浙江大学

程 序 设 计 专 题

大 程 序 报 告

2019~2020 春夏学期 2020 年 6 月 9 日

报告撰写注意事项

1） 图文并茂。文字通顺，语言流畅，无错别字。

2） 书写格式规范，排版良好，内容完整。

3） 存在拼凑、剽窃等现象一律认定为抄袭；0 分

4） 蓝色文字为说明，在最后提交的终稿版本，请删除这些文字。

目录

小型算法流程图绘制工具大程序设计

1 大程序简介

1.1 选题背景及意义

在对 libgraphics 与 simpleGUI 库有了更多学习了解之后，小组尝试通过 dev c++，以小组合作的方式设计一个流程图绘制工具，在设计、实现等过程中设计出一个便捷高效、实用性强的小型算法流程图绘制工具，同时增进对所学知识的掌握与理解，丰富程序设计与合作经验。

1.2 目标要求

该流程图绘制工具需要具有能够绘制箭头、矩形、圆角矩形、菱形、直角箭头、文本框等的功能，并且能够实现选择、复制、粘贴、删除、改变（拖移与缩放）等基本操作；在实现上述指令的过程中，程序运行流畅并具有良好反馈；能够新建、关闭作品，对绘制的作品能够进行保存和再次打开，基本实现流程图制作过程中需要实现的功能。

为了进一步提高使用体验，程序应具有可操作性强的菜单界面，直观的当前状态信息显示，便捷、符合用户使用习惯的快捷键系统等元素；对于绘制的图形，能够自定义线条颜色、画笔大小、填充颜色等；能够播放背景音乐，具有用户使用帮助，另外能够将绘制作品以图片形式保存，使其能够在软件之外展示。

1.3 术语说明

2 功能需求分析

本软件用于算法流程图的绘制，本小组结合日常软件的使用经验，从用户体验的角度出发考虑，分析功能需求如下。

首先作为流程图绘制软件，最基础的便是箭头、矩形、菱形、文本等基础图形元素的绘制。这样的绘制功能应通过鼠标的移动完成，以保证过程的精确性和高效性。为保证流程图的美观性、可读性和风格性，还应该支持对于图形元素颜色、线条粗细、填充等属性的自定义选择。

其次，在绘制过程当中，应当支持在现有已绘制基础上的即时修改以及编辑功能以提高工作效率。这便产生了移动、复制、粘贴、删除等编辑功能的需求。Libgraphics 库所提供的绘图功能仅支持在图形窗口进行绘制并显示，并不能对显

示内容进行数据上的存储和修改，要实现编辑功能，需要将图形通过结构体和链表的数据结构方式存储在内存当中。于是，绘图和修改过程实质上是对结构体数据的赋值修改。由于不同种类的图形所需的必要属性信息是不同的，为了实现链表的统一性，在链表中需要使用 void 指针类型变量来指向不同种类的结构体。为了避免操作失误改变已绘制的流程图，我们引入了选中功能，并且规定只能对已选中的图形进行编辑修改操作，提高了过程的准确性。

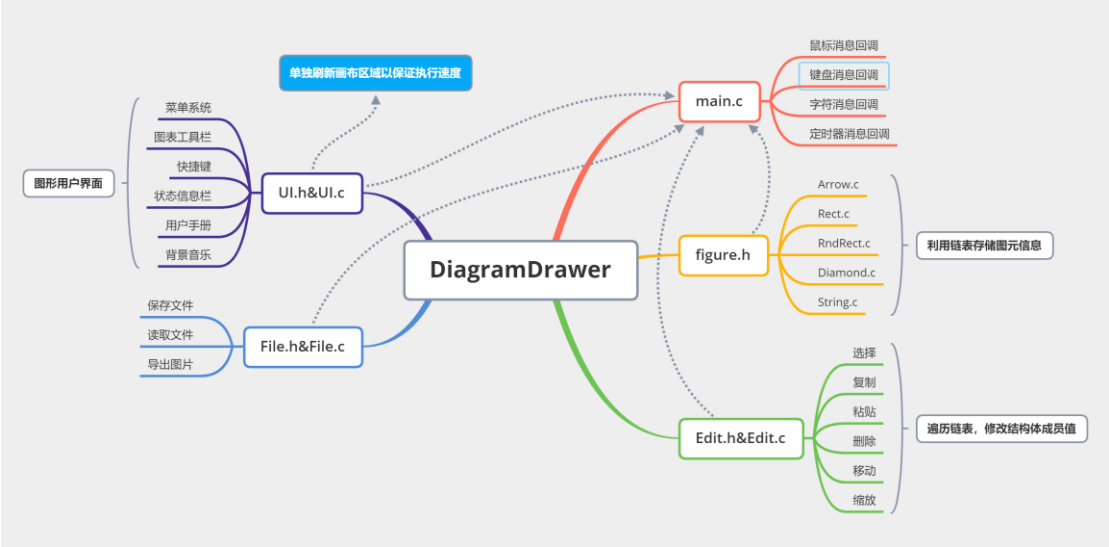
对于单个流程图的绘制，经常需要分多次进行完成，同时也存在多人合作分工完成一个流程图的情况，此时便产生了对于工程的保存与读取功能的需求。将部分完成的工程保存在一个文件当中，并且实现文件的读取，就能实现同一工程的阶段性完成，同时仅需将保存所得文件分享给合作伙伴，就能方便快捷的实现工程分工合作。

为了便利性考虑，绘制完成的流程图投入使用应该要能够脱离本软件单独查看，而对于工程保存所产生的文件为了可读取将只保存链表数据，缺乏可视性。故引入图片导出功能。

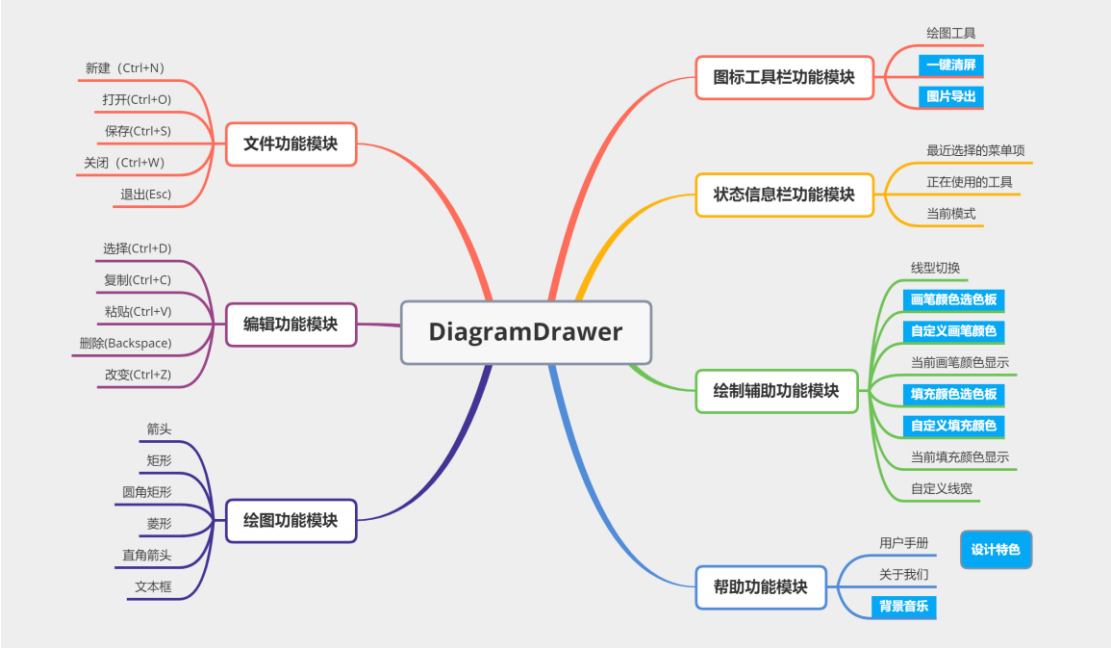
此类软件通常需要使用屏幕刷新，而 libgrphics 库所提供的绘图函数运行速度较慢，且 UI 界面的绘制量交较大，若每次都对整个界面进行刷新会使得程序运行效率低下，在绘制过程中产生卡顿，严重影响用户体验，所以应该实现对于画布和 UI 界面的刷新分离功能，减少 UI 界面的刷新次数。

