CIV 2802 – SISTEMAS GRÁFICOS PARA ENGENHARIA – 2010.1

Trab3: Trabalho de Produção de Desenhos e Manipulação de Dados via *Mouse*

Edição e visualização de primitivas gráficas

Objetivo

Diversos programas em computação gráfica criam objetos gráficos aplicando transformações geométricas a primitivas simples. Primitivas gráficas simples podem ser triângulos, quadrados ou círculos. Estas primitivas básicas podem ser copiadas, manipuladas geometricamente e coloridas para compor uma imagem gráfica desejada. O objetivo deste trabalho é:

(a) Implementar um programa para criar uma lista de primitivas gráficas bidimensionais (quadriláteros e círculos) na memória do computador. O programa permite a visualização e modificação das primitivas geometricamente através da manipulação das posições de seus vértices e a mudança da ordenação das primitivas dentro da lista. As cores das primitivas também podem ser especificadas. O programa também salva em um arquivo informações sobre a geometria corrente das primitivas na lista. Esses dados podem ser lidos pelo programa, recriando a lista salva no arquivo na memória do computador.

(b) Implementar uma estrutura de dados que armazena e gerencia a lista de primitivas gráficas. A estrutura de dados adotada é uma lista circular duplamente encadeada.

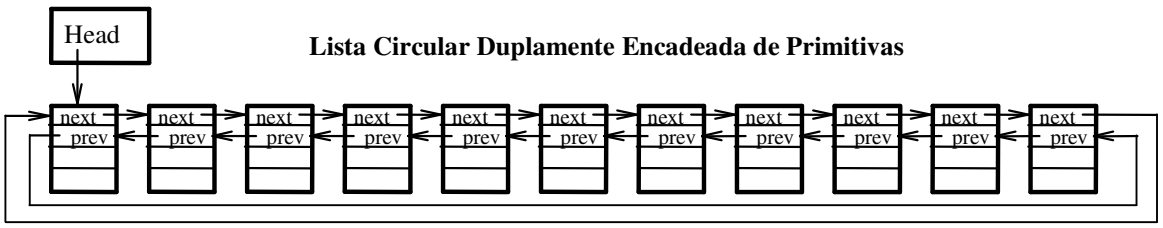
(c) Criar uma interface gráfica (menus, botões, janelas de desenho, etc.) para o programa.

(d) Implementar uma classe para manipular primitivas do tipo polígono quadrilateral, incluindo as funções que desenham esse tipo de primitiva em uma janela (*canvas*).

(e) Implementar as funções que gerenciam a entrada de dados via *mouse* atuando no *canvas* para a criação e manipulação de primitivas.

Especificação

No programa proposto, a lista de primitivas é implementada como uma lista circular duplamente encadeada, tal como ilustrado na figura abaixo. Cada registro da lista é uma estrutura contendo campos para o encadeamento da lista – ponteiros *prev* (*previous*) e *next* – e campos com as informações geométricas e outros atributos de uma primitiva. Uma lista duplamente encadeada permite uma manipulação mais eficiente da ordem das primitivas para desenho ao custo de memória adicional para um segundo ponteiro de encadeamento. A lista de primitivas deste trabalho é dita circular pois a primeira primitiva tem como primitiva anterior a última da lista, assim como a última tem como primitiva subseqüente a primeira. A lista tem um ponteiro para acesso, que é o ponteiro *Head* para a primeira primitiva da lista.



A estrutura de dados completa é descrita nos arquivos fornecidos para o trabalho. O aluno terá que complementar o programa fornecido, implementando várias funções que se encontram incompletas. Para fazer as complementações necessárias é preciso que se entenda completamente o programa fornecido. **Este entendimento faz parte do trabalho**.

Além de complementar as funções que operam sobre a estrutura de dados do programa, o aluno terá que implementar funções que, através de chamadas a funções do sistema gráfico CD, desenham o modelo na janela principal do programa e funções que controlam a visualização do modelo (dar *zoom*, mover enquadramento, etc.). Para fazer estas complementações é preciso que se entenda a parte do sistema gráfico CD que desenha primitivas descritas no sistema de coordenadas de modelagem (coordenadas reais).

Para implementar as funções que gerenciam a entrada de dados via *mouse* atuando no *canvas*, é preciso entender a sequência básica dos eventos associados a esta tarefa: (a) botão do *mouse* pressionado registra o primeiro ponto; (b) *mouse* arrastado registra as posições intermediárias do segundo ponto e dá um eco (retorno) visual e temporário da ação a ser executada; e (c) botão do *mouse* liberado registra a posição final do segundo ponto.

A parte do trabalho que compreende a criação da interface gráfica (menus, botões, etc.) do programa deve ser implementada com a complementação do arquivo *trab3.led* que especifica a configuração da interface gráfica na linguagem LED. O arquivo fornecido cria um diálogo de interface onde já são especificados uma área de desenho (*canvas*), uma área de mensagens para o usuário e a associação de teclas do teclado a funções do programa. O que se pede é a criação de menus e botões para acionar as mesmas funções que já são acionadas pelo teclado e para acionar algumas funções novas. Obrigatoriamente devem ser criados menus de barra (no topo do diálogo) e botões adjacentes.

