MÁRCIO FIGUEIREDO MOLLICA

SIMULADOR DE TORNO CNC

Trabalho de Formatura

Escola Politécnica da Universidade
de São Paulo

São Paulo
2001

MÁRCIO FIGUEIREDO MOLLICA

SIMULADOR DE TORNO CNC

Trabalho de conclusão de curso
apresentado à Escola Politécnica da
Universidade de São Paulo

Área de coordenação:
Departamento de Engenharia Mecânica

Orientador:
Prof. João Paulo Marcicano

São Paulo
2001

RESUMO

O objetivo deste trabalho é produzir um programa capaz de gerar o perfil da peça a ser usinada a partir de um código para máquinas de comando numérico. O procedimento adotado para a produção do programa seguiu a seguinte sequência: Identificação inicial das necessidades, especificação das necessidades, definição das especificações técnicas e funcionais, modularização do programa e codificação do programa. Esta última etapa foi feita em 2 partes: Primeiro, a codificação dos algoritmos de interpretação de código e de geração de perfil de peça, utilizando como plataforma um programa de CAD, depois, a codificação final, gerando um aplicativo independente com comandos e interfaces próprias. As principais limitações desta versão do programa são: a possibilidade de se simular apenas códigos para tornos, não é possível fazer simulação dinâmica do processo de usinagem, não há como considerar as particularidades das diferentes ferramentas e não é possível detectar interferências entre a ferramenta e partes da máquina ou da peça.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

1. INTRODUÇÃO

Máquinas de comando numérico são equipamentos de usinagem que são controlados por dispositivo eletrônico capaz de receber informações por meio de entrada própria, compilar estas informações e transmití-las em forma de comando à máquina operatriz, de modo que esta, sem a intervenção do operador, realize as operações na sequência programada. Estas máquinas se diferenciam das convencionais por não necessitarem de acessórios que proporcionem o controle dos movimentos da máquina, tais como gabaritos, cames, limites etc. e mesmo a interferência direta do operador. Estes movimentos são comandados através dos dados de entrada, que determinam os movimentos a serem executados, proporcionando ao equipamento e a peça uma condição bastante favorável, quando comparado com um equipamento convencional, além do que, são maiores as garantias de uniformidade de qualidade de peça para peça e de lote para lote. Além de solucionar usinagem de peças de grande complexidade, as máquinas de controle numérico vieram auxiliar na redução de tempos improdutivos, no posicionamento e retirada da ferramenta de corte, e ainda introduziram grande flexibilidade no processo de fabricação. Assim, o comando numérico é ainda hoje um dos mais dinâmicos processos de fabricação, constituindo um dos maiores desenvolvimentos para a automatização de máquinas operatrizes de usinagem, além de outras aplicações possíveis fora da indústria que utiliza o processo de usinagem.

Chamamos de máquinas CNC as máquinas de comando numérico contínuo, que são uma evolução em relação às máquinas CN, ou máquinas de comando numérico ponto a ponto. Elas se diferenciam pois as máquinas CN são de concepção mais simples, permitindo apenas o posicionamento dos eixos comandados de máquinas dentro do intervalo de precisão e repetibilidade previstos, porém, em movimento rápido e sem uma trajetória pré-determinada e controlada. Já as máquinas CNC permitem definir a trajetória da ferramenta tanto em sua forma quanto em sua velocidade de avanço. Além do CNC, houve outros avanços neste campo, como a introdução de microprocessadores e memórias ao comando numérico, o que permitiu um processamento de informações muito mais eficiente e maior flexibilidade na programação, uma vez que a sequência de comandos é armazenada na memória e é facilmente alterada. No comando numérico distribuído (DNC), um conjunto de

máquinas CNC é conectado a um microcomputador, que controla todos os processos em cada uma das máquinas. Há ainda o comando numérico adaptativo, que é implementado através de um sistema de controle contínuo por realimentação, onde dados do processo são constantemente comparados com dados de entrada, sendo assim possível a correção de eventuais erros.

O comando numérico pode ser aplicado em diferentes máquinas de usinagem. Seus usos mais comuns são: Centros de Torneamento, constituídos de tornos com grande capacidade de remoção de cavaco, que conseguem perfazer todas as operações possíveis em torneamento, como tornear, facear, fazer canais, roscar, contornos, operações internas e externas e muitas outras, com grande precisão e repetibilidade; Máquinas Simples, ou seja, máquinas de funções limitadas e específicas com posicionamento automático e preciso controlado por CN, tais como furadeiras, puncionadoras, soldadoras e outras; Centros de Usinagem, constituídos de máquinas horizontais ou verticais com grande capacidade de remoção de cavaco, capazes de fazer operações de faceamento, mandrilhamento, furação, roscamento, alargamento, operação de abrir canais, rasgos, contornos e muitas outras, com grande precisão e repetibilidade; Sistemas Integrados de Fabricação, constituído de duas ou mais máquinas CNC interligadas entre si com trocadores e transportadores automáticos de peça, máquinas estas que perfazem operações consequentes e simultâneas.

Atualmente são muitas as vantagens que justificam o uso de máquinas CNC. Entre outras, podemos citar:

- Diminuição de mão-de-obra e tempo de execução, já que é um equipamento automático;
- Diminuição do custo ferramental, já que as máquinas CNC dispensam muitos dos acessórios utilizados;
- Maior liberdade para os projetos, uma vez que o CNC permite a usinagem de perfis mais complexos;
- Maior versatilidade de produção;
- Maior fidelidade na especificação, devido à precisão das máquinas;
- Melhor controle de qualidade, devido à maior uniformidade na produção.

Apesar de todas estas vantagens anteriormente apresentadas, o uso de equipamentos CNC introduz algumas dificuldades quanto a sua programação. Normalmente estas máquinas são programadas em linguagem G, uma linguagem desenvolvida especialmente para este tipo de equipamento. Basicamente, um código em linguagem G é composto de uma seqüência de comandos a serem realizados pela máquina operatriz. Estes comandos são compostos de uma letra símbolo, que representa a função deste comando, e um valor numérico, dado entre faixa de valores pré-determinada, que representa o valor de uma grandeza física. A tabela 1 mostra alguns exemplos de comandos para centros de torneamento:

Letra Símbolo	Descrição da função	Formato	Unid
N	Função número de seqüência	xxxx	-
G	Função Preparatória	xx	-
X	Função Posicionamento do eixo transversal	±xxx,xxx	mm
Z	Função posicionamento do eixo longitudinal	±xxx,xxx	mm
I	Função de posicionamento auxiliar medido na direção de "x"	xxx,xxx	mm
K	Função de posicionamento auxiliar medido na direção de "z"	xxx,xxx	mm
R	Função de posicionamento auxiliar da ferramenta em "x"	xxx,xxx	mm
F	Função auxiliar de avanço	xxxx,x	mm/ min
S	Função auxiliar de velocidade de corte	xxxx	rpm
T	Função auxiliar de troca de ferramenta	xxxx	-
M	Função auxiliar miscelânea	xx	-

Tabela I: Exemplos de funções de um centro de torneamento

Na tabela acima podemos ver algumas das principais funções de um centro de torneamento. Cada letra identifica uma função distinta, e essa é seguida de número cujo formato aparece na 3ª coluna. Por exemplo, se o formato é xxx,xxx , significa que o número pode varia de 000,000 a 999,999, sendo a menor divisão 000,001. Por fim cada número é dado em uma unidade. Tomando com exemplo a função de posicionamento no eixo transversal, pode-se ter um comando x 57, que significa posicionar o eixo transversal a 57 mm da origem. Uma seqüência destes comandos simples forma um programa em código G, como no exemplo da tabela 2:

n	g	X	Z	I(r)	Auxiliares
n 10	G90				t1101 m12
n 20	G92	x 72000	z263000		
n 30	G98				s 2000
n 40	G96			r72000	s 150
n 50	G95				m 03
n 60	G01	x 25000	z35000		f0
n 70			z18000		f4000 m08
n 80		x 72000	z263000		f0m09
n 90					t0000m01
n 100	g90	x 72000	z275000		t2102 m11
n 110	g92				
n 120	g98			r72000	s760
n 130	g96				s80
n 140	g95	x 24826	z34936		m03
n 150	g01		z18000		f0
n 160	g33			k500	m08
n 170	g01	x 26000			
n 180			z34910		
n 190		x 24754			
n 20	g33		z18000	k500	
n 210	g01	x 26000			
n 220			z34890		
n 230		x 24700			
n 240	g33		z18000	k500	
n 250	g01	x 26000			
n 260			z34874		
n 270		x 24654			
n 280	g33		z18000	k500	
n 290	g01	x 72000			m09
n 300			z275000		f0 t0000

Tabela II: Exemplo de programa em código G

Podemos ver pelo exemplo da tabela 2 que apesar de os comandos serem bastante simples, há uma certa dificuldade em se programar em código G, principalmente quando se desejar usinar uma peça mais complexa. A dificuldade está principalmente em se determinar a seqüência de operações necessárias para a usinagem, ou seja, determinar quais operações e quando serão aplicadas. Além disso, é bastante difícil a realização de testes, ou seja, saber se determinado código vai dar origem a peça desejada. Por fim, deve-se ter cuidado adicional com interferências entre ferramenta e peça, o que é difícil de se determinar durante a programação.

2. ESTUDO DE VIABILIDADE

2.1 Identificação das Necessidades

O primeiro passo para se iniciar um projeto é identificar claramente e isolar a necessidade que ele deve atender. Assim, no caso do uso de máquinas CNC, forram identificadas as seguintes as seguintes necessidades:

- Realização de testes com códigos em linguagem G, ou seja, verificar de forma antecipada a geometria da peça a ser fabricada. Essa é uma necessidade bastante freqüente pois, para peças mais complexas, é difícil saber o perfil da peça gerada pelo código sem que ela seja usinada, o que pode causar desperdício de tempo e de material, além de dificultar a programação da máquina. Os testes são grande importância na fabricação pois é quando pode-se determinar erros de programação e projeto, evitando que estes afetem o resultado final.

- Identificação de possíveis intereferências. Também é bastante comum no uso de equipamentos de usinagem ocorrer interferências entre a peça e a ferramenta. Em processos mais complexos de fabricação nem sempre é possível evitar este tipo de problema, que dificilmente é identificado na fase de programação.

- Treinamento de operadores. Com uso em grande escala de equipamentos de controle numérico, é grande a demanda por técnicos qualificados para operarem as máquinas. Para suprir esta demanda, é necessário treinar muitos técnicos, mas nem sempre há disponibilidade de máquinas para isso.

2.2 Especificação Técnica da Necessidade

A maneira mais eficiente de se especificar uma necessidade é tratá-la como um sistema fechado, onde se conhece apenas as entradas e saídas desejadas. Após a análise das necessidades, pode-se identificar as entradas e saídas do sistema. Neste projeto, a necessidade é um sistema cuja entrada é um código em linguagem G, e a saída é um modelo da peça a ser produzida.

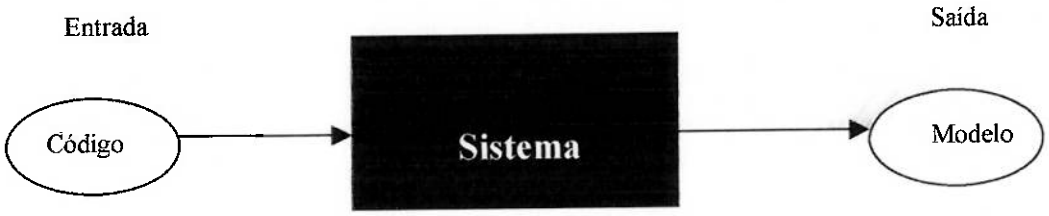

Figura 1 : Especificação da Necessidade

2.3 Síntese de Soluções

Para este problema, as necessidades encaminharam o projeto para uma única solução, que é um software simulador de máquina CNC. Não houve necessidade de se gerar várias soluções alternativas pois sabe-se que um software é um sistema perfeitamente capaz de atender as necessidades do projeto, ou seja, é capaz de ler um código como entrada e gerar um modelo como saída, o que de antemão elimina a necessidade de análise de viabilidade técnica do projeto.

3. OBJETIVOS E MOTIVAÇÃO

O objetivo deste trabalho é produzir um programa de computador didático capaz de simular um torno CNC. Este programa receberá como entrada comandos em linguagem G, que é a linguagem usada para programação de máquinas CNC, e terá como saída a geometria da peça que seria produzida num torno CNC através dos mesmos comandos, além de identificar eventuais interferências entre a peça gerada e as ferramentas.

Simuladores CNC são bastante usados na manufatura, porque permitem a verificação antecipada do perfil da peça a ser produzida, podendo-se assim testar o programa feito sem dispêndio de tempo e material. Além disso, pode ser usado para treinamento de operadores e programadores de máquinas CNC, uma vez que estes podem verificar o resultado de seus programas. Devido ao grande uso deste tipo de simulador, o que evidencia sua necessidade, já existem no mercado várias opções de programas similares.

No caso deste simulador, seu objetivo é didático, e deverá ser utilizado em laboratório de processos de fabricação para que os alunos possam aprender a programar máquinas CNC e testar programas antes de aplicá-los na fabricação de peças. Como este programa terá seu código aberto e devidamente documentado, será possível para alunos e professores entenderem a interpretação dos comandos, fazerem alterações ou mesmo expandir suas funções, reforçando seu caráter didático. Esta é a principal razão para se optar pela implementação do simulador ao invés de se usar um programa comercial, de código fechado.

4. ESPECIFICAÇÕES TÉCNICAS E FUNCIONAIS

4.1 Linguagem de Programação

A linguagem de programação escolhida para codificar o programa foi o *Visual Basic*, pois, além de ter todos os recursos para este projeto, é amplamente conhecida e há grande facilidade de se obter informações e suporte.