3 程序开发设计

3.1 总体架构设计



3.2 功能模块设计



3.3 数据结构设计

（1） 整型变量及整形数组
key：键盘按键回调；
button：鼠标按键回调；
event：鼠标、键盘等消息事件；
currentTool：当前选择工具；
currentPenSize：当前画笔宽度；
quit_confirm：确认退出标记；
i：循环计数器；
x, y：当前鼠标位置；
pasteTime：粘贴次数；
isPlayingMusic：音乐播放标记；
timerseconds：光标闪烁周期；
num[FLIST]：各类图形个数；
cnt：计数器；
（2） 双精度浮点型变量
winwidth, winheight：窗口尺寸；
（3） 布尔型变量
isDrawing：绘图模式标记；
isSelecting：选择模式标记；
isChanging：改变模式标记；
newFile：新文件标记；
isSaved：文件保存标记；

isCursorBlink：光标闪烁标记；
isEdite：文本编辑状态标记；
isDraw：绘制状态标记；
isChange：改变状态标记 ；
isMove：移动状态标记；
isCtrl：Ctrl 键标记；

（4） 字符串常量、字符串变量及指向字符串的指针数组

currentPenColor：当前画笔颜色；
currentFillColor：当前填充颜色；
currentPenType：当前线型；
“solid”：实线；
“dashed”：虚线；
selectedColor：选中状态颜色；
selectedLabel：当前选中菜单项；
textbuf[TEXTLEN+1]：当前文本缓冲区；
char *colorName[]：颜色名称；
“Black”：黑；
“White”：白；
“Red”：红；
“Blue”：蓝；
“Green”：绿；
“Yellow”：黄；
“Violet”：紫；
“Orange”：橘；
“Brown”：棕；
“Cyan”：青；
“Magenta”：品红；
“LightGray”：浅灰；
char *ToolName[]：工具名称；

（5） 结构与链表节点

```
typedef struct {/*箭头类型*/
  double x1, y1;/*起点坐标*/
      double x2, y2;/*终点坐标*/
      int penSize; /*笔粗细*/
      string penType;/*笔类型：solid, dashed */
      string penColor; /*笔颜色*/
      bool isSelected; /*选中*/
} *myArrow;

typedef struct {/*矩形类型*/
  double x1, y1;/*左上角坐标*/
      double x2, y2;/*右下角坐标*/
      int penSize; /*笔粗细*/
      string penType;/*笔类型：solid, dashed */
```

```
        string penColor; /*笔颜色*/
        string fillColor;/*填充颜色*/
        bool isSelected; /*选中*/
} *myRect;

typedef struct {/*圆角矩形类型*/
  double x1, y1;/*左上角坐标*/
        double x2, y2;/*右下角坐标*/
        int penSize; /*笔粗细*/
        string penType;/*笔类型：solid, dashed */
        string penColor; /*笔颜色*/
        string fillColor;/*填充颜色*/
        bool isSelected; /*选中*/
} *myRndRect;

typedef struct {/*菱形类型*/
  double x1, y1;/*左上角坐标*/
        double x2, y2;/*右下角坐标*/
        int penSize; /*笔粗细*/
        string penType;/*笔类型：solid, dashed */
        string penColor; /*笔颜色*/
        string fillColor;/*填充颜色*/
        bool isSelected; /*选中*/
} *myDiamond;

typedef struct {
  double x1, y1;/*左上角坐标*/
        double x2, y2;/*右下角坐标*/
        double angle;/*角度*/
        int penSize; /*笔粗细*/
        string penType;/*笔类型：solid, dashed */
        string penColor; /*笔颜色*/
        bool direction; /*方向：顺时针  1，逆时针  0*/
        bool isSelected; /*选中*/
} *myRtArrow;

typedef struct {/*文本类型*/
  string text;/*文本指针*/
  double x, y;/*文本显示起始位置坐标*/
        int pointSize; /*文字大小*/
        string penColor; /*颜色*/
        bool isSelected; /*选中*/
        int cursor; /*光标位置*/
        bool isBlink; /*光标是否处于闪烁状态*/
```