Abaixo estão relacionadas as teclas usadas e as correspondentes funções acionadas:

- "h" ou "H" (*Help*): Informações sobre as ações do programa.
- "Ctrl+Q" (*Quit*): Saida do programa.
- "Ctrl+I" (*Info*): Informações sobre o trabalho.
- "Ctrl+N" (*New*): Começa um novo modelo.
- "Ctrl+O" (*Open*): Carrega um novo modelo de arquivo.
- "Ctrl+S" (*Save*): Salva modelo em arquivo.
- "Alt+Q" (*Quadrilateral*): Especifica tipo de primitiva: quadrilátero.
- "Alt+C" (*Circle*): Especifica tipo de primitiva: cículo.
- "Ins" (*Insert*): Insere (cria) uma nova primitiva.
- "Del" ou "Backspace": Elimina primitiva selecionada.
- "Ctrl+F" (*Front*): Coloca primitiva selecionada na frente.
- "Ctrl+B" (*Back*): Coloca primitiva selecionada atrás.
- "w" (*White*): Define branco como cor corrente.
- "k" (*blacK*): Define preto como cor corrente.
- "m" (*Magenta*): Define magenta como cor corrente.
- "b" (*Blue*): Define azul como cor corrente.
- "c" (*Cyan*): Define ciano como cor corrente.
- "g" (*Green*): Define verde como cor corrente.
- "y" (*Yellow*): Define amarelo como cor corrente.
- "r" (*Red*): Define vermelho como cor corrente.

  (Para obter cores escuras tecle Shift+tecla.)
- "Ctrl+P" (*Print*): Imprime o conteúdo da janela de desenho.
- "Ctrl+C" (*Copy*): Copia para o *clipboard* o conteúdo da janela de desenho.
- "f" (*Fit*): Ajusta a visualização total do modelo na janela**.**
- "+" (*Zoom +*): Amplia o modelo na janela de desenho.
- "–" (*Zoom –*): Reduz o modelo a janela de desenho.
- "←" (*Pan Left*): Move o enquadramento da janela de desenho para a esquerda.
- "→" (*Pan Right*): Move o enquadramento da janela de desenho para a direita.
- "↓" (*Pan Down*): Move o enquadramento da janela de desenho para baixo.
- "↑" (*Pan Up*): Move o enquadramento da janela de desenho para cima.

Fornecido e Pedido

Os seguintes arquivos são fornecidos:

- *trab3.led*: Arquivo em linguagem LED que especifica a configuração da interface gráfica. Este arquivo deve ser complementado como parte deste trabalho.
- *drv.cpp*: *Driver* que faz a interface do programa para o sistema de interface IUP e que dirige o fluxo de controle do programa. Todas as funções de *callback* registradas no IUP devem ficar neste arquivo. Parte das funções a serem complementadas se encontram neste arquivo (os locais a serem complementados estão indicados pelo comentário **`/*** COMPLETE HERE: XX ***/`**). Este arquivo também deve ser complementado adicionando-se variáveis correspondentes aos elementos de interface e às ações de interface que forem criados na interface gráfica.
- *prj.cpp*: Arquivo onde estão localizadas as funções de projeto do programa, isto é, as funções que distribuem as tarefas para os outros módulos do programa. Essas funções são definidas como membros globais (*static*) da classe *Prj*. O texto de uma variável *char** de informação, definida neste arquivo, deve ser alterado de forma a indicar o autor do trabalho.
- *prj.h*: Arquivo com a definição da classe *Prj*.
- *prm.cpp*: Arquivo onde se encontra a definição da estrutura de dados e onde estão localizadas as funções que manipulam a estrutura de dados de primitivas gráficas do programa. Essas funções estão organizadas na classe *Prm*. Parte das funções a serem complementadas se encontram neste arquivo (veja comentários **`/*** COMPLETE HERE: XX ***/`**).
- *prm.h*: Arquivo com a definição da classe *Prm*.
- *circ.cpp*: Arquivo onde estão localizadas as funções que tratam de primitivas circulares. Essas funções estão organizadas na classe *Circ*, que é derivada da classe *Prm*.
- *circ.h*: Arquivo com a definição da classe *Circ*.
- *quad.cpp*: **Arquivo que deve ser criado para implementar a classe *Quad* que trata de primitivas quadrilaterais. Esta classe deve ser derivada da classe *Prm*.**
- *quad.h*: **Arquivo que deve ser criado com a definição da classe *Quad*.**
- *prmed.cpp*: Arquivo onde estão localizadas as funções que fazer a edição (criação e manipulação da primitivas gráficas do programa. Essas funções estão organizadas na classe *PrmEd* (veja comentários **`/*** COMPLETE HERE: XX ***/`**).
- *prmed.h*: Arquivo com a definição da classe *PrmEd*.
- *prmio.cpp*: Arquivo com as funções para ler e escrever modelos em arquivo. Essas funções estão organizadas na classe *PrimIO*.
- *prmio.h*: Arquivo com a definição da classe *PrmIO*.
- *dsp.cpp*: Arquivo onde estão localizadas as funções para desenhar o modelo no *canvas*. Essas funções estão organizadas na classe *Dsp* (veja comentários **`/*** COMPLETE HERE: XX ***/`**).
- *dsp.h*: Arquivo com a definição da classe *Dsp*.
- Bibliotecas de funções do sistema de interface IUP e do sistema gráfico CD, e os correspondentes arquivos de definições e protótipos.
- Arquivos para compilação e criação do executável (Visual C, versão 9).
- Arquivo com um exemplo de primitivas que compõem um quadro de Mondrian.

Pede-se um executável do programa final e os arquivos que foram modificados para implementar o trabalho. Pede-se também, em papel, somente as linhas que foram adicionadas nos arquivos. Estas linhas devem ser indicadas da seguinte forma:

```
COMPLETE HERE: XX
LINHAS ADICIONADAS
```