Exemplos e reflexões

Hoje vamos ter algumas reflexões e exemplos relacionados as variáveis.
Alguns exemplos são deliberadamente apresentados como irreais, pois queremos aprender a lógica e as técnicas relacionadas com as variáveis que podem ser aplicáveis independentemente das necessidades do negócio real e específica.
Será o operador, uma vez capaz de dominar os novos conceitos, aplicá-los à realidade: terá de decidir quando e se usar uma variável, as operações, a lógica mais adequada .....
As variáveis usadas nesses exemplos são todas variáveis comuns de #100 a #149, livremente utilizáveis pelo operador e livre de eventos paralelos.
O uso de outras variáveis do que os comuns é objeto de estudo futuro, porque elas têm um significado particular, não podem ser alterados ou interpretados pelo operador.
Sinta-se livre de usar as variáveis comuns dando-lhe o significado de sua escolha, mas preste atenção às variáveis do sistema ou as locais: é preciso antes uma explicação adequada.
As variáveis não devem derrubar o planejamento mas devem ajudar o operadore a ser mais flexível, escrever programas simples e editável rapidamente.
A definição da variável é inerente à própria palavra: algo que não tem valor constante, mas pode ser alterado para atender às suas necessidades.
Este é o conceito fundamental que nunca devemos esquecer!
Não é necessário escrever programas complicados para usar variáveis, pelo contrário, estas podem e devem ajudar-nos mesmo naqueles trabalho aparentemente simples. Sempre pergunte a si mesmo:
Quais os benefícios obtidos com as variáveis?
E' possível melhorar a legibilidade do programa? (Outros operadores devem instantaneamente compreender o seu significado).
E' possível melhorar sua eficiência?
E' explicado ou interpretado o significado de cada variável?
Certifique-se de ter muitas perguntas quando você está se preparando para usar variáveis: nossa tarefa é aprender o uso, mas também definir as vantagens reais.
O mais simples uso de variáveis é a escolha: estamos em uma encruzilhada e devemos escolher qual caminho tomar.
Sempre que nos encontramos nesta situação, é possivel usar uma variável.
Por exemplo:
-Para criar um perfil de desbaste e acabamento, é inútil escrever duas vezes o mesmo perfil, você vai usar uma variável cujo valor irá alterar automaticamente o valor do raio da ferramenta (#100=0 ->desbaste, #100=1-> acabamento).
-Se você criar uma rotina de controle para quebra de ferramenta, uma variável pode ser conveniente para evitar de parar a usinagem: a ferramenta está quebrado? (#100=0) não! Então continue! Sim! (#100=1) utiliza outra ferramenta e continua o programa.
-Realizamos o looping de uma parte do programa, o valor de uma variável pode nos dizer se temos alcançado o número apropriado de repetições ou não. Por exemplo uma interpolação circular ou helicoidal, #100 chegou a 0? não! continua a repetir. #100 chegou a 0? sim! parar a repetição.
Sempre que temos uma condição de comparação, podemos usar uma variável.
Agora vamos usar uma variável para fazer o looping de uma parte de programa, ou seja, uma sub-rotina.

```
-----
#100=10                 contactor
N10 SUB ROTINA          parte do programa para ser repetito
------
------
#100=#100-1             diminuir o contator
IF[#100NE0]GOTO10       comparação com o valor 0
N20 -----               continuação do programa
```

A #100 é definida para o valor de repetições (10).
É executada a subrotina (bloco N10, este bloco cobre toda a sub-rotina).
A #100 é diminuída, isto significa que houve uma repetição.