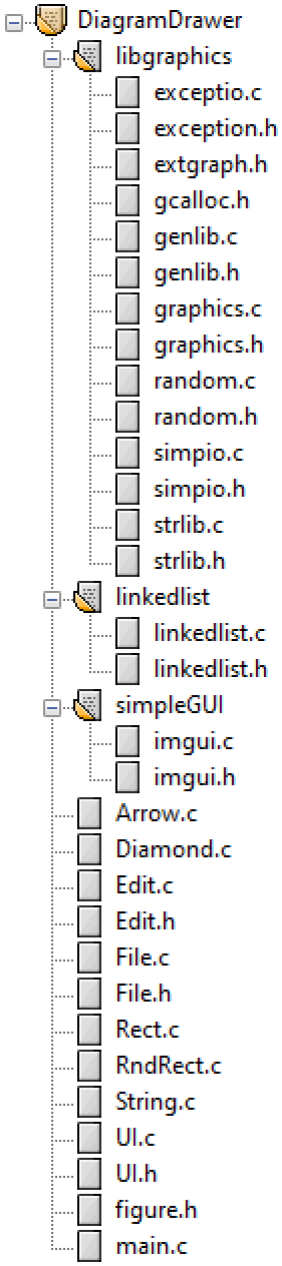
```
} *myString;
```

（6） 宏定义

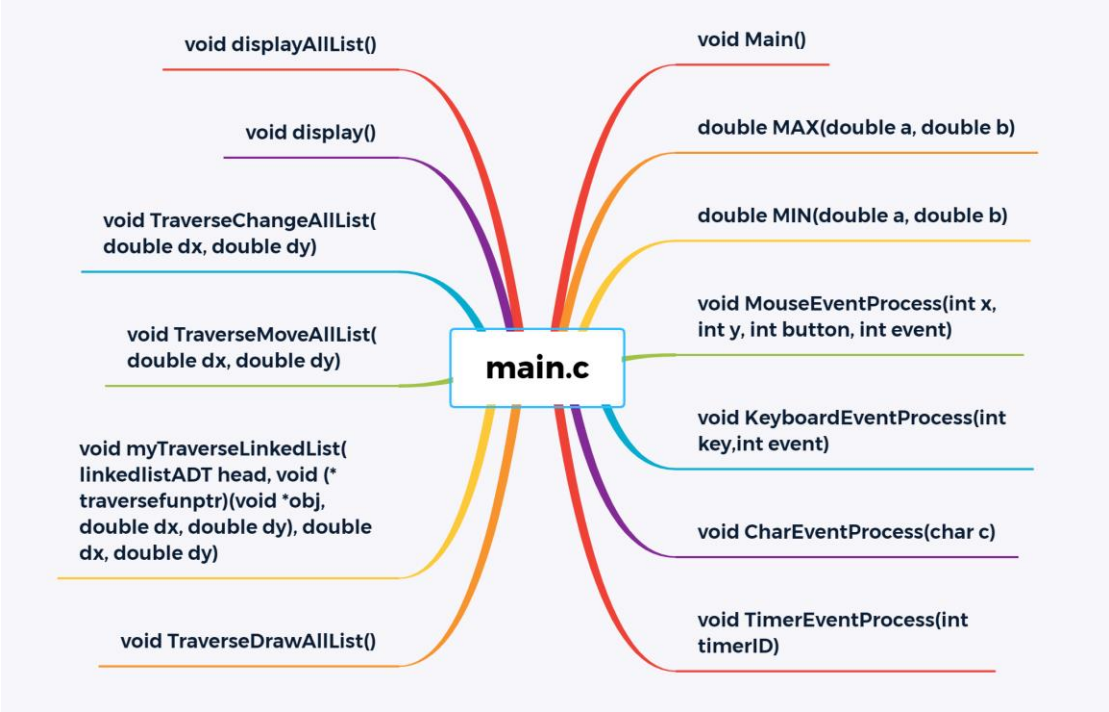
```
#define FLIST           6
#define ARROW           0
#define RECT            1
#define RNDRECT         2
#define DIAMOND         3
#define RTARROW         4
#define STRING          5

#define PI acos(-1)    /*圆周率*/
#define DegToReg(x) (x) * PI / 180.0    /*角度制转弧度制*/
#define RegToDeg(x) (x) * 180.0 / PI    /*弧度制转角度制*/

#define CURSOR "_"              /*光标符号*/
#define CURSOR_BLINK    1       /*光标闪烁定时器事件标志号*/
#define TEXTLEN    100          /*文本长度*/
```

3.4 源代码文件组织设计

<文件目录结构>


DiagramDrawer
libgraphics
exceptio.c
exception.h
extgraph.h
gcalloc.h
genlib.c
genlib.h
graphics.c
graphics.h
random.c
random.h
simpio.c
simpio.h
strlib.c
strlib.h
linkedlist
linkedlist.c
linkedlist.h
simpleGUI
imgui.c
imgui.h
Arrow.c
Diamond.c
Edit.c
Edit.h
File.c
File.h
Rect.c
RndRect.c
String.c
UI.c
UI.h
figure.h
main.c

1） 文件函数结构


main.c
void displayAllList()
void display()
void TraverseChangeAllList(double dx, double dy)
void TraverseMoveAllList(double dx, double dy)
void myTraverseLinkedList(linkedlistADT head, void (* traversefunptr)(void *obj, double dx, double dy), double dx, double dy)
void TraverseDrawAllList()
void Main()
double MAX(double a, double b)
double MIN(double a, double b)
void MouseEventProcess(int x, int y, int button, int event)
void KeyboardEventProcess(int key,int event)
void CharEventProcess(char c)
void TimerEventProcess(int timerID)

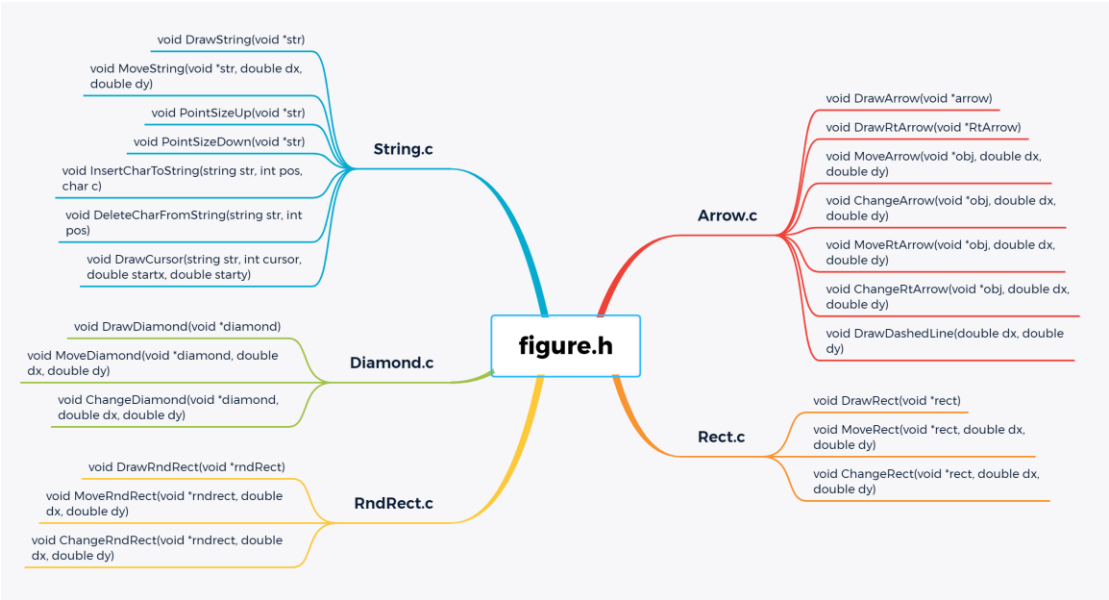
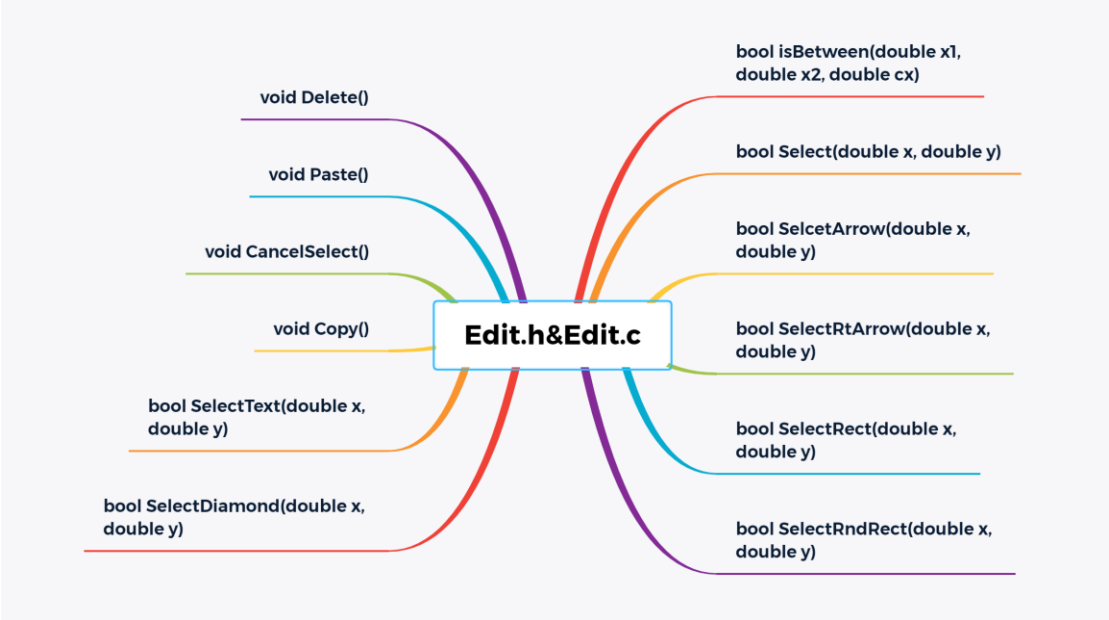
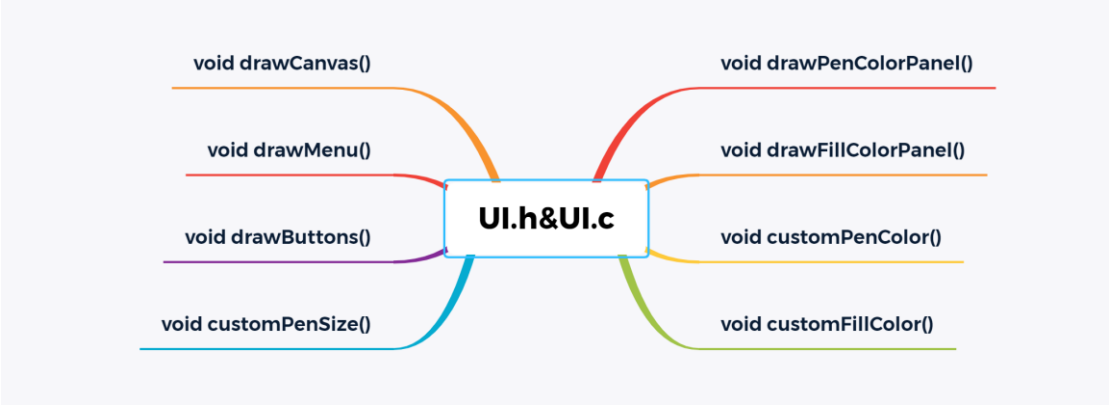