4.2 Entradas e Saídas

- Entradas: Código em linguagem G dado por um arquivo de texto ou por um editor de código, que será embutido no programa para facilitar a interface com o usuário.

- Saídas: Perfil da peça gerada pelo código.

4.3 Blocos Funcionais

São blocos de funções de baixo nível a serem implementadas para viablizarem o simulador, agrupadas de acordo com sua funcionalidade. Os principais blocos serão:

4.4 Interface com Usuário

Bloco presente na entrada e na saída do sistema, responsável pela leitura de comandos do usuário e pela monitoração do sistema.

4.5 Interpretação do Código

Bloco que interpreta instruções em linguagem G e passa comandos para o simulador.

4.6 Simulação

Bloco que realiza cada um dos passos dados pelo código. Atua sobre o modelo inicial da peça, transformando-o através de comandos. Permite simulação passo a passo.

4.7 Cálculos

Bloco com funções que realizam os cálculos necessários para a simulação. Realiza cálculo de posicionamento, em coordenadas relativas e absolutas e cálculo de interpolações.

4.8 Visualização

Bloco que permite o usuário ver o modelo da peça em produção.

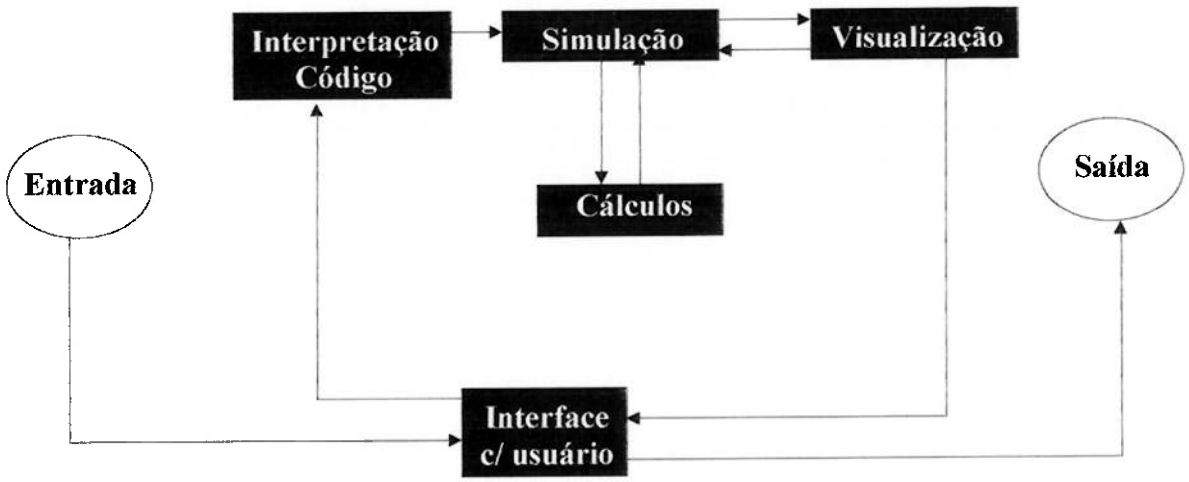

Figura 2 : Interação entre os blocos

5. APRESENTAÇÃO DO PROGRAMA

A interface programa simulador de torno CNC consiste basicamente de 2 blocos fundamentais: o editor de código e modelo da peça. O editor de código é o ambiente onde o usuário poderá introduzir o código da peça desejada, alterá-lo, salvá-lo em um arquivo e iniciar uma simulação. O modelo da peça é uma janela que mostrará o perfil da peça gerada pelo código.

5.1 Editor de Código

O editor de código é uma janela que possui uma planilha onde deverão ser inseridos os comandos desejados. Cada linha é uma instrução do programa e cada coluna é uma das funções apresentadas na tabela 1 (ver tabela 1). O formato dos números inseridos deve seguir os formatos especificados também na tabela 1. O funcionamento deste editor é análogo ao funcionamento de uma planilha eletrônica, como o Excel. As funções de edição permitidas são:

- Abrir: Abrir um arquivo gravado em um disco. Abre uma janela que permite explorar os discos e diretórios disponíveis. Tecla de atalho: "Ctrl + A".
- Salvar: Salva um arquivo em disco. Se arquivo editado já tiver nome, o comando apenas salva em seu local, se não, abre a janela "Salvar Como". Tecla de atalho: "Ctrl + S"
- Recortar: Copia o conteúdo selecionado para a área de transferência e limpa as células selecionadas. Tecla de atalho: "Ctrl + X".
- Copiar: Copia o conteúdo selecionado para a área de transferência e mantem as células selecionadas. Tecla de atalho: "Ctrl + C".
- Colar: Insere o conteúdo da área de transferência nas células selecionadas. Tecla de atalho: "Ctrl + V".
- Desfazer: Desfaz última ação realizada. Pode voltar até 50 passos realizados. Tecla de atalho: "Ctrl + Z".
- Refazer: Caso alguma ação tenha sido desfeita, poderá ser refeita através deste comando. Tecla de atalho: "Ctrl + F".
- Excluir: Exclui a linha selecionada. Tecla de atalho: "Ctrl + Del".

- Inserir acima: Insere uma linha vazia acima da célula selecionada. Tecla de atalho: "Ctrl + Insert".
- Inserir abaixo: Insere uma linha vazia abaixo da célula selecionada. Tecla de atalho: "Ctrl + Shift + Insert".
- Desenhar: Gera o modelo da peça a partir do código.

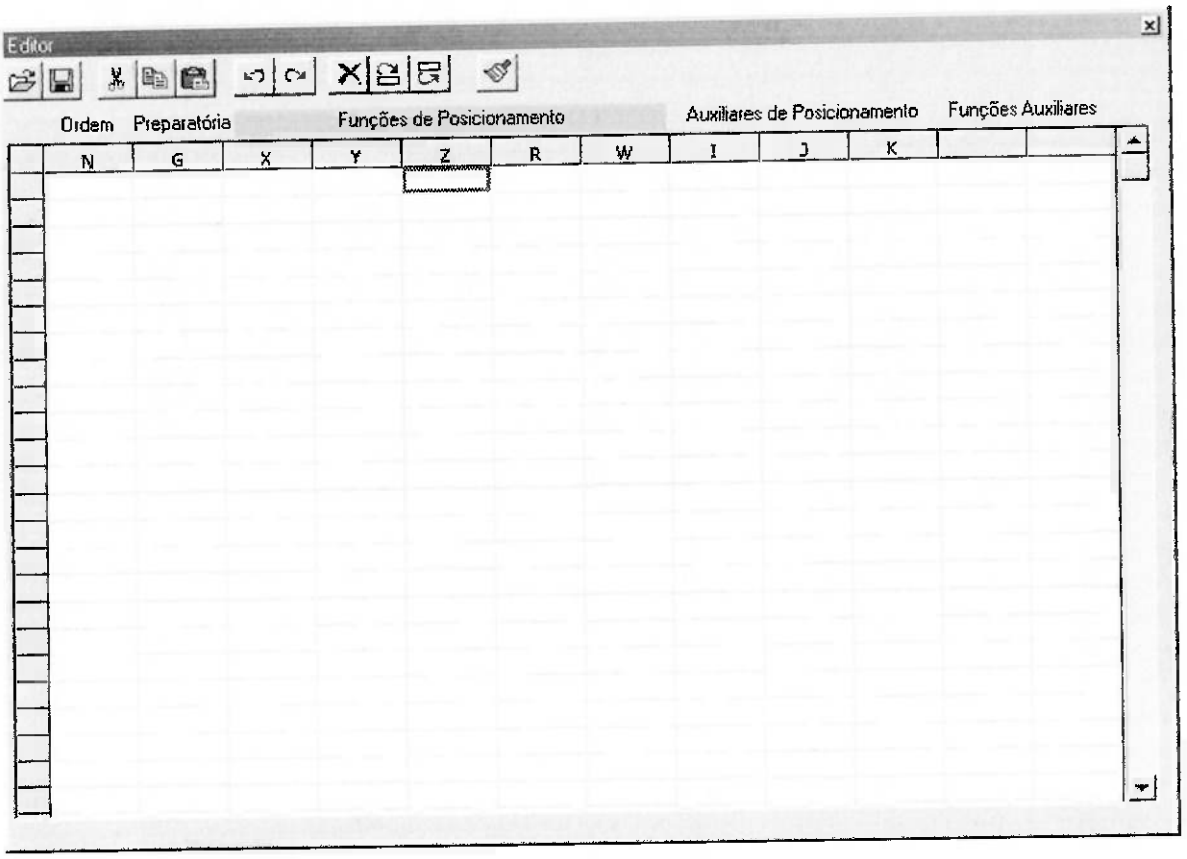

Figura 3: Janela do Editor de Código

5.2 Modelo da peça

Janela que exibe o perfil da peça gerada pela simulação do código editado no editor. As funções que podem ser usadas são:

- Função de sequência "n": Número de sequência da instrução.
- Função preparatória "g00": Posicionamento rápido da ferramenta.
- Função preparatória "g01": Interpolação linear.
- Função preparatória "g02": Interpolação circular em sentido horário.
- Função preparatória "g03": Interpolação circular em sentido anti-horário

- Função preparatória "g04": Permanência
- Função preparatória "g33": Modo de roscar
- Função preparatória "g90": Programação em sistema absoluto de coordenadas.
- Função preparatória "g91": Programação em sistema relativo de coordenadas.
- Função preparatória "g92": Define a origem do sistema de coordenadas.
- Função preparatória "g94": Define velocidade de avanço em m/mim.
- Função preparatória "g95": Define velocidade de avanço em mm/rot.
- Função preparatória "g96": Define velocidade de corte em mm/rot.
- Função preparatória "g97": Define velocidade de corte em rpm.
- Função preparatória "g98": Define velocidade máxima de corte.

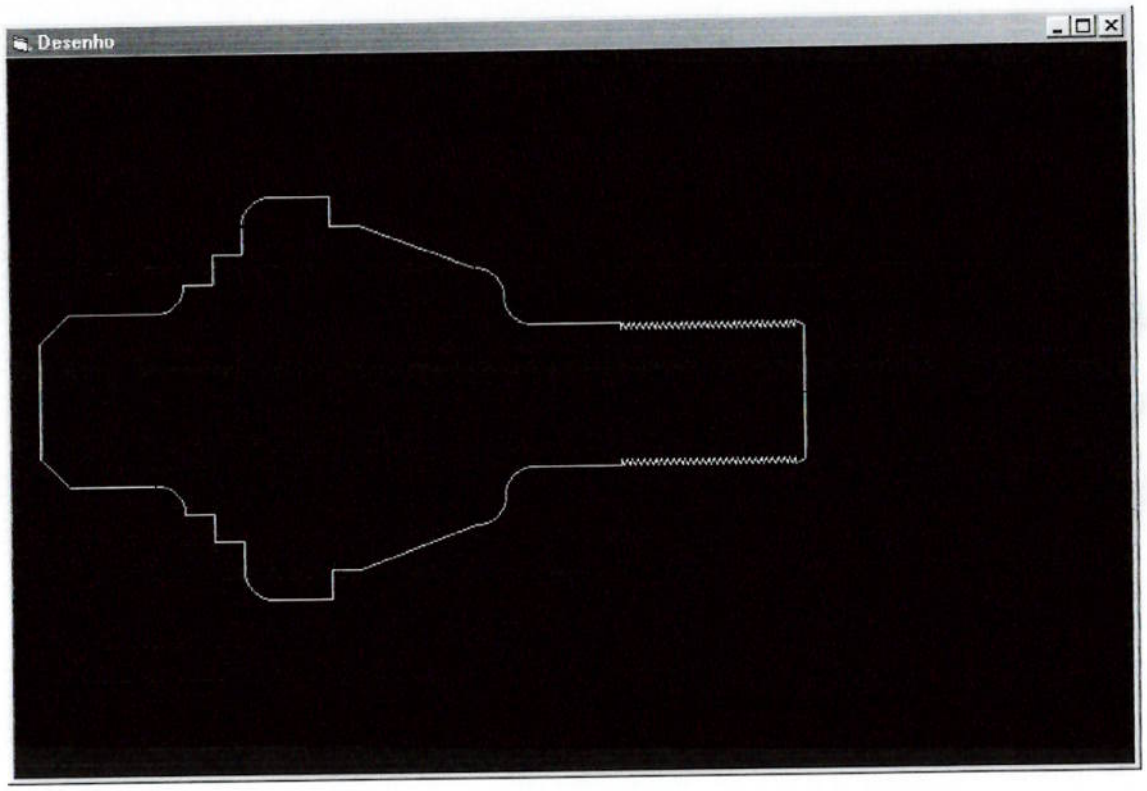

Figura 4: Janela do modelo da peça

6. CONCLUSÕES

O programa apresentado acima possui os requisitos básicos exigidos para uma simulação de código de usinagem por comando numérico, ou seja, é capaz de interpretar uma sequência de comandos comuns e gerar um modelo da peça a ser usinada. Assim, este programa atinge o objetivo proposto inicialmente, que era gerar um módulo basico de simulação de comando numérico. A partir deste módulo, é possível implementar outras funções muito importantes e úteis para um simulador como verificação de interferência da ferramenta, módulo de auxílio de cálculo de velocidades de avanço e de corte e simulação com diferentes tipos de ferramentas, tornando-o mais completo.

