void InitGraphics()——初始化图形界面

void MovePen(x,y)——移动到x与y

void DrawLine(dx,dy)——画直线，动dx与dy

void DrawArc(r,start,sweep)——半径为r，start角度开始，转sweep度

double GetWindowWidth()——窗口宽度

double GetWindowHeight()——窗口高度

double GetCurrentX()——笔的x位置

double GetCurrentY()——笔的y位置

void StartFilledRegion(grayScale)——填充颜色，grayScale为灰度（不透明度）

void void EndFilledRegion()——结束填充，与上式共用

void DrawTextString(text)——打印字符串

double TextStringWidth(text)——字符串打印长度

void SetFont(font)——设置字体

string GetFont()——得到字体

void SetPointSize(size)——设置字体大小

int GetPointSize()——得到字体大小

void SetStyle(style)——设置字体样式

int GetStyle()——得到字体样式

double GetFontAscent()——得到字体ascent

double GetFontDescent()——得到字体descent

double GetFontHeight()——得到字体height

double GetMouseX()——得到鼠标位置x

double GetMouseY()——得到鼠标位置y

bool MouseButtonIsDown()——判断鼠标是否按下

void WaitForMouseDown()——直到鼠标按下

WaitForMouseUp()——直到鼠标抬起

bool HasColor()——判断是否有颜色

void SetPenColor(color)——设定笔颜色

string GetPenColor()——得到笔颜色

void DefineColor(name,red,green,blue)——设定颜色名字和三原色

void SetPenSize(size)——设置笔的大小

int GetPenSize()——得到笔的大小

void SetEraseMode(mode)——改为橡皮擦的笔

bool GetEraseMode()——得到笔的性质

void SetWindowTitle(title)——设置图像标题

string GetWindowTitle()——得到图像标题

void UpdateDisplay()——更新图像

void Pause(seconds)——图像暂停seconds秒

void ExitGraphics()——停止作画

void SaveGraphicsState()——保存作图

void RestoreGraphicsState()——回溯作图，与上式子一起用

double GetFullScreenWidth()——整个屏幕宽度

double GetFullScreenHeight()——整个屏幕高度

void SetWindowSize(width, height)——设置图像宽度、高度

double GetXResolution()——宽的分辨率

double GetYResolution()——高的分辨率

linkedlistADT NewLinkedList()——建立一个新链表，返回链表头

void FreeLinkedList(head)——释放链表

linkedlistADT SearchNode(head, void \*obj,bool (\*equalfunptr)(void \*obj1, void \*obj2))——判断链表中是否存在obj

linkedlistADT InsertNode(head, linkedlistADT nodeptr, void \*obj)——加入obj，nodeptr=NULL则插入尾部

linkedlistADT DeleteNode(head, void \*obj,bool (\*equalfunptr)(void \*obj1, void \*obj2))——删除obj