figure.h
String.c
void DrawString(void *str)
void MoveString(void *str, double dx, double dy)
void PointSizeUp(void *str)
void PointSizeDown(void *str)
void InsertCharToString(string str, int pos, char c)
void DeleteCharFromString(string str, int pos)
void DrawCursor(string str, int cursor, double startx, double starty)
Diamond.c
void DrawDiamond(void *diamond)
void MoveDiamond(void *diamond, double dx, double dy)
void ChangeDiamond(void *diamond, double dx, double dy)
RndRect.c
void DrawRndRect(void *rndRect)
void MoveRndRect(void *rndrect, double dx, double dy)
void ChangeRndRect(void *rndrect, double dx, double dy)
Arrow.c
void DrawArrow(void *arrow)
void DrawRtArrow(void *RtArrow)
void MoveArrow(void *obj, double dx, double dy)
void ChangeArrow(void *obj, double dx, double dy)
void MoveRtArrow(void *obj, double dx, double dy)
void ChangeRtArrow(void *obj, double dx, double dy)
void DrawDashedLine(double dx, double dy)
Rect.c
void DrawRect(void *rect)
void MoveRect(void *rect, double dx, double dy)
void ChangeRect(void *rect, double dx, double dy)





2）多文件构成机制

main.c 为主文件，包含所有其他源文件以及 libgraphics 库，C 语言标准库。

Figure.h 中包含有 Arrow.c、Rect.c、RndRect.c、Diamond.c、String.c 中函数的声明以及结构体定义，使用头文件保护以保证只被编译一次；

Edit.h 中包含有 Edit.c 中定义的选择、复制、粘贴、删除函数的声明及全局变量，使用头文件保护以保证只被编译一次；

UI.h 中包含有 UI.c 中定义的绘画函数的声明及全局变量，使用头文件保护以保证只被编译一次；

File.h 中包含有 File.c 中定义的保存、读取、导出函数的声明及全局变量，使用头文件保护以保证只被编译一次；

3.5 函数设计描述

3.5.1 main.c

1) 函数原型：void Main()
   功能描述：Windows 应用程序主函数
   参数描述：无
   返回值描述：无
   重要局部变量定义：无
   重要局部变量用途描述：无
   函数算法描述： 命名窗口，设置窗口大小并初始化，注册鼠标、键盘、字符、定时器回调函数，新建链表，设定部分全局变量初值，刷新屏幕区域。
2) 函数原型：double MAX(double a, double b)
   功能描述：获得两个浮点数中较大的一个
   参数描述：
      a：待比较的第一个浮点数
      b：待比较的第二个浮点数
   返回值描述：返回两个浮点数中较大的一个
   重要局部变量定义：无
   重要局部变量用途描述：无
   函数算法描述：判断浮点数大小
3) 函数原型：double MIN(double a, double b)
   功能描述：获得两个浮点数中较小的一个
   参数描述：
      a：待比较的第一个浮点数
      b：待比较的第二个浮点数
   返回值描述：返回两个浮点数中较小的一个
   重要局部变量定义：无
   重要局部变量用途描述：无
   函数算法描述： 判断浮点数大小
4) 函数原型：void MouseEventProcess(int x, int y, int button, int event)
   功能描述：鼠标消息回调
   参数描述：
      x：鼠标位置的横坐标
      y：鼠标位置的纵坐标
      button：鼠标按键
      event：鼠标事件
   返回值描述：无
   重要局部变量定义：
      static bool isDraw;
      static bool isMove;
      static bool isChange;
      static double bx, by;

double lx, ly;

重要局部变量用途描述：

static bool isDraw：标记是否处于左键按下绘制状态；

static bool isMove：标记是否处于左键按下移动状态；

static bool isChange：标记是否处于右键按下缩放状态；

static double bx, by：记录鼠标按键时的位置坐标；

double lx, ly：记录当前鼠标位置坐标；

函数算法描述： 若处于绘制、选择、移动或缩放状态，则鼠标有事件时单独刷新画布区域，否则刷新整个屏幕；绘图模式下鼠标左键按下进入绘制状态，申请新的结构体空间并给结构体赋初值，插入到对应链表尾部，鼠标移动时更改成员变量的值，左键抬起时结束绘制状态；编辑模式下选择菜单或快捷键进入选择状态，鼠标左键按下时判断鼠标位置，遍历链表判断是否处于图形范围内进行选中，选中后选择菜单或快捷键，左键按下进入移动状态，鼠标移动更改成员变量的值，左键抬起结束移动状态，右键按下进入缩放状态，鼠标移动更改成员变量的值，右键抬起结束缩放状态。