7. BIBLIOGRAFIA

|1| Machado, A. Comando Numérico Aplicado às Máquinas-Ferramentas, 4ª ed., Ícone, São Paulo, 1990.

|2| Gibbs, D. e Crandell, T.M. Na introduction to CNC machining and programming, 6ª ed., Industrial, New York, 1991.

|3| Childs, J.J. Principals of Numerical Control, 7ª ed.,Industrial, New York, 1965.

|4| Sava, M. Computer Numerical Control Programming, 7ª ed., Prentice Hall, Englewoods Cliffs, 1990.

|5| Kaminski, P.C. Desenvolvendo Produtos com Planejamento, Criatividade e Qualidade, 1ª ed., Livros Técnicos e Científicos, Rio de janeiro, 2000.

8. ANEXOS

Código Fonte do Programa

Organização dos módulos:

Formulários:

Form1
Form2
Form2
Form4
MDIForm1

Módulos:

Module1
Module2

Módulos de Classe:

Ação
Ações

Form1

```
Private Sub Form_Load()

End Sub
```

Form2

```
Dim Mudou_Célula As Boolean
Dim Conteúdo_Anterior, Intr As String
Dim Fazer, Desfazer As Ações
Dim aux As Ação
Dim Dec As Boolean
Dim imgX As ListImage

Private Sub Form_Load()
   Set Fazer = New Ações
   Set Desfazer = New Ações
   With MSFlexGrid1
      .Row = 0
      .Col = 1
      .Text = "N"
   End With
   Label1(0).BackColor = &H8000000C
   For I = 1 To MSFlexGrid1.Cols - 1
      With MSFlexGrid1
         .ColWidth(I) = 800
         .ColAlignment(I) = 4
         .Col = I
         .CellAlignment = 5
      End With
   Next
   With MSFlexGrid1

      .ColWidth(0) = 300
      .RowHeight(0) = 250

      .Row = 0
      .Col = 2
      .Text = "G"

      .Row = 0
      .Col = 3
      .Text = "X"

      .Row = 0
      .Col = 4
      .Text = "Y"

      .Row = 0
      .Col = 5
      .Text = "Z"

      .Row = 0
      .Col = 6
      .Text = "R"

      .Row = 0
      .Col = 7
      .Text = "W"

      .Row = 0
      .Col = 8
      .Text = "I"

      .Row = 0
      .Col = 9
      .Text = "J"

      .Row = 0
      .Col = 10
      .Text = "K"

      .Row = 1
      .Col = 1
   End With
```

```
        Toolbar1.ImageList = ImageList1
        Toolbar2.ImageList = ImageList1
        Toolbar3.ImageList = ImageList1
        Toolbar4.ImageList = ImageList1
        Toolbar5.ImageList = ImageList1

        Toolbar1.Buttons(1).Image = "Abrir"
        Toolbar1.Buttons(2).Image = "Salvar"
        Toolbar2.Buttons(1).Image = "Recortar"
        Toolbar2.Buttons(2).Image = "Copiar"
        Toolbar2.Buttons(3).Image = "Colar"
        Toolbar3.Buttons(1).Image = "Desfazer"
        Toolbar3.Buttons(2).Image = "Refazer"
        Toolbar4.Buttons(1).Image = "Excluir"
        Toolbar4.Buttons(2).Image = "Acima"
        Toolbar4.Buttons(3).Image = "Abaixo"
        Toolbar5.Buttons(1).Image = "Desenhar"

        NovoArquivo = True

        SistCoord = Absoluto

        OrigemSist_x = 0

        OrigemSist_z = 0

        VelAvanço = mm_min

        VelCorte = m_min

        MaxVelCorte = False

    End Sub

    Private Sub MSFlexGrid1_KeyDown(KeyCode As Integer, Shift As Integer)
        'Enter - Ir para célula abaixo
        If (KeyCode = 13) And (MSFlexGrid1.Row < MSFlexGrid1.Rows - 1) Then
            MSFlexGrid1.Row = MSFlexGrid1.Row + 1
            Mudou_Célula = True
        End If

        'Esc - Cancelar edição na célula
        If (KeyCode = 27) And (Mudou_Célula = False) Then
            MSFlexGrid1.Text = Conteúdo_Anterior
        End If

        'Del - Excluir conteúdo da célula
        If (KeyCode = 46) And (Shift <> vbCtrlMask) Then
            Set aux = Desfazer.Add(MSFlexGrid1.Row, MSFlexGrid1.Col, MSFlexGrid1.RowSel, MSFlexGrid1.ColSel,
AçãoDeletarCélula, MSFlexGrid1.Clip)
            MSFlexGrid1.Text = ""
        End If

        'BackSpace - Exclui último caracter em uma célula
        If (KeyCode = 8) And (Mudou_Célula = False) Then
            If Len(MSFlexGrid1.Text) >= 1 Then
                MSFlexGrid1.Text = Mid(MSFlexGrid1.Text, 1, Len(MSFlexGrid1.Text) - 1)
            End If
        End If

        'Tab - Ir para célula ao lado
        If (KeyCode = 9) And (MSFlexGrid1.Col < MSFlexGrid1.Cols - 1) Then
            MSFlexGrid1.Col = MSFlexGrid1.Col + 1
            Mudou_Célula = True
        End If

        'Ctrl+C - Copiar seleção
        If (KeyCode = 67) And (Shift = vbCtrlMask) Then
            Set aux = Desfazer.Add(MSFlexGrid1.Row, MSFlexGrid1.Col, MSFlexGrid1.RowSel, MSFlexGrid1.ColSel,
AçãoCopiar, Clipboard.GetText)
            Clipboard.Clear
            Clipboard.SetText MSFlexGrid1.Clip
        End If
```

```
        'Ctrl+V - Colar conteúdo da área de transferência
        If (KeyCode = 86) And (Shift = vbCtrlMask) Then
          Texto = Clipboard.GetText
          For I = 1 To Len(Texto)
            If Asc(Mid(Texto, I, 1)) = 13 Then
              linhas = linhas + 1
            End If
            If (Asc(Mid(Texto, I, 1)) = 9) And (linhas = 0) Then
              colunas = colunas + 1
            End If
          Next
          MSFlexGrid1.ColSel = MSFlexGrid1.Col + colunas
          MSFlexGrid1.RowSel = MSFlexGrid1.Row + linhas
          Set aux = Desfazer.Add(MSFlexGrid1.Row, MSFlexGrid1.Col, MSFlexGrid1.RowSel, MSFlexGrid1.ColSel,
AçãoColar, MSFlexGrid1.Clip)
          MSFlexGrid1.Clip = Clipboard.GetText
        End If

        'Ctrl+X - Recortar seleção
        If (KeyCode = 88) And (Shift = vbCtrlMask) Then
          Set aux = Desfazer.Add(MSFlexGrid1.Row, MSFlexGrid1.Col, MSFlexGrid1.RowSel, MSFlexGrid1.ColSel,
AçãoRecortar, MSFlexGrid1.Clip)
          Clipboard.Clear
          Clipboard.SetText MSFlexGrid1.Clip
          MSFlexGrid1.Text = ""
        End If

        'Ctrl+Insert - Inserir linhas acima
        If (KeyCode = 45) And (Shift = vbCtrlMask) Then
          Set aux = Desfazer.Add(MSFlexGrid1.Row, MSFlexGrid1.Col, MSFlexGrid1.RowSel, MSFlexGrid1.ColSel,
AçãoExcluirLinha, "")
          Linha = MSFlexGrid1.Row
          Coluna = MSFlexGrid1.Col
          With MSFlexGrid1
            .BackColorSel = &H80000005
            .FocusRect = flexFocusNone
            .Rows = MSFlexGrid1.Rows + 1
            .Col = 1
            .RowSel = MSFlexGrid1.Rows - 2
            .ColSel = MSFlexGrid1.Cols - 1
          End With
          Texto = MSFlexGrid1.Clip
          With MSFlexGrid1
            .Row = MSFlexGrid1.Row + 1
            .RowSel = MSFlexGrid1.Rows - 1
            .ColSel = MSFlexGrid1.Cols - 1
            .Clip = Texto
            .Row = Linha
            .RowSel = Linha
            .Col = 1
            .ColSel = MSFlexGrid1.Cols - 1
            .Text = ""
            .Col = Coluna
            .ColSel = Coluna
            .BackColorSel = &H80000008
            .FocusRect = flexFocusHeavy
          End With
          Atualiza_Rótulos_Colunas
        End If

        'Ctrl+Del - Excluir linhas
        If (KeyCode = 46) And (Shift = vbCtrlMask) And (MSFlexGrid1.Rows > 2) Then
          Linha = MSFlexGrid1.Row
          Coluna = MSFlexGrid1.Col
          LinhaSel = MSFlexGrid1.RowSel
          ColunaSel = MSFlexGrid1.ColSel
          With MSFlexGrid1
            .BackColorSel = &H80000005
            .FocusRect = flexFocusNone
            .RowSel = MSFlexGrid1.Row
            .Col = 1
            .ColSel = MSFlexGrid1.Cols - 1
```

```
            End With
            Valor = MSFlexGrid1.Clip
            With MSFlexGrid1
               .BackColorSel = &H80000008
               .FocusRect = flexFocusHeavy
               .Row = Linha
               .Col = Coluna
               .RowSel = LinhaSel
               .ColSel = ColunaSel
            End With
            Set aux = Desfazer.Add(MSFlexGrid1.Row, MSFlexGrid1.Col, MSFlexGrid1.RowSel, MSFlexGrid1.ColSel,
AçãoInserirLinhaAcima, Valor)
            Linha = MSFlexGrid1.Row
            Coluna = MSFlexGrid1.Col
            MSFlexGrid1.BackColorSel = &H80000005
            MSFlexGrid1.FocusRect = flexFocusNone
            If MSFlexGrid1.Row <> MSFlexGrid1.Rows - 1 Then
               MSFlexGrid1.Row = MSFlexGrid1.Row + 1
               MSFlexGrid1.Col = 1
               MSFlexGrid1.RowSel = MSFlexGrid1.Rows - 1
               MSFlexGrid1.ColSel = MSFlexGrid1.Cols - 1
               Texto = MSFlexGrid1.Clip
               MSFlexGrid1.Row = MSFlexGrid1.Row - 1
               MSFlexGrid1.RowSel = MSFlexGrid1.Rows - 2
               MSFlexGrid1.ColSel = MSFlexGrid1.Cols - 1
               MSFlexGrid1.Clip = Texto
               MSFlexGrid1.Rows = MSFlexGrid1.Rows - 1
               MSFlexGrid1.Col = Coluna
               MSFlexGrid1.ColSel = Coluna
               MSFlexGrid1.BackColorSel = &H80000008
            Else
               MSFlexGrid1.Rows = MSFlexGrid1.Rows - 1
            End If
            MSFlexGrid1.FocusRect = flexFocusHeavy
            Atualiza_Rótulos_Colunas
         End If

         'Ctrl+Insert - Inserir linhas abaixo
         If (KeyCode = 45) And (Shift = vbCtrlMask + vbShiftMask) Then
            Set aux = Desfazer.Add(MSFlexGrid1.Row + 1, MSFlexGrid1.Col, MSFlexGrid1.RowSel,
MSFlexGrid1.ColSel, AçãoExcluirLinha, "")
            Linha = MSFlexGrid1.Row
            Coluna = MSFlexGrid1.Col
            MSFlexGrid1.FocusRect = flexFocusNone
            MSFlexGrid1.BackColorSel = &H80000005
            MSFlexGrid1.Rows = MSFlexGrid1.Rows + 1
            MSFlexGrid1.Row = MSFlexGrid1.Row + 1
            MSFlexGrid1.Col = 1
            MSFlexGrid1.RowSel = MSFlexGrid1.Rows - 2
            MSFlexGrid1.ColSel = MSFlexGrid1.Cols - 1
            Texto = MSFlexGrid1.Clip
            If MSFlexGrid1.Row < MSFlexGrid1.Rows - 1 Then
               MSFlexGrid1.Row = MSFlexGrid1.Row + 1
            End If
            With MSFlexGrid1
               .RowSel = MSFlexGrid1.Rows - 1
               .ColSel = MSFlexGrid1.Cols - 1
               .Clip = Texto
               .Row = Linha + 1
               .RowSel = Linha + 1
               .Col = 1
               .ColSel = MSFlexGrid1.Cols - 1
               .Text = ""
               .Row = Linha
               .RowSel = Linha
               .Col = Coluna
               .ColSel = Coluna
               .BackColorSel = &H80000008
               .FocusRect = flexFocusHeavy
            End With
         End If

         'Ctrl+Z - Desfazer Ação
```

```
        If (KeyCode = 90) And (Shift = vbCtrlMask) Then
          If Desfazer.Count > 0 Then
            Row = MSFlexGrid1.Row
            Col = MSFlexGrid1.Col
            RowSel = MSFlexGrid1.RowSel
            ColSel = MSFlexGrid1.ColSel
            Set aux = Fazer.Add(0, 0, 0, 0, AçãoNula, "")
            With Fazer(Fazer.Count)
              .Row = Desfazer(Desfazer.Count).Row
              .Col = Desfazer(Desfazer.Count).Col
              .RowSel = Desfazer(Desfazer.Count).RowSel
              .ColSel = Desfazer(Desfazer.Count).ColSel
              If (Desfazer(Desfazer.Count).Tipo = AçãoEditarCélula) Or (Desfazer(Desfazer.Count).Tipo =
AçãoDeletarCélula) Or (Desfazer(Desfazer.Count).Tipo = AçãoNula) Then
                .Tipo = Desfazer(Desfazer.Count).Tipo
              End If
              If (Desfazer(Desfazer.Count).Tipo = AçãoExcluirLinha) Then
                .Tipo = AçãoInserirLinhaAcima
              End If
              If (Desfazer(Desfazer.Count).Tipo = AçãoInserirLinhaAcima) Or (Desfazer(Desfazer.Count).Tipo =
AçãoInserirLinhaAbaixo) Then
                .Tipo = AçãoExcluirLinha
              End If
              If (Desfazer(Desfazer.Count).Tipo = AçãoCopiar) Then
                .Tipo = AçãoRecopiar
              End If
              If (Desfazer(Desfazer.Count).Tipo = AçãoColar) Then
                .Tipo = AçãoRecolar
              End If
              If (Desfazer(Desfazer.Count).Tipo = AçãoRecortar) Then
                .Tipo = AçãoRerrecortar
              End If
              With MSFlexGrid1
                .FocusRect = flexFocusNone
                .BackColorSel = &H80000005
                .Row = Desfazer(Desfazer.Count).Row
                .Col = Desfazer(Desfazer.Count).Col
                .RowSel = Desfazer(Desfazer.Count).RowSel
                .ColSel = Desfazer(Desfazer.Count).ColSel
              End With
              .Valor = MSFlexGrid1.Clip
              With MSFlexGrid1
                .Row = Row
                .Col = Col
                .RowSel = RowSel
                .ColSel = ColSel
                .BackColorSel = &H80000008
                .FocusRect = flexFocusHeavy
              End With
            End With
            Set aux = Desfazer(Desfazer.Count)
            Desfazer(Desfazer.Count).Fazer
            Desfazer.Remove (Desfazer.Count)
          End If
        End If

        'Ctrl+F - Refazer Ação
        If (KeyCode = 70) And (Shift = vbCtrlMask) Then
          If Fazer.Count > 0 Then
            Row = MSFlexGrid1.Row
            Col = MSFlexGrid1.Col
            RowSel = MSFlexGrid1.RowSel
            ColSel = MSFlexGrid1.ColSel
            Set aux = Desfazer.Add(0, 0, 0, 0, AçãoNula, "")
            With Desfazer(Desfazer.Count)
              .Row = Fazer(Fazer.Count).Row
              .Col = Fazer(Fazer.Count).Col
              .RowSel = Fazer(Fazer.Count).RowSel
              .ColSel = Fazer(Fazer.Count).ColSel
              .Tipo = Fazer(Fazer.Count).Tipo
              With MSFlexGrid1
                .FocusRect = flexFocusNone
                .BackColorSel = &H80000005
```

```
                .Row = Fazer(Fazer.Count).Row
                .Col = Fazer(Fazer.Count).Col
                .RowSel = Fazer(Fazer.Count).RowSel
                .ColSel = Fazer(Fazer.Count).ColSel
              End With
              .Valor = MSFlexGrid1.Clip
              With MSFlexGrid1
                .Row = Row
                .Col = Col
                .RowSel = RowSel
                .ColSel = ColSel
                .BackColorSel = &H80000008
                .FocusRect = flexFocusHeavy
              End With
            End With
            Fazer(Fazer.Count).Fazer
            Fazer.Remove (Fazer.Count)
          End If
        End If

        'Ctrl+A - Abrir arquivo
        If (KeyCode = 65) And (Shift = vbCtrlMask) Then
          Form3.Show
        End If

        'Ctrl+S - Salvar arquivo
        If (KeyCode = 83) And (Shift = vbCtrlMask) Then
          If NovoArquivo Then
            Form4.Show
          End If
          Salvar (NomeArquivo)
        End If

        Atualiza_Rótulos_Colunas
        'MSFlexGrid1.Text = KeyCode
      End Sub

      Private Sub MSFlexGrid1_KeyPress(KeyAscii As Integer)

      'Edição de uma célula
      If (KeyAscii > 31) And (KeyAscii <> 10) And (KeyAscii <> 127) Then
        If Mudou_Célula Then
          Set aux = Desfazer.Add(MSFlexGrid1.Row, MSFlexGrid1.Col, MSFlexGrid1.RowSel, MSFlexGrid1.ColSel,
AçãoEditarCélula, MSFlexGrid1.Text)
          Conteúdo_Anterior = MSFlexGrid1.Text
          MSFlexGrid1.Text = ""
          Mudou_Célula = False
          Dec = False
        End If
        If (MSFlexGrid1.Col = 1) And (KeyAscii >= 48) And (KeyAscii <= 57) And (Len(MSFlexGrid1.Text) < 3) Then
          MSFlexGrid1.Text = MSFlexGrid1.Text & Chr(KeyAscii)
        End If
        If (MSFlexGrid1.Col = 2) And (KeyAscii >= 48) And (KeyAscii <= 57) And (Len(MSFlexGrid1.Text) < 2) Then
          MSFlexGrid1.Text = MSFlexGrid1.Text & Chr(KeyAscii)
        End If
        If ((MSFlexGrid1.Col = 3) Or (MSFlexGrid1.Col = 4) Or (MSFlexGrid1.Col = 5) Or (MSFlexGrid1.Col = 6) Or
(MSFlexGrid1.Col = 7)) And (((KeyAscii >= 48) And (KeyAscii <= 57)) Or ((KeyAscii >= 44) And (KeyAscii <= 46)))
And ((Len(MSFlexGrid1.Text) < 7) Or ((Len(MSFlexGrid1.Text) < 8) And (Mid(MSFlexGrid1.Text, 1, 1) = "-"))) Then
          If (Dec) And (KeyAscii <> 44) And (Len(MSFlexGrid1.Text) - Len(Intr) < 3) Then
            MSFlexGrid1.Text = MSFlexGrid1.Text & Chr(KeyAscii)
          End If
          If (Not Dec And KeyAscii = 44) Then
            Dec = True
            MSFlexGrid1.Text = MSFlexGrid1.Text & Chr(KeyAscii)
            Intr = MSFlexGrid1.Text
          End If
          If Not Dec Then
            MSFlexGrid1.Text = MSFlexGrid1.Text & Chr(KeyAscii)
            If ((Len(MSFlexGrid1.Text) = 3) And (Mid(MSFlexGrid1.Text, 1, 1) <> "-")) Or ((Len(MSFlexGrid1.Text)
= 4) And (Mid(MSFlexGrid1.Text, 1, 1) = "-")) Then
              MSFlexGrid1.Text = MSFlexGrid1.Text & ","
              Dec = True
              Intr = MSFlexGrid1.Text
```

```
            End If
          End If
        End If
        If ((MSFlexGrid1.Col = 8) Or (MSFlexGrid1.Col = 9) Or (MSFlexGrid1.Col = 10)) And (((KeyAscii >= 48) And