void TraverseLinkedList(head, void (\*traversefunptr)(void \*obj))——便利链表

void \*ithNodeobj(head, int index)——返回第index的指针

linkedlistADT NextNode(head, nodeptr)——返回nodeptr的下一个指针

void \*NodeObj( head, nodeptr)——返回nodeptr的dataptr

void Randomize()——初始化时间

int RandomInteger(low,high)——low-high的随机整数

double RandomReal(low, high)——low-high的随机实数

bool RandomChance(p)——随机0-1实数与p比较

int GetInteger()——得到整形

int GetLong()——得到长整型

int GetChar()——得到实数

string GetLine()——得到字符串

string Concat(s1, s2)——拼接s1和s2

char IthChar(s,i)——返回字符串第i号字母

string SubString(s, p1, p2)——返回p1到p2的子串

string CharToString(ch)——字符变字符串

int StringLength(s)——返回字符串长度

string CopyString(s)——复制字符串

bool StringEqual(s1, s2)——判断s1和s2相不相等

int StringCompare(s1,s2)——比较s1和s2大小

int FindChar(ch, text, start)——找到text中start后第一个ch位置

int FindString(str,text,start)——找到text中start后第一个str位置

string ConvertToLowerCase(s)——字符串变小写

string ConvertToUpperCase(s)——字符串变大写

string IntegerToString(n)——整数变字符串

int StringToInteger(s)——字符串变整数

string RealToString(d)——实数变字符串

double StringToReal(s)——字符串变实数

void InitGUI()——初始化GUI

mySetPenColor(color)——改笔颜色

void setButtonColors(char \*frame, char\*label, char \*hotFrame, char \*hotLabel, int fillflag)——改button颜色，frame框颜色，label文字颜色，hot~画过颜色，fillflag是否填充

setMenuColors，setTextBoxColors一样

void usePredefinedColors(k) void usePredefinedButtonColors(k);

void usePredefinedMenuColors(k) void usePredefinedTexBoxColors(k)

——使用预订颜色，第一个设置三个

void drawLabel(double x, double y, char \*label)——显示字符串标签

void drawRectangle(x, y, w, h, fillflag)——画矩形

void drawBox(x, y, w, h, fillflag, char \*label, char xalignment, char \*labelColor)

——画矩形，label填充字符串，xalignment(L左对齐R右对齐其他居中)

void uiGetMouse(int x, int y, int button, int event);

void uiGetKeyboard(int key, int event) void uiGetChar(int ch);

——获得用户的鼠标/键盘/文本的输入

int button(int id, double x, double y, double w, double h, char \*label);

button显示一个按钮（button）

if( button(GenUUID(0),x,y,w,h,label) ) {

执行语句，响应用户按下该按钮

}

id - 唯一号 x,y - 位置 w,h - 宽度和高度 label - 按钮上的文字标签

返回值 0 - 用户没有点击（按下并释放）按钮 1 - 点击了按钮

int menuList(int id, double x, double y, double w, double wlist, double h, char \*labels[], int n);

显示一组菜单

用法：choice = menuList(GenUUID(0),x,y,w,h,labels,n);

id - 菜单的唯一号 x,y - 菜单的位置 w,h - 菜单项的大小 wlist - 菜单列表的宽度

labels[] - 标签文字，[0]是菜单类别，[1...n-1]是菜单列表 n - 菜单项的个数

返回值

-1 - 用户没有点击（按下并释放）按钮

>=0 - 用户选中的菜单项 index （在labels[]中）

\* 应用举例：

char \* menuListFile[] = {"File", "Open | Ctrl-O", "Close","Exit | Ctrl-E"};

int selection;

selection = menuList(GenUIID(0), x, y, w, wlist, h, menuListFile, sizeof(menuListFile)/sizeof(menuListFile[0]));

if( selection==3 )

exit(-1); // user choose to exit

void drawMenuBar(double x, double y, double w, double h)——

用菜单的颜色，画一个矩形，位置(x,y),宽高(w,h)一般 w 等于窗口的宽度，h 和菜单项的高度匹配

int textbox(int id, double x, double y, double w, double h, char textbuf[], int buflen);

功能：显示一个文本编辑框，显示和编辑文本字符串

用法：textbox(GenUUID(0),x,y,w,h,textbuf,buflen);

或者if( textbox(GenUUID(0),x,y,w,h,textbuf,buflen) ) {

文本字符串被编辑修改了，执行相应语句}

id - 唯一号，一般用GenUUID(0), 或用GenUUID（k)（k是循环变量）

x,y - 文本框位置 w,h - 文本框的宽度和高度

textbuf - 被编辑的文本字符串（以\0结尾）buflen - 可存储的文本字符串的最大长度

返回值 0 - 文本没有被编辑 1 - 被编辑了

registerCharEvent(CharEventProcess); 字符

registerKeyboardEvent(KeyboardEventProcess);鼠标

registerMouseEvent(MouseEventProcess); 键盘

registerTimerEvent(TimerEventProcess); 定时器