5) 函数原型：void KeyboardEventProcess(int key,int event)

功能描述：键盘消息回调，实现快捷键

参数描述：

key：键盘按键

event：键盘事件

返回值描述：无

重要局部变量定义：无

重要局部变量用途描述：无

函数算法描述： Ctrl 键按下时更改 isCtrl 的值为 TRUE，抬起时更改为 FALSE；将选择菜单函数写入实现快捷键。

6) 函数原型：void CharEventProcess(char c)

功能描述：字符消息回调函数，实现文本编辑

参数描述：

c：输入字符

返回值描述：无

重要局部变量定义：

int len;

重要局部变量用途描述：

int len：记录当前文本缓冲区中的字符串长度；

函数算法描述：回车时退出文本输入状态，擦除光标，更改部分成员变量的值，将结构体插入到文本框链表尾部，注销闪烁定时器；回退或输入字符时擦除光标，更改文本缓冲区内容，重新显示光标。

7) 函数原型：void TimerEventProcess(int timerID)

功能描述：定时器消息回调，实现文本框光标闪烁

参数描述：

timerID：定时器标号

返回值描述：无

重要局部变量定义：无

重要局部变量用途描述：无

函数算法描述： 每 500ms 更改一次 isBlink 的值。

8) 函数原型：void TraverseDrawAllList()
功能描述： 绘制所有图形
参数描述：无
返回值描述：无
重要局部变量定义：无
重要局部变量用途描述：无
函数算法描述： 调用 linkedlist.c 中的 TraverseLinkedList 函数和各类图形的绘制函数。

9) 函数原型：void myTraverseLinkedList(linkedlistADT head, void (*traversefunptr)(void *obj, double dx, double dy), double dx, double dy)
功能描述：可以传送两个浮点型参数的遍历链表函数
参数描述：
head：链表的头结点；
(*traversefunptr)(void *obj, double dx, double dy)：函数指针；
dx：需传送的第一个浮点型参数；
dy：需传送的第二个浮点型参数；
返回值描述：无
重要局部变量定义：
linkedlistADT nodeptr;
重要局部变量用途描述：
linkedlistADT nodeptr：作为遍历链表的光标，存储当前指向的结构体；
函数算法描述： 若函数指针为空则返回；遍历链表各节点进行函数指针对应的操作。

10) 函数原型：void TraverseMoveAllList(double dx, double dy)
功能描述：移动链表中所有选中的图形
参数描述：
dx：横坐标改变量；
dy：纵坐标改变量；
返回值描述：无
重要局部变量定义：无
重要局部变量用途描述：无
函数算法描述：调用 myTraverseLinkedList 函数和各类图形的移动函数。

11) 函数原型：void TraverseChangeAllList(double dx, double dy)
功能描述：缩放链表中所有选中的图形
参数描述：
dx：横坐标改变量；
dy：纵坐标改变量；
返回值描述：无
重要局部变量定义：无
重要局部变量用途描述：无
函数算法描述：调用 myTraverseLinkedList 函数和各类图形的缩放函数。

12) 函数原型：void display()

功能描述：刷新屏幕
参数描述：无
返回值描述：无
重要局部变量定义：无
重要局部变量用途描述：无
函数算法描述：清屏后重新绘制用户界面和画布区域。

13) 函数原型：void displayAllList()
功能描述：刷新画布区域
参数描述：无
返回值描述：无
重要局部变量定义：无
重要局部变量用途描述：无
函数算法描述：绘制白色矩形覆盖画布区域，再调用 TraverseDrawAllList 函数重新绘制所有图形。

3.5.2 Arrow.c

1) 函数原型：void DrawArrow(void *arrow)
功能描述：绘制箭头
参数描述：
arrow：指向箭头数据结构体的指针
返回值描述：空
重要局部变量定义：
double angle;
double dx, dy;
int penSize;
string color
重要局部变量用途描述：
double angle：直线与水平夹角，用于箭头三角形的绘制；
double dx：水平宽度；
double dy：水平高度；
int penSize：保存先前画笔粗细，用于回档；
string color：保存先前画笔颜色，用于回档；
函数算法描述：由宽高和角度通过平面几何方法，利用 lib 库中的 DrawLine 函数画出直线和三角形箭头，若要画图形已被选中，则用品红色画。

2) 函数原型：void DrawRtArrow(void *RtArrow)
功能描述：绘制直角箭头
参数描述：
RtArrow：指向直角箭头数据结构体的指针
返回值描述：空
重要局部变量定义：
double angle;
double dx, dy;
int penSize;

string color;

重要局部变量用途描述：

double angle：折线第二段与水平线夹角，用于绘制箭头部分三角形；

double dx：水平宽度；

double dy：水平高度；

int penSize：保存先前画笔粗细，用于回档；

string color：保存先前画笔颜色，用于回档；

函数算法描述：以起点为远点，判断终点所在象限，以确定先画横还是先画竖以及重点的三角形箭头方向，若要画图形已被选中，则用品红色画。

3) 函数原型：void MoveArrow(void *obj, double dx, double dy)

功能描述：移动箭头

参数描述：

obj：所要编辑箭头的结构体指针；

dx：水平移动距离；

dy：竖直移动距离；

返回值描述：空

重要局部变量定义：无

重要局部变量用途描述：无

函数算法描述：由 dx 与 dy 对箭头结构体内起点和终点的坐标数据进行修改

4) 函数原型：void ChangeArrow(void *obj, double dx, double dy)

功能描述：缩放箭头

参数描述：

obj：索要编辑箭头的结构体指针；

dx：箭头末端水平移动距离；

dy：箭头末端竖直移动距离；

返回值描述：空

重要局部变量定义：无

重要局部变量用途描述：无

函数算法描述：由 dx 与 dy 对箭头结构体内终点的坐标数据进行修改

5) 函数原型：void MoveRtArrow(void *obj, double dx, double dy)

功能描述：移动直角箭头

参数描述：

obj：所要编辑的直角箭头的结构体指针；

dx：水平移动距离；

dy：竖直移动距离；

返回值描述：空

重要局部变量定义：无

重要局部变量用途描述：无

函数算法描述：由 dx 与 dy 对直角箭头结构体内起点和终点的坐标数据进行修改

6) 函数原型：void ChangeRtArrow(void *obj, double dx, double dy)

功能描述：缩放直角箭头

参数描述：

obj：所要编辑的直角箭头的结构体指针；

dx：箭头末端水平移动距离；
dy：箭头末端竖直移动距离；
返回值描述：空
重要局部变量定义：无
重要局部变量用途描述：无
函数算法描述：由 dx 与 dy 对直角箭头结构体内终点的坐标数据进行修改