(KeyAscii <= 57)) Or ((KeyAscii = 44) Or (KeyAscii = 46))) And (Len(MSFlexGrid1.Text) < 7) Then
          If (Dec) And (KeyAscii <> 44) And (Len(MSFlexGrid1.Text) - Len(Intr) < 3) Then
            MSFlexGrid1.Text = MSFlexGrid1.Text & Chr(KeyAscii)
          End If
          If (Not Dec And KeyAscii = 44) Then
            Dec = True
            MSFlexGrid1.Text = MSFlexGrid1.Text & Chr(KeyAscii)
            Intr = MSFlexGrid1.Text
          End If
          If Not Dec Then
            MSFlexGrid1.Text = MSFlexGrid1.Text & Chr(KeyAscii)
            If ((Len(MSFlexGrid1.Text) = 3) And Mid(MSFlexGrid1.Text, 1, 1) <> "-") Then
              MSFlexGrid1.Text = MSFlexGrid1.Text & ","
              Dec = True
              Intr = MSFlexGrid1.Text
            End If
          End If
        End If

        If (MSFlexGrid1.Col = 11) Or (MSFlexGrid1.Col = 12) Then
          If (MSFlexGrid1.Text = "") And ((KeyAscii = 102) Or (KeyAscii = 115) Or (KeyAscii = 116) Or (KeyAscii =
109)) Then
            MSFlexGrid1.Text = MSFlexGrid1.Text & Chr(KeyAscii)
          Else
            If Mid(MSFlexGrid1.Text, 1, 1) = "f" And (KeyAscii >= 48) And (KeyAscii <= 57) And
(Len(MSFlexGrid1.Text) < 4) Then
              MSFlexGrid1.Text = MSFlexGrid1.Text & Chr(KeyAscii)
            End If
            If Mid(MSFlexGrid1.Text, 1, 1) = "s" And (KeyAscii >= 48) And (KeyAscii <= 57) And
(Len(MSFlexGrid1.Text) < 3) Then
              MSFlexGrid1.Text = MSFlexGrid1.Text & Chr(KeyAscii)
            End If
            If Mid(MSFlexGrid1.Text, 1, 1) = "t" And (KeyAscii >= 48) And (KeyAscii <= 57) And
(Len(MSFlexGrid1.Text) < 3) Then
              MSFlexGrid1.Text = MSFlexGrid1.Text & Chr(KeyAscii)
            End If
            If Mid(MSFlexGrid1.Text, 1, 1) = "m" And (KeyAscii >= 48) And (KeyAscii <= 57) And
(Len(MSFlexGrid1.Text) < 3) Then
              MSFlexGrid1.Text = MSFlexGrid1.Text & Chr(KeyAscii)
            End If
          End If
        End If
      End If
      'MSFlexGrid1.Text = KeyAscii
      End Sub

      Private Sub MSFlexGrid1_LeaveCell()
        If (MSFlexGrid1.Col = 3) Or (MSFlexGrid1.Col = 4) Or (MSFlexGrid1.Col = 5) Or (MSFlexGrid1.Col = 6) Or
(MSFlexGrid1.Col = 7) Or (MSFlexGrid1.Col = 8) Or (MSFlexGrid1.Col = 9) Or (MSFlexGrid1.Col = 10) Then
          If ((Len(MSFlexGrid1.Text) = 4) And (Mid(MSFlexGrid1.Text, 4, 4) = ",") Or ((Len(MSFlexGrid1.Text) = 5)
And (Mid(MSFlexGrid1.Text, 4, 4) = ","))) Then
            MSFlexGrid1.Text = Replace(MSFlexGrid1.Text, ",", "")
          End If
        End If
      End Sub

      Private Sub MSFlexGrid1_RowColChange()
        'Detecta mudança de seleção de célula
        Mudou_Célula = True
        Atualiza_Rótulos_Colunas
      End Sub

      Public Sub Atualiza_Rótulos_Colunas()
        For I = 0 To 4
          Label1(I).BackColor = &H8000000F
        Next
        If MSFlexGrid1.Col = 1 Then
          Label1(0).BackColor = &H8000000C
        End If
```

```
        If MSFlexGrid1.Col = 2 Then
          Label1(1).BackColor = &H8000000C
        End If
        If MSFlexGrid1.Col = 3 Or MSFlexGrid1.Col = 4 Or MSFlexGrid1.Col = 5 Or MSFlexGrid1.Col = 6 Or
MSFlexGrid1.Col = 7 Then
          Label1(2).BackColor = &H8000000C
        End If
        If MSFlexGrid1.Col = 8 Or MSFlexGrid1.Col = 9 Or MSFlexGrid1.Col = 10 Then
          Label1(3).BackColor = &H8000000C
        End If
        If MSFlexGrid1.Col = 11 Or MSFlexGrid1.Col = 12 Then
          Label1(4).BackColor = &H8000000C
        End If
      End Sub

      Private Sub Toolbar1_ButtonClick(ByVal Button As ComctlLib.Button)
        If Button.Key = "Abrir" Then
          Form3.Show
        End If
        If Button.Key = "Salvar" Then
          If NovoArquivo Then
            Form4.Show
          End If
            Salvar (NomeArquivo)
        End If
      End Sub
      Private Sub Toolbar2_ButtonClick(ByVal Button As ComctlLib.Button)
        If Button.Key = "Recortar" Then
          Set aux = Desfazer.Add(MSFlexGrid1.Row, MSFlexGrid1.Col, MSFlexGrid1.RowSel, MSFlexGrid1.ColSel,
AçãoRecortar, MSFlexGrid1.Clip)
          Clipboard.Clear
          Clipboard.SetText MSFlexGrid1.Clip
          MSFlexGrid1.Text = ""
        End If
        If Button.Key = "Copiar" Then
          Set aux = Desfazer.Add(MSFlexGrid1.Row, MSFlexGrid1.Col, MSFlexGrid1.RowSel, MSFlexGrid1.ColSel,
AçãoCopiar, Clipboard.GetText)
          Clipboard.Clear
          Clipboard.SetText MSFlexGrid1.Clip
        End If
        If Button.Key = "Colar" Then
          Texto = Clipboard.GetText
          For I = 1 To Len(Texto)
            If Asc(Mid(Texto, I, 1)) = 13 Then
              linhas = linhas + 1
            End If
            If (Asc(Mid(Texto, I, 1)) = 9) And (linhas = 0) Then
              colunas = colunas + 1
            End If
          Next
          MSFlexGrid1.ColSel = MSFlexGrid1.Col + colunas
          MSFlexGrid1.RowSel = MSFlexGrid1.Row + linhas
          Set aux = Desfazer.Add(MSFlexGrid1.Row, MSFlexGrid1.Col, MSFlexGrid1.RowSel, MSFlexGrid1.ColSel,
AçãoColar, MSFlexGrid1.Clip)
          MSFlexGrid1.Clip = Clipboard.GetText
        End If
      End Sub
      Private Sub Toolbar3_ButtonClick(ByVal Button As ComctlLib.Button)
        If Button.Key = "Desfazer" Then
          If Desfazer.Count > 0 Then
            Row = MSFlexGrid1.Row
            Col = MSFlexGrid1.Col
            RowSel = MSFlexGrid1.RowSel
            ColSel = MSFlexGrid1.ColSel
            Set aux = Fazer.Add(0, 0, 0, 0, AçãoNula, "")
            With Fazer(Fazer.Count)
              .Row = Desfazer(Desfazer.Count).Row
              .Col = Desfazer(Desfazer.Count).Col
              .RowSel = Desfazer(Desfazer.Count).RowSel
              .ColSel = Desfazer(Desfazer.Count).ColSel
              If (Desfazer(Desfazer.Count).Tipo = AçãoEditarCélula) Or (Desfazer(Desfazer.Count).Tipo =
AçãoDeletarCélula) Or (Desfazer(Desfazer.Count).Tipo = AçãoNula) Then
                .Tipo = Desfazer(Desfazer.Count).Tipo
```

```
                End If
                If (Desfazer(Desfazer.Count).Tipo = AçãoExcluirLinha) Then
                    .Tipo = AçãoInserirLinhaAcima
                End If
                If (Desfazer(Desfazer.Count).Tipo = AçãoInserirLinhaAcima) Or (Desfazer(Desfazer.Count).Tipo =
AçãoInserirLinhaAbaixo) Then
                    .Tipo = AçãoExcluirLinha
                End If
                If (Desfazer(Desfazer.Count).Tipo = AçãoCopiar) Then
                    .Tipo = AçãoRecopiar
                End If
                If (Desfazer(Desfazer.Count).Tipo = AçãoColar) Then
                    .Tipo = AçãoRecolar
                End If
                If (Desfazer(Desfazer.Count).Tipo = AçãoRecortar) Then
                    .Tipo = AçãoRerrecortar
                End If
                With MSFlexGrid1
                    .FocusRect = flexFocusNone
                    .BackColorSel = &H80000005
                    .Row = Desfazer(Desfazer.Count).Row
                    .Col = Desfazer(Desfazer.Count).Col
                    .RowSel = Desfazer(Desfazer.Count).RowSel
                    .ColSel = Desfazer(Desfazer.Count).ColSel
                End With
                .Valor = MSFlexGrid1.Clip
                With MSFlexGrid1
                    .Row = Row
                    .Col = Col
                    .RowSel = RowSel
                    .ColSel = ColSel
                    .BackColorSel = &H80000008
                    .FocusRect = flexFocusHeavy
                End With
            End With
            Set aux = Desfazer(Desfazer.Count)
            Desfazer(Desfazer.Count).Fazer
            Desfazer.Remove (Desfazer.Count)
        End If
    End If
    If Button.Key = "Refazer" Then
        If Fazer.Count > 0 Then
            Row = MSFlexGrid1.Row
            Col = MSFlexGrid1.Col
            RowSel = MSFlexGrid1.RowSel
            ColSel = MSFlexGrid1.ColSel
            Set aux = Desfazer.Add(0, 0, 0, 0, AçãoNula, "")
            With Desfazer(Desfazer.Count)
                .Row = Fazer(Fazer.Count).Row
                .Col = Fazer(Fazer.Count).Col
                .RowSel = Fazer(Fazer.Count).RowSel
                .ColSel = Fazer(Fazer.Count).ColSel
                .Tipo = Fazer(Fazer.Count).Tipo
                With MSFlexGrid1
                    .FocusRect = flexFocusNone
                    .BackColorSel = &H80000005
                    .Row = Fazer(Fazer.Count).Row
                    .Col = Fazer(Fazer.Count).Col
                    .RowSel = Fazer(Fazer.Count).RowSel
                    .ColSel = Fazer(Fazer.Count).ColSel
                End With
                .Valor = MSFlexGrid1.Clip
                With MSFlexGrid1
                    .Row = Row
                    .Col = Col
                    .RowSel = RowSel
                    .ColSel = ColSel
                    .BackColorSel = &H80000008
                    .FocusRect = flexFocusHeavy
                End With
            End With
            Fazer(Fazer.Count).Fazer
            Fazer.Remove (Fazer.Count)
```

```
            End If
          End If
          Atualiza_Rótulos_Colunas
        End Sub
        Private Sub Toolbar4_ButtonClick(ByVal Button As ComctlLib.Button)
          If Button.Key = "Excluir" Then
            Linha = MSFlexGrid1.Row
            Coluna = MSFlexGrid1.Col
            LinhaSel = MSFlexGrid1.RowSel
            ColunaSel = MSFlexGrid1.ColSel
            With MSFlexGrid1
              .BackColorSel = &H80000005
              .FocusRect = flexFocusNone
              .RowSel = MSFlexGrid1.Row
              .Col = 1
              .ColSel = MSFlexGrid1.Cols - 1
            End With
            Valor = MSFlexGrid1.Clip
            With MSFlexGrid1
              .BackColorSel = &H80000008
              .FocusRect = flexFocusHeavy
              .Row = Linha
              .Col = Coluna
              .RowSel = LinhaSel
              .ColSel = ColunaSel
            End With
            Set aux = Desfazer.Add(MSFlexGrid1.Row, MSFlexGrid1.Col, MSFlexGrid1.RowSel, MSFlexGrid1.ColSel,
AçãoInserirLinhaAcima, Valor)
            Linha = MSFlexGrid1.Row
            Coluna = MSFlexGrid1.Col
            MSFlexGrid1.BackColorSel = &H80000005
            MSFlexGrid1.FocusRect = flexFocusNone
            If MSFlexGrid1.Row <> MSFlexGrid1.Rows - 1 Then
              MSFlexGrid1.Row = MSFlexGrid1.Row + 1
              MSFlexGrid1.Col = 1
              MSFlexGrid1.RowSel = MSFlexGrid1.Rows - 1
              MSFlexGrid1.ColSel = MSFlexGrid1.Cols - 1
              Texto = MSFlexGrid1.Clip
              MSFlexGrid1.Row = MSFlexGrid1.Row - 1
              MSFlexGrid1.RowSel = MSFlexGrid1.Rows - 2
              MSFlexGrid1.ColSel = MSFlexGrid1.Cols - 1
              MSFlexGrid1.Clip = Texto
              MSFlexGrid1.Rows = MSFlexGrid1.Rows - 1
              MSFlexGrid1.Col = Coluna
              MSFlexGrid1.ColSel = Coluna
              MSFlexGrid1.BackColorSel = &H80000008
            Else
              MSFlexGrid1.Rows = MSFlexGrid1.Rows - 1
            End If
            MSFlexGrid1.FocusRect = flexFocusHeavy
            Atualiza_Rótulos_Colunas
          End If
          If Button.Key = "Acima" Then
            Set aux = Desfazer.Add(MSFlexGrid1.Row, MSFlexGrid1.Col, MSFlexGrid1.RowSel, MSFlexGrid1.ColSel,
AçãoExcluirLinha, "")
            Linha = MSFlexGrid1.Row
            Coluna = MSFlexGrid1.Col
            With MSFlexGrid1
              .BackColorSel = &H80000005
              .FocusRect = flexFocusNone
              .Rows = MSFlexGrid1.Rows + 1
              .Col = 1
              .RowSel = MSFlexGrid1.Rows - 2
              .ColSel = MSFlexGrid1.Cols - 1
            End With
            Texto = MSFlexGrid1.Clip
            With MSFlexGrid1
              .Row = MSFlexGrid1.Row + 1
              .RowSel = MSFlexGrid1.Rows - 1
              .ColSel = MSFlexGrid1.Cols - 1
              .Clip = Texto
              .Row = Linha
              .RowSel = Linha
```

```
            .Col = 1
            .ColSel = MSFlexGrid1.Cols - 1
            .Text = ""
            .Col = Coluna
            .ColSel = Coluna
            .BackColorSel = &H80000008
            .FocusRect = flexFocusHeavy
          End With
          Atualiza_Rótulos_Colunas
        End If
        If Button.Key = "Abaixo" Then
          Set aux = Desfazer.Add(MSFlexGrid1.Row + 1, MSFlexGrid1.Col, MSFlexGrid1.RowSel,
MSFlexGrid1.ColSel, AçãoExcluirLinha, "")
          Linha = MSFlexGrid1.Row
          Coluna = MSFlexGrid1.Col
          MSFlexGrid1.FocusRect = flexFocusNone
          MSFlexGrid1.BackColorSel = &H80000005
          MSFlexGrid1.Rows = MSFlexGrid1.Rows + 1
          MSFlexGrid1.Row = MSFlexGrid1.Row + 1
          MSFlexGrid1.Col = 1
          MSFlexGrid1.RowSel = MSFlexGrid1.Rows - 2
          MSFlexGrid1.ColSel = MSFlexGrid1.Cols - 1
          Texto = MSFlexGrid1.Clip
          If MSFlexGrid1.Row < MSFlexGrid1.Rows - 1 Then
            MSFlexGrid1.Row = MSFlexGrid1.Row + 1
          End If
          With MSFlexGrid1
            .RowSel = MSFlexGrid1.Rows - 1
            .ColSel = MSFlexGrid1.Cols - 1
            .Clip = Texto
            .Row = Linha + 1
            .RowSel = Linha + 1
            .Col = 1
            .ColSel = MSFlexGrid1.Cols - 1
            .Text = ""
            .Row = Linha
            .RowSel = Linha
            .Col = Coluna
            .ColSel = Coluna
            .BackColorSel = &H80000008
            .FocusRect = flexFocusHeavy
          End With
        End If
      End Sub

      Private Sub Toolbar5_ButtonClick(ByVal Button As ComctlLib.Button)

        If Button.Key = "Desenhar" Then
        Reset
        NovoDesenho = True
        Form1.Show
        ScaleMode = vbMillimeters
        PontoFinal_x = 0
        PontoFinal_z = 0
        n = 1
        While Form2.MSFlexGrid1.TextMatrix(n, 1) <> Empty
          Atualiza_Coordenadas (n)

          '#################################################################
          If NovoDesenho And (Função = 2 Or Função = 3 Or Função = 1) Then
            Linha_ T(PontoInicial_x, CoordenadaX), T(0, CoordenadaZ), T(PontoInicial_x, CoordenadaX),
T(PontoInicial_z, CoordenadaZ)
            Linha_ T(PontoInicial_x, CoordenadaX), T(0, CoordenadaZ), T(PontoInicial_x, CoordenadaX), T(-1 *
PontoInicial_z, CoordenadaZ)
            NovoDesenho = False
          End If
          '#################################################################

          'Se for G00
          If Função = 0 Then

          End If
```

```
            'Se for G01
            If Função = 1 Then
               Interpolação_Linear
            End If

            'Se for G02 ou G03
            If Função = 2 Or Função = 3 Then
               Interpolação_Circular
            End If

            'Se for G04
            If Função = 4 Then
               Permanência = PontoFinal_x
            End If

            'Se for G33
            If Função = 33 Then
               Modo_Roscar
            End If

            'Se for G90
            If Função = 90 Then
               SistCoord = Absoluto
            End If

            'Se for G91
            If Função = 91 Then
               SistCoord = Relativo
            End If

            'Se for G92
            If Função = 92 Then
               OrigemSist_x = PontoFinal_x
               OrigemSist_z = PontoFinal_z
            End If

            'Se for G94
            If Função = 94 Then
               VelAvanço = mm_min
            End If

            'Se for G95
            If Função = 95 Then
               VelAvanço = mm_rot
            End If

            'Se for G96
            If Função = 96 Then
               VelCorte = m_min
            End If

            'Se for G97
            If Função = 97 Then
               VelCorte = rpm
            End If

            'Se for G98
            If Função = 98 Then
               MaxVelCorte = True
            End If

            n = n + 1
         Wend

         '#################################################################################
         Linha_ T(PontoFinal_x, CoordenadaX), T(PontoFinal_z, CoordenadaZ), T(PontoFinal_x, CoordenadaX), T(0,
CoordenadaZ)
         Linha_ T(PontoFinal_x, CoordenadaX), T(-1 * PontoFinal_z, CoordenadaZ), T(PontoFinal_x, CoordenadaX),
T(0, CoordenadaZ)
         '#################################################################################

         End If
```

```
End Sub
```