O valor do #100 é comparado com 0, porque 0 significa ter feito 10 repetições (o que queríamos).
Se o valor de #100 é diferente de 0 (NE), o programa retorna ao bloco 10 (GOTO10) porque não temo atingido o número desejado de repetições.
Se o valor do #100 é igual a 0, não há retorno e o programa continua linearmente com o bloco N20.
Os passos para executar um looping de uma parte do programa são:
Definir uma variável para o valor exigido para repetições (contactor).
Escrever a parte repetitiva do programa.
Executar a variação do contator.
Fazer uma comparação.
Estas etapas não precisam necessariamente seguir a ordem listada!
E' possível comparar antes da repetição do ciclo, ou aumentar em vez de diminuir o contador...
É importante compreender o conceito lógico, cuja expressão pode ser mutável

exemplos,

```
(1)   N8 #100=0                    (2)   #100=10
      N10 SUB ROTINA                      N10 IF[#100LT#110] GOTO20
      N50 #100=#100+1                     SUB ROTINA
      N52 IF[#100NE10] GOTO10             #100=#100-5
      N54 -----                           GOTO10
                                          N20 ------
```

Então o programador é livre de escolher de acordo com a ordem necessidades do negócio desejado, mantendo a lógica listada acima.
Um bom programador é capaz de escrever qualquer programa com variáveis.
Um excelente programador é capaz mesmo comentar sobre o significado das variaveis (muito importante em programas com múltiplas variáveis).
Assim, em programas complexos é uma boa idéia dar uma explicação sucinta do significado das variáveis usando parênteses. Recomenda-se não ir ao mar com a explicação porque iria prolongar o programa, diminuir a velocidade e perder o seu significado.

```
programa não explicado        programa explicado

-----                         -----
-----                         -----
#100=20                       #100=20 (contator)
#101=0                        #101=0  (desbaste/acabamento)
#102=30                       #102=30 (profundidade Z de usinagem)
#103=1000                     #103=1000 (avançar)
#104=0                        #104=0 (variável de controle)
----                          -----
```

O segundo exemplo é claramente legível, não requer investigação no âmbito do programa para entender o significado. O mais simples programas não exigem explicação alguma se for mantida identificação uniforme, ou seja, tenta-se usar o nome de uma variável sempre com o mesmo significado.
Por exemplo, em uma repetição cíclica de uma sub-rotina sempre se usará a #120 como contactor.
Os exemplos acima escritos são totalmente funcionais e sintaticamente corretos.
Mas devemos considerar alguns aspectos relacionados com o tempo de processamento do CNC.
Quando você executar uma frase GOTO, é procurado o número de seqüência especificada (para frente).
Por esta razão, o pulo para trás leva mais tempo do pulo para frente (é preciso ler o programa inteiro e voltar). Na medida do possível o uso de GOTO deve ser sempre dirigido para frente, e possivelmente, não muito longe do desvio em si.
Nos programas cumprido é sensível o tempo necessário para encontrar o número de seqüência (o CNC modernos não tem esses problemas, já que comentários são ignorados).

Existe uma melhor maneira de realizar a repetição de trechos de programas?
Sim, usando a instrução WHILE.
Como exercício, reescreva o primeiro programa usando WHILE.
(E' preferível parar e procurar a solução!)
O loop WHILE reduz significativamente o tempo de execução do programa porque o CNC não tem que olhar constantemente para a seqüência de números, mas bloqueia a pesquisa para o ciclo repetitivo.
Escrever um programa pode ser feito de várias maneiras, devemos sempre pesar com muito cuidado (especialmente as primeiras vezes) a operação lógica melhor, o ciclo mais apropriado; a experiência e aplicação, mais uma vez, será de grande ajuda.
Um exemplo pouco real do uso de variáveis (M98/M99).

```
G90 X50 Y0
X35.355 Y35.355
X0 Y50
X-35.355 Y35.355
X-50 Y0
X-Y-35.355 35.355
X0 Y-50
X35.355 Y-35.355
```

O programa representa 8 pontos equidistantes distribuídos por uma circunferência de diâmetro 100 (o primeiro ponto a zero grau).
Para realizar centralização, furação, rosqueamento (M10) e alargamento è natural usar M98: cada ferramenta vai chamar a sub-rotina que executa o movimento devido.

```
T1 M6       ferramenta 1, ambiente de trabalho (omiti configurações do trabalho)
M98 P....

T2 M6       ferramenta 2, ambiente de trabalho
M98 P....

T3 M6       ferramenta 3, ambiente de trabalho
M98 P....

T4 M6       ferramenta 4, ambiente de trabalho
M98 P....
```