7) 函数原型：void DrawDashedLine(double dx, double dy)
功能描述：画虚线（用法与 DrawLine() 函数相似）
参数描述：
dx：水平宽度；
dy：水平高度；
返回值描述：空
重要局部变量定义：
double angle, Length, currentLen, DashLen;
double InitX, InitY;
double cx, cy;
重要局部变量用途描述：
double angle：直线角度；
double Length：线的总长；
double currentLen：记录已画的长度；
double DashLen：单个虚线长度和间隙长度；
double InitX：初始画笔横坐标；
double InitY：初始画笔纵坐标；
double cx：当前画笔横坐标；
double cy：当前画笔纵坐标；
函数算法描述：以画一小段线段再空出相等距离的方式画出虚线，重复循环，直到当前长度和总长差值小于一次循环，通过记算画完最后一小段的距离

3.5.3 Rect.c

1) 函数原型：void DrawRect(void *rect)
功能描述：绘制矩形
参数描述：
rect：指向矩形数据结构体的指针
返回值描述：空
重要局部变量定义：
double a，b;
int penSize;
string color
重要局部变量用途描述：
double a：矩形长；
double b：矩形宽；
int penSize：保存先前画笔粗细，用于回档；
string color：保存先前画笔颜色，用于回档；

函数算法描述：由长和宽绘制指定画笔粗细颜色、填充色的矩形，若要画图形已被选中，则用品红色画。

2) 函数原型：void MoveRect(void *rect, double dx, double dy)
功能描述：移动矩形
参数描述：
rect：所要编辑矩形的结构体指针
dx：水平移动距离；
dy：竖直移动距离；
返回值描述：空
重要局部变量定义：无
重要局部变量用途描述：无
函数算法描述：由 dx 与 dy 对矩形结构体内左上角和右下角的坐标数据进行修改

3) 函数原型：void ChangeRect(void *rect, double dx, double dy)
功能描述：缩放矩形
参数描述：
rect：所要编辑矩形的结构体指针
dx：水平移动距离；
dy：竖直移动距离；
返回值描述：空
重要局部变量定义：无
重要局部变量用途描述：无
函数算法描述：由 dx 与 dy 对矩形结构体内右下角的坐标数据进行修改

3.5.4 RndRect.c

1) 函数原型：void DrawRndRect(void *rndRect)
功能描述：绘制圆角矩形
参数描述：
rndRect：指向圆角矩形数据结构体的指针
返回值描述：空
重要局部变量定义：
double a，b，cr；
int penSize；
string color
重要局部变量用途描述：
double a：矩形长；
double b：矩形宽；
double cr：圆角半径；
int penSize：保存先前画笔粗细，用于回档；
string color：保存先前画笔颜色，用于回档；
函数算法描述：由长和宽计算圆角半径，绘制指定画笔粗细颜色、填充色的圆角矩形，若要画图形已被选中，则用品红色画。

2) 函数原型：void MoveRndRect(void *rndrect, double dx, double dy)

功能描述：移动圆角矩形
参数描述：
rndrect：所要编辑圆角矩形的结构体指针
dx：水平移动距离；
dy：竖直移动距离；
返回值描述：空
重要局部变量定义：无
重要局部变量用途描述：无
函数算法描述：由 dx 与 dy 对圆角矩形结构体内左上角和右下角的坐标数据进行修改

3) 函数原型：void ChangeRndRect(void *rndrect, double dx, double dy)
功能描述：缩放圆角矩形
参数描述：
rndrect：所要编辑圆角矩形的结构体指针
dx：水平移动距离；
dy：竖直移动距离；
返回值描述：空
重要局部变量定义：无
重要局部变量用途描述：无
函数算法描述：由 dx 与 dy 对圆角矩形结构体内右下角的坐标数据进行修改

3.5.5 Diamond.c

1) 函数原型：void DrawDiamond(void *diamond)
功能描述：绘制菱形
参数描述：
diamond：指向菱形数据结构体的指针
返回值描述：空
重要局部变量定义：
double a, b;
int penSize;
string color
重要局部变量用途描述：
double a：菱形矩形外框长；
double b：菱形矩形外框宽；
int penSize：保存先前画笔粗细，用于回档；
string color：保存先前画笔颜色，用于回档；
函数算法描述：由菱形的矩形外框长和宽绘制指定画笔粗细颜色、填充色的菱形，若要画图形已被选中，则用品红色画。

2) 函数原型：void MoveDiamond(void *diamond, double dx, double dy)
功能描述：移动菱形
参数描述：
diamond：所要编辑菱形的结构体指针
dx：水平移动距离；

dy：竖直移动距离；
返回值描述：空
重要局部变量定义：无
重要局部变量用途描述：无
函数算法描述：由 dx 与 dy 对菱形结构体（矩形外框）左上角和右下角的坐标数据进行修改

3) 函数原型：void ChangeDiamond(void *diamond, double dx, double dy)
功能描述：缩放菱形
参数描述：
diamond：所要编辑菱形的结构体指针
dx：水平移动距离；
dy：竖直移动距离；
返回值描述：空
重要局部变量定义：无
重要局部变量用途描述：无
函数算法描述：由 dx 与 dy 对菱形结构体（矩形外框）右下角的坐标数据进行修改

3.5.6 String.c

1) 函数原型：void DrawString(void *str)
功能描述：绘制文本框
参数描述：
str：指向文本框数据结构体的指针
返回值描述：无
重要局部变量定义：
string color;
int pointsize;
重要局部变量用途描述：
string color：记录当前画笔颜色；
int pointsize：记录当前字体大小；
函数算法描述： 移动画笔到文本框其实坐标，设置画笔颜色和大小，绘制字符串，恢复原本画笔颜色和大小。

2) 函数原型：void MoveString(void *str, double dx, double dy)
功能描述：移动文本框
参数描述：
str：所要移动的文本框数据结构体指针
dx：横坐标改变量
dy：纵坐标改变量
返回值描述：无
重要局部变量定义：无
重要局部变量用途描述：无
函数算法描述：更改文本框起始坐标的值。

3) 函数原型：void PointSizeUp(void *str)

功能描述：放大文本框字体
参数描述：
str：所要编辑的文本框数据结构体指针
返回值描述：无
重要局部变量定义：无
重要局部变量用途描述：无
函数算法描述：若文本框字体大小小于 50，增大 5。

4) 函数原型：void PointSizeDown(void *str)
功能描述：缩小文本框字体
参数描述：
str：所要编辑的文本框数据结构体指针
返回值描述：无
重要局部变量定义：无
重要局部变量用途描述：无
函数算法描述： 若文本框字体大小大于 50，减小 5。

5) 函数原型：void InsertCharToString(string str, int pos, char c)
功能描述：在字符串的光标后插入一个字符
参数描述：
str：字符串；
pos:插入字符的位置
c：待插入的字符
返回值描述：无
重要局部变量定义：无
重要局部变量用途描述：无
函数算法描述： 将光标后的所有字符向后移动一位，将 c 赋给光标处的字符。