Form3

```
Private Sub Command1_Click()
   Abrir
End Sub

Private Sub Abrir()
   Dim Comando As CodigoG
   NomeArquivo = File1.Path & "\" & File1.FileName
   With Form2.MSFlexGrid1
      .BackColorSel = &H80000005
      .FocusRect = flexFocusNone
      .Row = 1
      .Col = 1
      .RowSel = Form2.MSFlexGrid1.Rows - 1
      .ColSel = Form2.MSFlexGrid1.Cols - 1
      .Text = ""
      .BackColorSel = &H80000008
      .FocusRect = flexFocusHeavy
      .RowSel = 1
      .ColSel = 1
   End With
   Open NomeArquivo For Random As #1
   I = 1
   Do While Not EOF(1)
      Get #1, I, Comando
      If Comando.n <> Empty Then
         Form2.MSFlexGrid1.TextMatrix(I, 1) = Comando.n
      End If
      If Comando.G <> Empty Then
         Form2.MSFlexGrid1.TextMatrix(I, 2) = Comando.G
      End If
      If Comando.X <> Empty Then
         Form2.MSFlexGrid1.TextMatrix(I, 3) = Comando.X
      End If
      If Comando.Y <> Empty Then
         Form2.MSFlexGrid1.TextMatrix(I, 4) = Comando.Y
      End If
      If Comando.Z <> Empty Then
         Form2.MSFlexGrid1.TextMatrix(I, 5) = Comando.Z
      End If
      If Comando.R <> Empty Then
         Form2.MSFlexGrid1.TextMatrix(I, 6) = Comando.R
      End If
      If Comando.W <> Empty Then
         Form2.MSFlexGrid1.TextMatrix(I, 7) = Comando.W
      End If
      If Comando.I <> Empty Then
         Form2.MSFlexGrid1.TextMatrix(I, 8) = Comando.I
      End If
      If Comando.J <> Empty Then
         Form2.MSFlexGrid1.TextMatrix(I, 9) = Comando.J
      End If
      If Comando.K <> Empty Then
         Form2.MSFlexGrid1.TextMatrix(I, 10) = Comando.K
      End If
      If Comando.Aux1 <> Empty Then
         Form2.MSFlexGrid1.TextMatrix(I, 11) = Comando.Aux1
      End If
      If Comando.Aux2 <> Empty Then
         Form2.MSFlexGrid1.TextMatrix(I, 12) = Comando.Aux2
      End If
      I = I + 1
   Loop
   NomeArquivo = Left(NomeArquivo, Len(NomeArquivo) - 4)
   NovoArquivo = False
   Unload Form3
   Form2.Show
   Close #1
End Sub

Private Sub Command2_Click()
```

```
    Unload Form3
End Sub

Private Sub Dir1_Change()
    File1.Path = Dir1.Path
End Sub

Private Sub Dir1_KeyDown(KeyCode As Integer, Shift As Integer)
    If KeyCode = 13 Then
        If File1.FileName <> "" Then
            Abrir
            Unload Form3
        End If
    ElseIf KeyCode = 27 Then
        Unload Form3
    End If
End Sub

Private Sub Drive1_Change()
    Dim fs, d, s
    Set fs = CreateObject("Scripting.FileSystemObject")
    Set d = fs.GetDrive(fs.GetDriveName(Drive1.Drive))
    If d.Isready Then
        Dir1.Path = Drive1.Drive
    Else
        teste = MsgBox("Drive " & Drive1.Drive & "\ não disponível", vbCritical)
        Dir1.Path = "c:"
        Drive1.Drive = "c:"
    End If
End Sub

Public Sub Form3_KeyDown(KeyCode As Integer, Shift As Integer)
    If KeyCode = 13 Then
        If File1.FileName <> "" Then
            Abrir
            Unload Form3
        End If
    ElseIf KeyCode = 27 Then
        Unload Form3
    End If
End Sub

Private Sub Drive1_KeyDown(KeyCode As Integer, Shift As Integer)
    If KeyCode = 13 Then
        If File1.FileName <> "" Then
            Abrir
            Unload Form3
        End If
    ElseIf KeyCode = 27 Then
        Unload Form3
    End If
End Sub

Private Sub File1_KeyDown(KeyCode As Integer, Shift As Integer)
    If KeyCode = 13 Then
        If File1.FileName <> "" Then
            Abrir
            Unload Form3
        End If
    ElseIf KeyCode = 27 Then
        Unload Form3
    End If
End Sub
```