Tentamos utilizar as variáveis para conseguir o mesmo efeito de M98.
A chamada do subprograma (M98) não é um problema, é necessário definir um número para o bloco inicial N100 G90 X50 Y0.
O retorno da subrotina é mais complexo: é preciso desviar para um bloco específico, mas qual?
E' aqui que intervém a variável, que nos permite desviar para um bloco que ainda não sabemos.
Por exemplo, GOTO #100 desvio para bloco cujo valor é definido em #100.
Apresento o programa com as novas especificações.

```
N100 G90 X50 Y0         (início do subprograma)
X35.355 Y35.355
x0 Y50
X-35.355 y35.355
x-50 y0
X-35.355 Y-35.355
y x0-50
X35.355 Y-35.355
GOTO #100               (retorno da subrotina)
```

O programa começa sempre no bloco 100 e sempre retornará a definição de #100.
Assim, cada ferramenta chama o bloco 100 e irá definir o valor da #100, enquanto a sub-rotina reverte para o valor de #100.

Sumário

```
T1 M6             ferramenta 1, ambiente de trabalho
#100=10           configuração #100
GOTO100           chamada de subprograma

N10 T2 M6         ferramenta 2, ambiente de trabalho
#100=20           configuração #100
GOTO100           chamada de subprograma

N20 T3 M6         ferramenta 3, ambiente de trabalho
#100=30           configuração #100
GOTO100           chamada de subprograma

N30 T4 M6         ferramenta 4, ambiente de trabalho
#100=200          configuração #100

SUB ROTINA
N100 G90 X50 Y0
X35.355 Y35.355
x0 Y50
X-35355 y35.355
x-50 y0
X-Y-35 355 35 355
y x0-50
X35.355 Y-35355
GOTO # 100        voltando da sub-rotina
N200 final
```

Cada ferramenta define a #100 para o retorno do subprograma, então desvia para o mesmo, exceto o último que vai directamente para a subrotina.
Se você comparar este programa com um programa ISO que usa M98 você vai achar que é o mesmo (executar um exemplo real).
Este exemplo puramente académico, mostra que usar variáveis não é derrubar o planejamento mas que se pode escrever um programa de maneiras diferentes.
Os exemplos previamente examinados usavam as variáveis no mais intuitiva e simples: variável como um contator para a repetição de partes do programa.
Reafirmamos a noção de que estamos tentando aprender: o uso de variáveis, a lógicas, então não há o menor interesse da concretude do programa (por enquanto): a aplicação prática virá com a aprendizagem.

Para aprender a lidar com as variáveis é necessário fazer exercícios.
Aqui está uma série de exercícios para a aplicação dos conceitos aprendidos

- Criar um programa que pode fazer o ciclo de chamada de todas as ferramentas.

- Criar um programa que pode fazer o ciclo de chamada de todas as ferramentas, capaz de parar quando encontrar a ferramenta desejada (por exemplo, T7).

-Realizar o perfil, até uma profundidade de 30 mm, de uma peça 100 * 100.
Tendo um processamento de desbaste e um de acabamento: como seria vantajoso usar variáveis? Tentar escrever um programa com múltiplas variáveis.

-Escrever um programa que pode somar ou subtrair 10 números seqüenciais a partir de qualquer número.

-Escrever um programa que pode mover um eixo da máquina com base na quantidade de números seqüenciais de 1 a 20, parando imediatamente depois de passar o número 100.

(((((1 +2) +3) 4) 5) 6) .. se for maior que 100 - pára>

-Você pode reescrever o programa anterior, com um ciclo diferente?

-Escrever um programa para mover qualquer eixo de 0 a 100 (1mm por vez), parando a cada múltiplo de 3 e para cada número em que aparece o 3.

-Fazer três exemplos reais de como usar variáveis.

-É possível repetir uma parte do programa sem a ajuda de um contator?

Podemos fazer um monte de exemplos, mas aqui é melhor você aprender o conceito de variável.
A próxima vez iremos escrever programas paramétricos reais mostrando a verdadeira utilização das variáveis. Agora temos que estudar e experimentar.
Estou aqui para qualquer esclarecimento.
Até.