6) 函数原型：void DeleteCharFromString(string str, int pos)
功能描述：在字符串的光标前删除一个字符
参数描述：
str：字符串；
pos:删除字符的位置
返回值描述：无
重要局部变量定义：无
重要局部变量用途描述：无
函数算法描述：将光标后的所有字符向前移动一位。

7) 函数原型：void DrawCursor(string str, int cursor, double startx, double starty)
功能描述：绘制光标
参数描述：
str：字符串；
cursor：光标位置；
startx：起始横坐标；
starty：起始纵坐标；
返回值描述：无

重要局部变量定义：无
重要局部变量用途描述：无
函数算法描述： 若光标在文本长度范围之外则返回；移动画笔到光标应在位置，单独绘制。

3.5.7 Edit.c

1) 函数原型：bool isBetween(double x1, double x2, double cx)
功能描述：判断数值上 cx 是否在 x1 和 x2 之间
参数描述：
double x1：判定范围
double x2：判定范围
double cx：所需判断的数值
返回值描述：若 cx 在 x1 和 x2 之间，返回 TRUE，否则返回 FALSE
重要局部变量定义：无
重要局部变量用途描述：无
函数算法描述：首先判断 x1 和 x2 的大小关系以确定区间上下限，判断 cx 是否在区间内
2) 函数原型：bool Select(double x, double y)
功能描述：根据当前鼠标位置选中一个图形
参数描述：
x：当前鼠标横坐标
y：当前鼠标纵坐标
返回值描述：若成功选中，返回 TRUE；若未能选中，返回 FALSE
重要局部变量定义：无
重要局部变量用途描述：无
函数算法描述：按顺序调用各种类型图形的选中函数，若选中某种图形，返回
3) 函数原型：bool SelcetArrow(double x, double y)
功能描述：根据当前鼠标位置选中一个箭头
参数描述：
x：当前鼠标横坐标
y：当前鼠标纵坐标
返回值描述：若成功选中，返回 TRUE；若未能选中，返回 FALSE
重要局部变量定义：
double angle;
double dx, dy;
bool haveSelected;
重要局部变量用途描述：
double angle：直线与水平方向夹角，用于记算选中判定范围
double dx：水平宽度
double dy：水平高度
bool haveSelected：记录是否已经选中一个箭头，并作为返回值
函数算法描述：通过水平宽度正负和垂直高度正负值和比例关系得出 angle

数值。斜率不存在：选中范围为箭头线段左右平移 0.15 所得矩形。斜率存在：选中范围为箭头线段上下平移 0.15 所得矩形。遍历链表，判断鼠标坐标是否在当前节点图形的选中范围内，若在，将该节点的 isSelected 变量赋值 TRUE，若不在，进入下一个节点。

4) 函数原型：bool SelectRtArrow(double x, double y)
功能描述：根据当前鼠标位置选中一个直角箭头
参数描述：
x：当前鼠标横坐标
y：当前鼠标纵坐标
返回值描述：若成功选中，返回 TRUE；若未能选中，返回 FALSE
重要局部变量定义：
bool haveSelected;
double dx, dy;
重要局部变量用途描述：
bool haveSelected：记录是否已经选中一个箭头，并作为返回值
double dx：水平宽度
double dy：水平高度
函数算法描述：将判定范围定以为直角箭头中两条线段两端各延长 0.2 所得线段为中位线的矩形。遍历链表，判断鼠标坐标是否在当前节点图形的选中范围内，若在，将该节点的 isSelected 变量赋值 TRUE，若不在，进入下一个节点。

5) 函数原型：bool SelectRect(double x, double y)
功能描述：根据当前鼠标位置选中一个矩形
参数描述：
x：当前鼠标横坐标
y：当前鼠标纵坐标
返回值描述：若成功选中，返回 TRUE；若未能选中，返回 FALSE
重要局部变量定义：
bool haveSelected;
重要局部变量用途描述：
bool haveSelected：记录是否已经选中一个箭头，并作为返回值
函数算法描述：选中范围定为矩形本身的范围。遍历链表，判断鼠标坐标是否在当前节点图形的选中范围内，若在，将该节点的 isSelected 变量赋值 TRUE，若不在，进入下一个节点。

6) 函数原型：bool SelectRndRect(double x, double y)
功能描述：根据当前鼠标位置选中一个圆角矩形
参数描述：
x：当前鼠标横坐标
y：当前鼠标纵坐标
返回值描述：若成功选中，返回 TRUE；若未能选中，返回 FALSE
重要局部变量定义：
bool haveSelected;
重要局部变量用途描述：
bool haveSelected：记录是否已经选中一个箭头，并作为返回值

函数算法描述：选中范围定为将该圆角矩形补全后的矩形的范围。遍历链表，判断鼠标坐标是否在当前节点图形的选中范围内，若在，将该节点的 isSelected 变量赋值 TRUE，若不在，进入下一个节点。

7) 函数原型：bool SelectDiamond(double x, double y)
功能描述：根据当前鼠标位置选中一个菱形
参数描述：
x：当前鼠标横坐标
y：当前鼠标纵坐标
返回值描述：若成功选中，返回 TRUE；若未能选中，返回 FALSE
重要局部变量定义：
double angle;
bool haveSelected;
重要局部变量用途描述：
double angle：菱形左上角线段与水平方向夹角，用于记算选中范围
bool haveSelected：记录是否已经选中一个箭头，并作为返回值
函数算法描述：选中范围定位菱形本身范围。遍历链表，判断鼠标坐标是否在当前节点图形的选中范围内，若在，将该节点的 isSelected 变量赋值 TRUE，若不在，进入下一个节点。

8) 函数原型：bool SelectText(double x, double y)
功能描述：根据当前鼠标位置选中一个文本框
参数描述：
x：当前鼠标横坐标
y：当前鼠标纵坐标
返回值描述：若成功选中，返回 TRUE；若未能选中，返回 FALSE
重要局部变量定义：
bool haveSelected;
int ori_pointSize;
重要局部变量用途描述：
bool haveSelected：记录是否已经选中一个箭头，并作为返回值
int ori_pointSize：记录当前点大小，用于判定完成后复原
函数算法描述：判定范围定位矩形，高度为字体高度，宽度为显示出的字符串长度。遍历链表，判断鼠标坐标是否在当前节点图形的选中范围内，若在，将该节点的 isSelected 变量赋值 TRUE，若不在，进入下一个节点。

9) 函数原型：void Copy()
功能描述：将所有已选中图形复制
参数描述：无
返回值描述：无
重要局部变量定义：
myArrow Arrow, Arrow_add;
myRtArrow RtArrow, RtArrow_add;
myRect Rect, Rect_add;
myRndRect RndRect, RndRect_add;
myDiamond Diamond, Diamond_add;
myString String, String_add;

linkedlistADT Readptr;

重要局部变量用途描述：

每一种图形对应的结构体指针各两个，一个用于承接 void 指针，一个用于加入到用于存储复制所得图形的链表

linkedlistADT Readptr：链表节点，用于记录当前正在读的节点

函数算法描述：首先清空复制链表，清除上一次复制。遍历 list 链表，若当前节点结构体的 isSelected 变量为 TRUE，将这个节点结构体内所有变量赋值给对应类型的用于插入复制链表的结构体，将所得结构体插入复制链表。