Form4

```
Private Sub Command1_Click()
   If Text1.Text <> "" Then
      NomeArquivo = Dir1.Path & "\" & Text1.Text
      NovoArquivo = False
      Salvar (NomeArquivo)
      Unload Form4
   End If
End Sub

Private Sub Command2_Click()
   Unload Form4
End Sub

Private Sub Dir1_Change()
   File1.Path = Dir1.Path
End Sub

Private Sub Dir1_KeyDown(KeyCode As Integer, Shift As Integer)
   If KeyCode = 13 Then
      If Text1.Text <> "" Then
         NomeArquivo = Dir1.Path & "\" & Text1.Text
         NovoArquivo = False
         Salvar (NomeArquivo)
         Unload Form4
      End If
   ElseIf KeyCode = 27 Then
      Unload Form4
   End If
End Sub

Private Sub Drive1_Change()
   Dim fs, d, s
   Set fs = CreateObject("Scripting.FileSystemObject")
   Set d = fs.GetDrive(fs.GetDriveName(Drive1.Drive))
   If d.Isready Then
      Dir1.Path = Drive1.Drive
   Else
      teste = MsgBox("Drive " & Drive1.Drive & "\ não disponível", vbCritical)
      Dir1.Path = "c:"
      Drive1.Drive = "c:"
   End If
End Sub

Private Sub Drive1_KeyDown(KeyCode As Integer, Shift As Integer)
   If KeyCode = 13 Then
      If Text1.Text <> "" Then
         NomeArquivo = Dir1.Path & "\" & Text1.Text
         NovoArquivo = False
         Salvar (NomeArquivo)
         Unload Form4
      End If
   ElseIf KeyCode = 27 Then
      Unload Form4
   End If
End Sub

Private Sub File1_Click()
   Text1.Text = Left(File1.FileName, Len(File1.FileName) - 4)
End Sub

Public Sub Form4_KeyDown(KeyCode As Integer, Shift As Integer)
   If KeyCode = 13 Then
      If Text1.Text <> "" Then
         NomeArquivo = Dir1.Path & "\" & Text1.Text
         NovoArquivo = False
         Salvar (NomeArquivo)
         Unload Form4
      End If
   ElseIf KeyCode = 27 Then
      Unload Form4
```

```
    End If
End Sub

Private Sub File1_KeyDown(KeyCode As Integer, Shift As Integer)
    If KeyCode = 13 Then
        If Text1.Text <> "" Then
            NomeArquivo = Dir1.Path & "\" & Text1.Text
            NovoArquivo = False
            Salvar (NomeArquivo)
            Unload Form4
        End If
    ElseIf KeyCode = 27 Then
        Unload Form4
    End If
End Sub

Private Sub Text1_KeyDown(KeyCode As Integer, Shift As Integer)
    If KeyCode = 13 Then
        If Text1.Text <> "" Then
            NomeArquivo = Dir1.Path & "\" & Text1.Text
            NovoArquivo = False
            Salvar (NomeArquivo)
            Unload Form4
        End If
    ElseIf KeyCode = 27 Then
        Unload Form4
    End If
End Sub
```

MDIForm1

```
Private Sub Abrir_Click()
   Form3.Show
End Sub

Private Sub Desenho_Click()
   Form1.Show
End Sub

Private Sub Editor_Click()
   Form2.Show
End Sub

Private Sub MDIForm_Terminate()
   Unload MDIForm1
   Unload Form1
   Unload Form2
   Unload Form3
   Unload Form4
End Sub

Private Sub Salvar_Click()
   If NomeArquivo <> "" Then
    ' Salvar (NomeArquivo)
   End If
End Sub

Private Sub SalvarComo_Click()
   Form4.Show
End Sub
```

Module1

```
Public Type CodigoG
   n As String
   G As String
   X As String
   Y As String
   Z As String
   R As String
   W As String
   I As String
   J As String
   K As String
   Aux1 As String
   Aux2 As String
End Type

Public NomeArquivo As String

Public NovoArquivo, NovoDesenho As Boolean

Public PontoInicial_x, PontoInicial_z, PontoFinal_x, PontoFinal_z As Single

Public Centro_x, Centro_z, Centro1_x, Centro1_z, Centro2_x, Centro2_z As Single

Public Raio As Single

Public Função As Integer

Public AuxI, AuxJ, AuxK As Single

Public AnguloInicial, AnguloFinal As Single

Public Escala As Long

Public dz As Single

Public Enum TipoCoordenada
   CoordenadaX = 0
   CoordenadaZ = 1
End Enum

Public Permanência As Single

Public Enum SistemaCoordenadas
   Absoluto = 0
   Relativo = 1
End Enum

Public SistCoord As SistemaCoordenadas

Public OrigemSist_x, OrigemSist_z As Single

Public Enum VelocidadeAvanço
   mm_min = 0
   mm_rot = 1
End Enum

Public VelAvanço As VelocidadeAvanço

Public Enum VelocidadeCorte
   m_min = 0
   rpm = 1
End Enum

Public VelCorte As VelocidadeCorte

Public MaxVelCorte As Boolean

Public Sub Salvar(Nome As String)
   Dim Comando As CodigoG
```

```
  m = 0
  While Form2.MSFlexGrid1.TextMatrix(m, 1) <> "" 'Empty
    m = m + 1
  Wend
  NomeArquivo = Nome
  NovoArquivo = False
  Open NomeArquivo & ".enc" For Random As #1
  For I = 1 To m
     Comando.n = Form2.MSFlexGrid1.TextMatrix(I, 1)
     Comando.G = Form2.MSFlexGrid1.TextMatrix(I, 2)
     Comando.X = Form2.MSFlexGrid1.TextMatrix(I, 3)
     Comando.Y = Form2.MSFlexGrid1.TextMatrix(I, 4)
     Comando.Z = Form2.MSFlexGrid1.TextMatrix(I, 5)
     Comando.R = Form2.MSFlexGrid1.TextMatrix(I, 6)
     Comando.W = Form2.MSFlexGrid1.TextMatrix(I, 7)
     Comando.I = Form2.MSFlexGrid1.TextMatrix(I, 8)
     Comando.J = Form2.MSFlexGrid1.TextMatrix(I, 9)
     Comando.K = Form2.MSFlexGrid1.TextMatrix(I, 10)
     Comando.Aux1 = Form2.MSFlexGrid1.TextMatrix(I, 11)
     Comando.Aux2 = Form2.MSFlexGrid1.TextMatrix(I, 12)
     Put #1, I, Comando
  Next
  Close #1
End Sub
```

