funcionamento do programa.
- Linguagem "case sensitive", diferindo maiúsculas e minúsculas.
- Espaços não influenciam no funcionamento do programa, podendo ser utilizados conforme desejo do programador.
- Estar atento às orientações e exemplos, evitando erros de digitação.
- Utilização de variáveis não declaradas acarretará erro no funcionamento do programa, que não encontrará a variável em seu vetor de variáveis, e se encerrará.

Variáveis

Declaração: As variáveis em Gomes são, como já citado, todas do tipo double. Essas devem ser declaradas antecedidas da palavra *Var, dois pontos, o nome da variável a ser declarada, e ponto e vírgula, como nos exemplos seguintes:*

```
Var:a;
  ou
Var : b;
```

Atribuição: A atribuição segue o estilo da declaração. Tendo a variável a qual se atribuirá valor iniciando a linha, seguida da igualdade e o valor que será colocado nela. Podendo também conter, após a igualdade, uma operação entre dois números, duas variáveis ou entre um número e uma variável, além dos já conhecidos incremento (a++) e decremento (a--). Exemplos:

```
a = 1;
b = a + 2;
c = b - a;
d = 4 * 3;
```

Atribuição com entrada do teclado: Para entrada de valores pelo teclado, deve-se utilizar a palavra Digite. Após ela, entre parênteses, colocar a variável na qual o valor deve ser inserido. Vide abaixo exemplo:

```
Var: a;
Var: b;
Imprime([Digite um valor: ]);
Digite(a);
Imprime([Digite um valor: ]);
Digite(b);
```

Operações

As operações seguem as seguintes regras: devem conter dois elementos, podendo estes ser número ou variável. Executam multiplicação, divisão, adição, subtração e módulo (%). Exemplos, vide abaixo.

```
Var:a;
Var:b;
a=2+5;
a=a+1;
a=a-2;
a=a/2;
a=a*3;
a=a%2;
```

Impressão

A impressão na tela deve ser chamada pela palavra *Imprime* e, entre parênteses, os elementos a serem impressos. Estes podem ser variáveis ou strings, conforme exemplos abaixo. No primeiro, imprime apenas a variável. No segundo, imprime o texto definido pelo usuário entre colchetes, e a variável, definida após a vírgula. Por fim, tem a opção de imprimir apenas o texto, definido entre colchetes.

```
Imprime (a);
Imprime ([texto],a);
Imprime ([texto]);
```

Controles de Fluxo

A linguagem gomes se utiliza de dois controladores de fluxo em sua sintaxe: o condicional Se e o laço Enquanto. Os comandos podem ser utilizados a qualquer momento, em qualquer ordem (respeitando sua sintaxe), aninhados ou não. Segue abaixo a descrição detalhada de cada um.

Controle de Fluxo SE

O controle de fluxo da linguagem deve ser chamado da seguinte forma: utilizando-se a palavra *Inicio.se*, e entre parênteses a comparação a ser feita, podendo conter apenas a comparação entre dois elementos. Estes podem ser números ou variáveis. Em relação aos comparadores, podem ser utilizados a igualdade (==), sinais de maior ou menor (>, <), maior ou igual, menor ou igual (>=, <=) e diferente de (!=). Para finalizar o Se, utiliza-se *Fim.se*. *D*entro do controle de fluxo, todas as funções já apresentadas da linguagem podem ser utilizadas. Temos ainda o Senao.se, que somente será executado caso tenhamos "falso" na primeira condição. Para finalizar o comando se utiliza Fim.senao .

```
Inicio.se(a>b);
    Imprime([Primeiro maior que o segundo]);
Fim.se;
Senao;
    Imprime([Primeiro menor ou igual ao segundo]);
Fim.senao;
```

Controle de fluxo (While)

A linguagem Gomes possui ainda o controle de fluxo while, aqui denominado Enquanto. O funcionamento segue a mesma lógica aprendida em outras linguagens, bem como do comando Inicio.se, já apresentado. Inicia-se a linha com a palavra Enquanto, e entre parênteses colocamos a comparação a ser feita. Os operadores da comparação são os mesmos utilizados no Inicio.se. Para finalizar o escopo do comando enquanto, utiliza-se Fim.enquanto, que retornará para a linha na qual se inicia o comando de repetição. Estando a condição não satisfeita mais, os comandos internos serão ignorados, continuando a execução do código na linha seguinte do Fim.enquanto. Abaixo apresentamos o exemplo de uma implementação do Enquando.

```
Var:a;
Var:b;
a=2;
b=3;
Enquanto(a<=2);
    Imprime([Entrei no primeiro]);
    Enquanto(b<=3);
        Imprime([Entrei no segundo]);
        b++;
        Inicio.se(b==4);
            Imprime([Estou no IF]);
        Fim.se;
    Fim.enquanto;
    a++;
Fim.enquanto;
```

Comandos de saída

Por fim, nossa linguagem possui dois mecanismos de saída. O primeiro, que efetua a saída do programa, é chamado por Fim.do.programa , podendo ser utilizado a qualquer momento, em qualquer lugar do código (dentro de uma condição ou laço, por exemplo). O segundo, chamado por quebra.laco, tem seu uso determinado dentro do laço Enquanto. Ao ser encontrado, este comando sairá do laço em que está inserido. Seguem abaixo exemplos de utilização de ambos.

```
//Programa escrito em linguagem Gomes
//Calcula se o número dado é primo
Var:a;
Var:b;
Var:c;
Var:eh_primo;
Imprime([Digite um numero maior que 2: ]);
Digite(a);
b=2;
Enquanto(b<a);
    c=a%b;
    Inicio.se(c==0);
        eh_primo=1;
        Quebra.laco;
    Fim.se;
    b ++;
Fim.enquanto;
Inicio.se(a<=2);
    Imprime([Numero invalido, mas primo]);
    Fim.do.programa;
Fim.se;
Inicio.se(eh_primo==0);
    Imprime([Numero primo.]);
Fim.se;
Senao
    Imprime([Nao eh primo.]);
Fim.senao;
Fim.do.programa;
```