Module2

```
Public Sub Linha_(ByVal X1 As Single, ByVal Y1 As Single, ByVal X2 As Single, ByVal Y2 As Single)
  Form1.Line (X1, Y1)-(X2, Y2), QBColor(15)
End Sub

Public Sub Arco_(ByVal Xc As Single, ByVal Yc As Single, ByVal Raio As Single, ByVal Inicio As Single, Fim
As Single)
  Form1.Circle (Xc, Yc), Raio, QBColor(15), Inicio, Fim
End Sub

Public Sub Reset()
  PontoInicial_x = Empty
  PontoInicial_z = Empty
  PontoFinal_x = Empty
  PontoFinal_z = Empty
  Centro_x = Empty
  Centro_z = Empty
  Centro1_x = Empty
  Centro1_z = Empty
  Centro2_x = Empty
  Centro2_z = Empty
  Raio = Empty
  Função = Empty
  AuxI = Empty
  AuxJ = Empty
  AuxK = Empty
  AnguloInicial = Empty
  AnguloFinal = Empty
  Escala = Empty
End Sub

Public Sub Atualiza_Coordenadas(n As Long)

  ScaleMode = vbMillimeters
  dz = Abs(PontoInicial_z - PontoFinal_z)

  PontoInicial_x = PontoFinal_x
  PontoInicial_z = PontoFinal_z

  If Form2.MSFlexGrid1.TextMatrix(n, 2) <> Empty Then
    Função = CInt(Form2.MSFlexGrid1.TextMatrix(n, 2))
  End If

  If Form2.MSFlexGrid1.TextMatrix(n, 3) <> Empty Then
    If SistCoord = Absoluto Then
      PontoFinal_x = OrigemSist_x + CSng(Form2.MSFlexGrid1.TextMatrix(n, 3))
    Else
      PontoFinal_x = OrigemSist_x + PontoFinal_x + CSng(Form2.MSFlexGrid1.TextMatrix(n, 3))
    End If
  End If

  If Form2.MSFlexGrid1.TextMatrix(n, 5) <> Empty Then
    If SistCoord = Absoluto Then
      PontoFinal_z = OrigemSist_z + CSng(Form2.MSFlexGrid1.TextMatrix(n, 5)) / 2
    Else
      PontoFinal_z = OrigemSist_z + PontoFinal_z + CSng(Form2.MSFlexGrid1.TextMatrix(n, 5)) / 2
    End If
  End If

  If Form2.MSFlexGrid1.TextMatrix(n, 6) <> Empty Then
    Raio = CSng(Form2.MSFlexGrid1.TextMatrix(n, 6))
  End If

  If Form2.MSFlexGrid1.TextMatrix(n, 8) <> Empty Then
    AuxI = CSng(Form2.MSFlexGrid1.TextMatrix(n, 8))
  End If

  If Form2.MSFlexGrid1.TextMatrix(n, 9) <> Empty Then
    AuxJ = CSng(Form2.MSFlexGrid1.TextMatrix(n, 9))
  End If
```

```
        If Form2.MSFlexGrid1.TextMatrix(n, 10) <> Empty Then
            AuxK = CSng(Form2.MSFlexGrid1.TextMatrix(n, 10))
        End If

    End Sub

    Public Function T(ByVal Coord As Single, Tipo As TipoCoordenada) As Single
        Escala = 1
        If Tipo = CoordenadaX Then
            T = 5 + Coord / Escala
        ElseIf Tipo = CoordenadaZ Then
            T = ((Form1.Height - Form1.Top) / (2 * 56.7)) - (Coord / Escala)
        End If
    End Function

    Public Sub Interpolação_Linear()
        Linha_ T(PontoInicial_x, CoordenadaX), T(PontoInicial_z, CoordenadaZ), T(PontoFinal_x, CoordenadaX),
T(PontoFinal_z, CoordenadaZ)
        Linha_ T(PontoInicial_x, CoordenadaX), T(-1 * PontoInicial_z, CoordenadaZ), T(PontoFinal_x, CoordenadaX),
T(-1 * PontoFinal_z, CoordenadaZ)
    End Sub

    Public Sub Interpolação_Circular()

        If Raio <> Empty Then
            Coordenadas_Centro
        End If

        If Função = 2 Then
            If (PontoFinal_x < PontoInicial_x) And (PontoFinal_z > PontoInicial_z) Then

                CALCULA_COORD_CENTRO_SENTHORARIO_1QUADRANTE
                CALCULA_ANGULOS_SENTHORARIO_1QUADRANTE

            ElseIf (PontoFinal_x < PontoInicial_x) And (PontoFinal_z < PontoInicial_z) Then

                CALCULA_COORD_CENTRO_SENTHORARIO_2QUADRANTE
                CALCULA_ANGULOS_SENTHORARIO_2QUADRANTE

            ElseIf (PontoFinal_x > PontoInicial_x) And (PontoFinal_z < PontoInicial_z) Then

                CALCULA_COORD_CENTRO_SENTHORARIO_3QUADRANTE
                CALCULA_ANGULOS_SENTHORARIO_3QUADRANTE

            ElseIf (PontoFinal_x > PontoInicial_x) And (PontoFinal_z > PontoInicial_z) Then

                CALCULA_COORD_CENTRO_SENTHORARIO_4QUADRANTE
                CALCULA_ANGULOS_SENTHORARIO_4QUADRANTE

            End If

        Else
            If (PontoFinal_x < PontoInicial_x) And (PontoFinal_z > PontoInicial_z) Then

                CALCULA_COORD_CENTRO_SENTANTIHORARIO_1QUADRANTE
                CALCULA_ANGULOS_SENTANTIHORARIO_1QUADRANTE

            ElseIf (PontoFinal_x < PontoInicial_x) And (PontoFinal_z < PontoInicial_z) Then

                CALCULA_COORD_CENTRO_SENTANTIHORARIO_2QUADRANTE
                CALCULA_ANGULOS_SENTANTIHORARIO_2QUADRANTE

            ElseIf (PontoFinal_x > PontoInicial_x) And (PontoFinal_z < PontoInicial_z) Then

                CALCULA_COORD_CENTRO_SENTANTIHORARIO_3QUADRANTE
                CALCULA_ANGULOS_SENTANTIHORARIO_3QUADRANTE

            ElseIf (PontoFinal_x > PontoInicial_x) And (PontoFinal_z > PontoInicial_z) Then

                CALCULA_COORD_CENTRO_SENTANTIHORARIO_4QUADRANTE
                CALCULA_ANGULOS_SENTANTIHORARIO_4QUADRANTE

            End If
```

```
        End If

        Raio = ((PontoInicial_x - Centro_x) ^ 2 + (PontoInicial_z - Centro_z) ^ 2) ^ 0.5
        Arco_ T(Centro_x, CoordenadaX), T(Centro_z, CoordenadaZ), Raio, AnguloInicial, AnguloFinal
        Arco_ T(Centro_x, CoordenadaX), T(-1 * Centro_z, CoordenadaZ), Raio, Abs(2 * 3.14159265359 -
AnguloFinal), Abs(2 * 3.14159265359 - AnguloInicial)

    End Sub

    Public Sub Coordenadas_Centro()
        a = 2 * PontoFinal_x - 2 * PontoInicial_x
        b = 2 * PontoFinal_z - 2 * PontoInicial_z
        c = PontoInicial_x ^ 2 - PontoFinal_x ^ 2 + PontoInicial_z ^ 2 - PontoFinal_z ^ 2
        d = 1 + (b / a) ^ 2
        e = 2 * b * c / a ^ 2 + 2 * b * PontoInicial_x / a - 2 * PontoInicial_z
        f = PontoInicial_x ^ 2 + PontoInicial_z ^ 2 + (c / a) ^ 2 + 2 * c * PontoInicial_x / a - Raio ^ 2
        Centro1_z = (-e + (e ^ 2 - 4 * d * f) ^ 0.5) / (2 * d)
        Centro1_x = -(b * Centro1_z + c) / a
        Centro2_z = (-e - (e ^ 2 - 4 * d * f) ^ 0.5) / (2 * d)
        Centro2_x = -(b * Centro2_z + c) / a
    End Sub

    Private Sub CALCULA_COORD_CENTRO_SENTHORARIO_1QUADRANTE()
        If Raio <> Empty Then
            If Centro1_x >= PontoInicial_x And Centro1_z >= PontoInicial_z Then
                Centro_x = Centro1_x
                Centro_z = Centro1_z
            End If
            If Centro2_x >= PontoInicial_x And Centro2_z >= PontoInicial_z Then
                Centro_x = Centro2_x
                Centro_z = Centro2_z
            End If
        Else
            Centro_x = PontoInicial_x
            Centro_z = PontoInicial_z
            If AuxI <> Empty Then
                Centro_x = PontoInicial_x + AuxI
            End If
            If AuxK <> Empty Then
                Centro_z = PontoInicial_z + AuxK
            End If
        End If
    End Sub

    Private Sub CALCULA_COORD_CENTRO_SENTHORARIO_2QUADRANTE()
        If Raio <> Empty Then
            If Centro1_x <= PontoInicial_x And Centro1_z >= PontoInicial_z Then
                Centro_x = Centro1_x
                Centro_z = Centro1_z
            End If
            If Centro2_x <= PontoInicial_x And Centro2_z >= PontoInicial_z Then
                Centro_x = Centro2_x
                Centro_z = Centro2_z
            End If
        Else
            Centro_x = PontoInicial_x
            Centro_z = PontoInicial_z
            If AuxI <> Empty Then
                Centro_x = PontoInicial_x - AuxI
            End If
            If AuxK <> Empty Then
                Centro_z = PontoInicial_z + AuxK
            End If
        End If
    End Sub

    Private Sub CALCULA_COORD_CENTRO_SENTHORARIO_3QUADRANTE()
        If Raio <> Empty Then
            If Centro1_x <= PontoInicial_x And Centro1_z <= PontoInicial_z Then
                Centro_x = Centro1_x
                Centro_z = Centro1_z
            End If
            If Centro2_x <= PontoInicial_x And Centro2_z <= PontoInicial_z Then
```

```
        Centro_x = Centro2_x
        Centro_z = Centro2_z
      End If
    Else
      Centro_x = PontoInicial_x
      Centro_z = PontoInicial_z
      If AuxI <> Empty Then
        Centro_x = PontoInicial_x - AuxI
      End If
      If AuxK <> Empty Then
        Centro_z = PontoInicial_z - AuxK
      End If
    End If
End Sub

Private Sub CALCULA_COORD_CENTRO_SENTHORARIO_4QUADRANTE()
  If Raio <> Empty Then
    If Centro1_x >= PontoInicial_x And Centro1_z <= PontoInicial_z Then
      Centro_x = Centro1_x
      Centro_z = Centro1_z
    End If
    If Centro2_x >= PontoInicial_x And Centro2_z <= PontoInicial_z Then
      Centro_x = Centro2_x
      Centro_z = Centro2_z
    End If
  Else
    Centro_x = PontoInicial_x
    Centro_z = PontoInicial_z
    If AuxI <> Empty Then
      Centro_x = PontoInicial_x + AuxI
    End If
    If AuxK <> Empty Then
      Centro_z = PontoInicial_z - AuxK
    End If
  End If
End Sub

Private Sub CALCULA_COORD_CENTRO_SENTANTIHORARIO_1QUADRANTE()
  If Raio <> Empty Then
    If Centro1_x <= PontoInicial_x And Centro1_z <= PontoInicial_z Then
      Centro_x = Centro1_x
      Centro_z = Centro1_z
    End If
    If Centro2_x <= PontoInicial_x And Centro2_z <= PontoInicial_z Then
      Centro_x = Centro2_x
      Centro_z = Centro2_z
    End If
  Else
    Centro_x = PontoInicial_x
    Centro_z = PontoInicial_z
    If AuxI <> Empty Then
      Centro_x = PontoInicial_x - AuxI
    End If
    If AuxK <> Empty Then
      Centro_z = PontoInicial_z - AuxK
    End If
  End If
End Sub

Private Sub CALCULA_COORD_CENTRO_SENTANTIHORARIO_2QUADRANTE()
  If Raio <> Empty Then
    If Centro1_x >= PontoInicial_x And Centro1_z <= PontoInicial_z Then
      Centro_x = Centro1_x
      Centro_z = Centro1_z
    End If
    If Centro2_x >= PontoInicial_x And Centro2_z <= PontoInicial_z Then
      Centro_x = Centro2_x
      Centro_z = Centro2_z
    End If
  Else
    Centro_x = PontoInicial_x
    Centro_z = PontoInicial_z
    If AuxI <> Empty Then
```

```
        Centro_x = PontoInicial_x + AuxI
      End If
      If AuxK <> Empty Then
        Centro_z = PontoInicial_z - AuxK
      End If
    End If
End Sub

Private Sub CALCULA_COORD_CENTRO_SENTANTIHORARIO_3QUADRANTE()
    If Raio <> Empty Then
      If Centro1_x >= PontoInicial_x And Centro1_z >= PontoInicial_z Then
        Centro_x = Centro1_x
        Centro_z = Centro1_z
      End If
      If Centro2_x >= PontoInicial_x And Centro2_z >= PontoInicial_z Then
        Centro_x = Centro2_x
        Centro_z = Centro2_z
      End If
    Else
      Centro_x = PontoInicial_x
      Centro_z = PontoInicial_z
      If AuxI <> Empty Then
        Centro_x = PontoInicial_x + AuxI
      End If
      If AuxK <> Empty Then
        Centro_z = PontoInicial_z + AuxK
      End If
    End If
End Sub

Private Sub CALCULA_COORD_CENTRO_SENTANTIHORARIO_4QUADRANTE()
    If Raio <> Empty Then
      If Centro1_x <= PontoInicial_x And Centro1_z >= PontoInicial_z Then
        Centro_x = Centro1_x
        Centro_z = Centro1_z
      End If
      If Centro2_x <= PontoInicial_x And Centro2_z >= PontoInicial_z Then
        Centro_x = Centro2_x
        Centro_z = Centro2_z
      End If
    Else
      Centro_x = PontoInicial_x
      Centro_z = PontoInicial_z
      If AuxI <> Empty Then
        Centro_x = PontoInicial_x - AuxI
      End If
      If AuxK <> Empty Then
        Centro_z = PontoInicial_z + AuxK
      End If
    End If
End Sub

Private Sub CALCULA_ANGULOS_SENTHORARIO_1QUADRANTE()
    If PontoFinal_x = Centro_x Then
      AnguloInicial = 3 * 3.14159265359 / 2
    Else
      AnguloInicial = Atn(Abs(PontoFinal_z - Centro_z) / Abs(PontoFinal_x - Centro_x)) + 3.14159265359
    End If
    If PontoInicial_x = Centro_x Then
      AnguloFinal = 3 * 3.14159265359 / 2
    Else
      AnguloFinal = Atn(Abs(PontoInicial_z - Centro_z) / Abs(PontoInicial_x - Centro_x)) + 3.14159265359
    End If
End Sub

Private Sub CALCULA_ANGULOS_SENTHORARIO_2QUADRANTE()
    If PontoFinal_z = Centro_z Then
      AnguloInicial = 2 * 3.14159265359
    Else
      AnguloInicial = Atn(Abs(PontoFinal_x - Centro_x) / Abs(PontoFinal_z - Centro_z)) + 3 * 3.14159265359 / 2
    End If
    If PontoInicial_z = Centro_z Then
      AnguloFinal = 2 * 3.14159265359
```

```
    Else
        AnguloFinal = Atn(Abs(PontoInicial_x - Centro_x) / Abs(PontoInicial_z - Centro_z)) + 3 * 3.14159265359 / 2
    End If
End Sub

Private Sub CALCULA_ANGULOS_SENTHORARIO_3QUADRANTE()
    If PontoFinal_x = Centro_x Then
        AnguloInicial = 3.14159265359 / 2
    Else
        AnguloInicial = Atn(Abs(PontoFinal_z - Centro_z) / Abs(PontoFinal_x - Centro_x))
    End If
    If PontoInicial_x = Centro_x Then
        AnguloFinal = 3.14159265359 / 2
    Else
        AnguloFinal = Atn(Abs(PontoInicial_z - Centro_z) / Abs(PontoInicial_x - Centro_x))
    End If
End Sub

Private Sub CALCULA_ANGULOS_SENTHORARIO_4QUADRANTE()
    If PontoFinal_z = Centro_z Then
        AnguloInicial = 3.14159265359
    Else
        AnguloInicial = Atn(Abs(PontoFinal_x - Centro_x) / Abs(PontoFinal_z - Centro_z)) + 3.14159265359 / 2
    End If
    If PontoInicial_z = Centro_z Then
        AnguloFinal = 3.14159265359
    Else
        AnguloFinal = Atn(Abs(PontoInicial_x - Centro_x) / Abs(PontoInicial_z - Centro_z)) + 3.14159265359 / 2
    End If
End Sub

Private Sub CALCULA_ANGULOS_SENTANTIHORARIO_1QUADRANTE()
    If PontoInicial_x = Centro_x Then
        AnguloInicial = 3.14159265359 / 2
    Else
        AnguloInicial = Atn(Abs(PontoInicial_z - Centro_z) / Abs(PontoInicial_x - Centro_x))
    End If
    If PontoFinal_x = Centro_x Then
        AnguloFinal = 3.14159265359 / 2
    Else
        AnguloFinal = Atn(Abs(PontoFinal_z - Centro_z) / Abs(PontoFinal_x - Centro_x))
    End If
End Sub

Private Sub CALCULA_ANGULOS_SENTANTIHORARIO_2QUADRANTE()
    If PontoInicial_z = Centro_z Then
        AnguloInicial = 3.14159265359
    Else
        AnguloInicial = Atn(Abs(PontoInicial_x - Centro_x) / Abs(PontoInicial_z - Centro_z)) + 3.14159265359 / 2
    End If
    If PontoFinal_z = Centro_z Then
        AnguloFinal = 3.14159265359
    Else
        AnguloFinal = Atn(Abs(PontoFinal_x - Centro_x) / Abs(PontoFinal_z - Centro_z)) + 3.14159265359 / 2
    End If
End Sub

Private Sub CALCULA_ANGULOS_SENTANTIHORARIO_3QUADRANTE()
    If PontoInicial_x = Centro_x Then
        AnguloInicial = 3 * 3.14159265359 / 2
    Else
        AnguloInicial = Atn(Abs(PontoInicial_z - Centro_z) / Abs(PontoInicial_x - Centro_x)) + 3.14159265359
    End If
    If PontoFinal_x = Centro_x Then
        AnguloFinal = 3 * 3.14159265359 / 2
    Else
        AnguloFinal = Atn(Abs(PontoFinal_z - Centro_z) / Abs(PontoFinal_x - Centro_x)) + 3.14159265359
    End If
End Sub

Private Sub CALCULA_ANGULOS_SENTANTIHORARIO_4QUADRANTE()
    If PontoInicial_z = Centro_z Then
        AnguloInicial = 2 * 3.14159265359
```

```
        Else
          AnguloInicial = Atn(Abs(PontoInicial_x - Centro_x) / Abs(PontoInicial_z - Centro_z)) + 3 * 3.14159265359 /
2
        End If
        If PontoFinal_z = Centro_z Then
          AnguloFinal = 2 * 3.14159265359
        Else
          AnguloFinal = Atn(Abs(PontoFinal_x - Centro_x) / Abs(PontoFinal_z - Centro_z)) + 3 * 3.14159265359 / 2
        End If
      End Sub

      Public Sub Modo_Roscar()
        Passo = AuxI
        nPassos = Abs(PontoFinal_x - PontoInicial_x) \ AuxI
        Inicio = PontoInicial_x
        z1 = PontoInicial_z
        z2 = z1 + dz
        For I = 1 To nPassos
          Linha_ T(Inicio, CoordenadaX), T(z1, CoordenadaZ), T(Inicio + Passo / 2, CoordenadaX), T(z2, CoordenadaZ)
          Linha_ T(Inicio, CoordenadaX), T(-z1, CoordenadaZ), T(Inicio + Passo / 2, CoordenadaX), T(-z2,
CoordenadaZ)
          Inicio = Inicio + Passo / 2
          Linha_ T(Inicio, CoordenadaX), T(z2, CoordenadaZ), T(Inicio + Passo / 2, CoordenadaX), T(z1, CoordenadaZ)
          Linha_ T(Inicio, CoordenadaX), T(-z2, CoordenadaZ), T(Inicio + Passo / 2, CoordenadaX), T(-z1,
CoordenadaZ)
          Inicio = Inicio + Passo / 2
        Next
        Linha_ T(Inicio, CoordenadaX), T(z1, CoordenadaZ), T(PontoFinal_x, CoordenadaX), T(PontoFinal_z,
CoordenadaZ)
        Linha_ T(Inicio, CoordenadaX), T(-z1, CoordenadaZ), T(PontoFinal_x, CoordenadaX), T(-PontoFinal_z,
CoordenadaZ)
      End Sub
```

Ação

```
Public Enum CódigoAções
   AçãoNula = 0
   AçãoEditarCélula = 1
   AçãoDeletarCélula = 2
   AçãoExcluirLinha = 3
   AçãoInserirLinhaAcima = 4
   AçãoInserirLinhaAbaixo = 5
   AçãoCopiar = 6
   AçãoColar = 7
   AçãoRecortar = 8
   AçãoRecopiar = 9
   AçãoRecolar = 10
   AçãoRerrecortar = 11
End Enum

'local variable(s) to hold property value(s)
Private mvarRow As Long 'local copy
Private mvarCol As Long 'local copy
Private mvarRowSel As Long 'local copy
Private mvarColSel As Long 'local copy
Private mvarTipo As CódigoAções 'local copy
Private mvarValor As Variant 'local copy
Public Sub Fazer()
   If (mvarTipo = AçãoEditarCélula) Or (mvarTipo = AçãoDeletarCélula) Then
      With Form2.MSFlexGrid1
         .Row = mvarRow
         .Col = mvarCol
         .RowSel = mvarRowSel
         .ColSel = mvarColSel
         .Clip = mvarValor
      End With
   End If
   If mvarTipo = AçãoExcluirLinha Then
      Linha = mvarRow
      Coluna = mvarCol
      Form2.MSFlexGrid1.BackColorSel = &H80000005
      Form2.MSFlexGrid1.FocusRect = flexFocusNone
      If mvarRow <> Form2.MSFlexGrid1.Rows - 1 Then
         Form2.MSFlexGrid1.Row = mvarRow + 1
         Form2.MSFlexGrid1.Col = 1
         Form2.MSFlexGrid1.RowSel = Form2.MSFlexGrid1.Rows - 1
         Form2.MSFlexGrid1.ColSel = Form2.MSFlexGrid1.Cols - 1
         Texto = Form2.MSFlexGrid1.Clip
         Form2.MSFlexGrid1.Row = Form2.MSFlexGrid1.Row - 1
         Form2.MSFlexGrid1.RowSel = Form2.MSFlexGrid1.Rows - 2
         Form2.MSFlexGrid1.ColSel = Form2.MSFlexGrid1.Cols - 1
         Form2.MSFlexGrid1.Clip = Texto
         Form2.MSFlexGrid1.Rows = Form2.MSFlexGrid1.Rows - 1
         Form2.MSFlexGrid1.Col = Coluna
         Form2.MSFlexGrid1.ColSel = Coluna
         Form2.MSFlexGrid1.BackColorSel = &H80000008
      Else
         Form2.MSFlexGrid1.Rows = Form2.MSFlexGrid1.Rows - 1
      End If
      Form2.MSFlexGrid1.FocusRect = flexFocusHeavy
      Form2.Atualiza_Rótulos_Colunas
   End If
   If mvarTipo = AçãoInserirLinhaAcima Then
         Linha = Form2.MSFlexGrid1.Row
      Coluna = Form2.MSFlexGrid1.Col
      With Form2.MSFlexGrid1
         .BackColorSel = &H80000005
         .FocusRect = flexFocusNone
         .Rows = Form2.MSFlexGrid1.Rows + 1
         .Col = 1
         .RowSel = Form2.MSFlexGrid1.Rows - 2
         .ColSel = Form2.MSFlexGrid1.Cols - 1
      End With
      Texto = Form2.MSFlexGrid1.Clip
      With Form2.MSFlexGrid1
```

```
        .Row = Form2.MSFlexGrid1.Row + 1
        .RowSel = Form2.MSFlexGrid1.Rows - 1
        .ColSel = Form2.MSFlexGrid1.Cols - 1
        .Clip = Texto
        .Row = Linha
        .RowSel = Linha
        .Col = 1
        .ColSel = Form2.MSFlexGrid1.Cols - 1
        .Text = ""
        .Col = Coluna
        .ColSel = Coluna
        .BackColorSel = &H80000008
        .FocusRect = flexFocusHeavy
    End With
    Form2.Atualiza_Rótulos_Colunas

    With Form2.MSFlexGrid1
        .Row = mvarRow
        .Col = 1
        .RowSel = mvarRow
        .ColSel = Form2.MSFlexGrid1.Cols - 1
        .Clip = mvarValor
        .Col = mvarCol
        .RowSel = mvarRowSel
        .ColSel = mvarColSel
    End With
End If

If mvarTipo = AçãoCopiar Then
    Clipboard.SetText (mvarValor)
    With Form2.MSFlexGrid1
        .Row = mvarRow
        .Col = mvarCol
        .RowSel = mvarRowSel
        .ColSel = mvarColSel
    End With
End If

If mvarTipo = AçãoRecopiar Then
    Clipboard.Clear
    With Form2.MSFlexGrid1
        .Row = mvarRow
        .Col = mvarCol
        .RowSel = mvarRowSel
        .ColSel = mvarColSel
    End With
    Clipboard.SetText Form2.MSFlexGrid1.Clip
End If

If mvarTipo = AçãoColar Then
    With Form2.MSFlexGrid1
        .Row = mvarRow
        .Col = mvarCol
        .RowSel = mvarRowSel
        .ColSel = mvarColSel
    End With
    Form2.MSFlexGrid1.Clip = mvarValor
End If

If mvarTipo = AçãoRecolar Then
    Texto = Clipboard.GetText
    For I = 1 To Len(Texto)
        If Asc(Mid(Texto, I, 1)) = 13 Then
            linhas = linhas + 1
        End If
        If (Asc(Mid(Texto, I, 1)) = 9) And (linhas = 0) Then
            colunas = colunas + 1
        End If
    Next
    Form2.MSFlexGrid1.Row = mvarRow
    Form2.MSFlexGrid1.Col = mvarCol
    Form2.MSFlexGrid1.ColSel = Form2.MSFlexGrid1.Col + colunas
    Form2.MSFlexGrid1.RowSel = Form2.MSFlexGrid1.Row + linhas
```

```
        Form2.MSFlexGrid1.Clip = Clipboard.GetText
    End If

    If mvarTipo = AçãoRecortar Then
        With Form2.MSFlexGrid1
            .Row = mvarRow
            .Col = mvarCol
            .RowSel = mvarRowSel
            .ColSel = mvarColSel
        End With
        Form2.MSFlexGrid1.Clip = mvarValor
    End If

    If mvarTipo = AçãoRerrecortar Then
        Clipboard.Clear
        With Form2.MSFlexGrid1
            .Row = mvarRow
            .Col = mvarCol
            .RowSel = mvarRowSel
            .ColSel = mvarColSel
        End With
        Clipboard.SetText Form2.MSFlexGrid1.Clip
        Form2.MSFlexGrid1.Text = ""
    End If
End Sub

Public Property Let Valor(ByVal vData As Variant)
'used when assigning a value to the property, on the left side of an assignment.
'Syntax: X.Valor = 5
    mvarValor = vData
End Property


Public Property Set Valor(ByVal vData As Variant)
'used when assigning an Object to the property, on the left side of a Set statement.
'Syntax: Set x.Valor = Form1
    Set mvarValor = vData
End Property


Public Property Get Valor() As Variant
'used when retrieving value of a property, on the right side of an assignment.
'Syntax: Debug.Print X.Valor
    If IsObject(mvarValor) Then
        Set Valor = mvarValor
    Else
        Valor = mvarValor
    End If
End Property

Public Property Let Tipo(ByVal vData As CódigoAções)
'used when assigning a value to the property, on the left side of an assignment.
'Syntax: X.Tipo = 5
    mvarTipo = vData
End Property

Public Property Get Tipo() As CódigoAções
'used when retrieving value of a property, on the right side of an assignment.
'Syntax: Debug.Print X.Tipo
    Tipo = mvarTipo
End Property


Public Property Let ColSel(ByVal vData As Long)
'used when assigning a value to the property, on the left side of an assignment.
'Syntax: X.ColSel = 5
    mvarColSel = vData
End Property


Public Property Get ColSel() As Long
'used when retrieving value of a property, on the right side of an assignment.
'Syntax: Debug.Print X.ColSel
```

```
    ColSel = mvarColSel
End Property


Public Property Let RowSel(ByVal vData As Long)
'used when assigning a value to the property, on the left side of an assignment.
'Syntax: X.RowSel = 5
    mvarRowSel = vData
End Property


Public Property Get RowSel() As Long
'used when retrieving value of a property, on the right side of an assignment.
'Syntax: Debug.Print X.RowSel
    RowSel = mvarRowSel
End Property


Public Property Let Col(ByVal vData As Long)
'used when assigning a value to the property, on the left side of an assignment.
'Syntax: X.Col = 5
    mvarCol = vData
End Property


Public Property Get Col() As Long
'used when retrieving value of a property, on the right side of an assignment.
'Syntax: Debug.Print X.Col
    Col = mvarCol
End Property


Public Property Let Row(ByVal vData As Long)
'used when assigning a value to the property, on the left side of an assignment.
'Syntax: X.Row = 5
    mvarRow = vData
End Property


Public Property Get Row() As Long
'used when retrieving value of a property, on the right side of an assignment.
'Syntax: Debug.Print X.Row
    Row = mvarRow
End Property
```

Ações

```
'local variable to hold collection
Private mCol As Collection
Public Function Add(Row As Long, Col As Long, RowSel As Long, ColSel As Long, Tipo As CódigoAções, Valor
As Variant, Optional sKey As String) As Ação
    'create a new object
    Dim objNewMember As Ação
    Set objNewMember = New Ação


    'set the properties passed into the method
    objNewMember.Row = Row
    objNewMember.Col = Col
    objNewMember.RowSel = RowSel
    objNewMember.ColSel = ColSel
    'If IsObject(Tipo) Then
    '   Set objNewMember.Tipo = Tipo
    'Else
      objNewMember.Tipo = Tipo
    'End If
    'If IsObject(Valor) Then
    '   Set objNewMember.Valor = Valor
    'Else
      objNewMember.Valor = Valor
    'End If
    If Len(sKey) = 0 Then
      mCol.Add objNewMember
    Else
      mCol.Add objNewMember, sKey
    End If


    'return the object created
    Set Add = objNewMember
    Set objNewMember = Nothing

    If mCol.Count > 50 Then
      mCol.Remove (1)
    End If


  End Function

  Public Property Get Item(vntIndexKey As Variant) As Ação
    'used when referencing an element in the collection
    'vntIndexKey contains either the Index or Key to the collection,
    'this is why it is declared as a Variant
    'Syntax: Set foo = x.Item(xyz) or Set foo = x.Item(5)
   Set Item = mCol(vntIndexKey)
  End Property


  Public Property Get Count() As Long
    'used when retrieving the number of elements in the
    'collection. Syntax: Debug.Print x.Count
    Count = mCol.Count
  End Property


  Public Sub Remove(vntIndexKey As Variant)
    'used when removing an element from the collection
    'vntIndexKey contains either the Index or Key, which is why
    'it is declared as a Variant
    'Syntax: x.Remove(xyz)


    mCol.Remove vntIndexKey
  End Sub
```

```
Public Property Get NewEnum() As IUnknown
    'this property allows you to enumerate
    'this collection with the For ..Each syntax
    Set NewEnum = mCol.[_NewEnum]
End Property


Private Sub Class_Initialize()
    'creates the collection when this class is created
    Set mCol = New Collection
End Sub


Private Sub Class_Terminate()
    'destroys collection when this class is terminated
    Set mCol = Nothing
End Sub
```