10) 函数原型：void CancelSelect()

功能描述：取消所有图形的选中

参数描述：无

返回值描述：无

重要局部变量定义：

int i;

重要局部变量用途描述：

int i：循环变量，用于遍历所有链表

函数算法描述：遍历所有链表，将每个节点结构体内的 isSelected 变量的值赋为 FALSE

11) 函数原型：void Paste()

功能描述：粘贴上一次复制的图形

参数描述：无

返回值描述：无

重要局部变量定义：

myArrow Arrow, Arrow_add;
myRtArrow RtArrow, RtArrow_add;
myRect Rect, Rect_add;
myRndRect RndRect, RndRect_add;
myDiamond Diamond, Diamond_add;
myString String, String_add;
linkedlistADT Readptr;

重要局部变量用途描述：

每一种图形对应的结构体指针各两个，一个用于承接 void 指针，一个用于加入到用于存储复制所得图形的链表

linkedlistADT Readptr：链表节点，用于记录当前正在读的节点

函数算法描述：遍历复制链表，将这个节点结构体内所有变量赋值给对应类型的用于插入 list 链表的结构体，将所得结构体插入 list 链表。

12) 函数原型：void Delete()

功能描述：删除当前选中的图形

参数描述：无

返回值描述：无

重要局部变量定义：

myArrow Arrow, Arrow_add;
myRtArrow RtArrow, RtArrow_add;

myRect Rect, Rect_add;
myRndRect RndRect, RndRect_add;
myDiamond Diamond, Diamond_add;
myString String, String_add;
linkedlistADT Readptr, preNodeptr;

重要局部变量用途描述：

linkedlistADT nodeptr：链表节点，用于记录当前正在读的节点
linkedlistADT preNodeptr：当前读取节点的前一个节点，用于删除后的链表连接

函数算法描述：遍历 list 链表，若当前节点结构体的 isSelected 变量值为 TRUE，将该节点的 next 变量赋值给前一个节点，将这个节点的内存释放。

3.5.8 File.c

1) 函数原型：void SaveFile(linkedlistADT list[])

功能描述：保存当前已绘制的图形到 txt 文件

参数描述：

list[]：结构体二级指针，用于指向 main.c 当中的 list 数组存储的链表

返回值描述：无

重要局部变量定义：

FILE *fp;
linkedlistADT nodeptr;
char FileName[50];

重要局部变量用途描述：

FILE *fp：用于打开文件
linkedlistADT nodeptr：链表节点，用于记录当前正在读的节点
char FileName[50]：要保存的文件名

函数算法描述：以只写形式打开一个文件，先将链表中不同类型的图形数据输出至文件，便于下次读取。遍历链表，按一定格式将所有图形结构体的信息输出至 txt 文件内。

2) 函数原型：void LoadFile(linkedlistADT list[])

功能描述：读取先前保存的 txt 文件并插入至 list 链表

参数描述：

list[]：结构体二级指针，用于指向 main.c 当中的 list 数组存储的链表

返回值描述：无

重要局部变量定义：

FILE *fp;
linkedlistADT nodeptr;
char FileName[50];

重要局部变量用途描述：

FILE *fp：用于打开文件
linkedlistADT nodeptr：链表节点，用于记录当前正在读的节点

char FileName[50]：要保存的文件名

函数算法描述：将当前的 list 链表全部清空。读取文件开头的各类图形的个数，确定之后每种图形读取的循环次数。在每次循环中，按先前的格式读取 txt 中保存的信息，赋值给结构体，并将结构体插入至链表。

3) 函数原型：saveDiagram(char* fileName)
功能描述：将绘制好的流程图截图并保存为 BMP 文件
参数描述：无
返回值描述：无
重要局部变量用途描述：无
函数算法描述：调用 WinAPI 函数库中的函数进行 Windows 编程

3.5.9 UI.c

1) 函数原型：void drawPenColorPanel()
功能描述：绘制画笔颜色选色板
参数描述：无
返回值描述：无
重要局部变量用途描述：
string color：记录当前画笔颜色；
char *colorName[]：存储颜色名称；
double ColorPieceW：选色板色块宽度；
double ColorPieceH：选色板色块高度；
int i, j, cnt：计数器；
函数算法描述：按照 char *colorName[]中颜色顺序分别绘制色块按钮，选择后会更改 currentPenColor。

2) 函数原型：void drawFillColorPanel()
功能描述：绘制填充颜色选色板
参数描述：无
返回值描述：无
重要局部变量用途描述：
string color：记录当前画笔颜色；
char *colorName[]：存储颜色名称；
double ColorPieceW：选色板色块宽度；
double ColorPieceH：选色板色块高度；
int i, j, cnt：计数器；
函数算法描述：按照 char *colorName[]中颜色顺序分别绘制色块按钮，选择后会更改 currentFillColor。

3) 函数原型：void customPenColor()
功能描述：自定义画笔颜色
参数描述：无
返回值描述：无
重要局部变量用途描述：
static char R_input[10], G_input[10], B_input[10]：记录输入的 RGB 数值的对应字符；

int RGB[3]：记录 RGB 数值；

函数算法描述：利用 textbox 获得用户输入的 RGB 数值，定义为颜色"customPenColor"并赋给 currentPenColor。

4) 函数原型：void customFillColor()

功能描述：自定义填充颜色

参数描述：无

返回值描述：无

重要局部变量用途描述：

static char R_input[10], G_input[10], B_input[10]：记录输入的 RGB 数值的对应字符；

int RGB[3]：记录 RGB 数值；

函数算法描述：利用 textbox 获得用户输入的 RGB 数值，定义为颜色"customFillColor"并赋给 currentFillColor。

5) 函数原型：void customPenSize()

功能描述：自定义线宽

参数描述：无

返回值描述：无

重要局部变量用途描述：

static char inputPenSize[10]：记录输入的线宽数值对应的字符；

int penSize：记录当前线宽；

string color：记录当前颜色

函数算法描述： 利用 textbox 获得用户输入的线宽数值，赋给 currentPenSize。

6) 函数原型：void drawButtons()

功能描述：绘制图标工具栏各个按钮

参数描述：无

返回值描述：无

重要局部变量用途描述：

double fH：当前字体高度；

double h：控件高度；

double w：控件宽度；

char fileName[30]：导出 BMP 文件的文件名；

int isSolid：判断当前线型是否为实线；

函数算法描述：按下控件时修改相应全局变量的值，调用相应函数完成操作。

7) 函数原型：void drawMenu()

功能描述：绘制菜单系统和状态栏

参数描述：无

返回值描述：无

重要局部变量用途描述：

double fH：当前字体高度；

double h：控件高度；

double w：控件宽度；

double xindent：缩进长度；

int selection：记录选择了哪一菜单项；

string color：记录当前画笔颜色；
int penSize：记录当前画笔大小；
函数算法描述：选择菜单项修改相应全局变量的值，调用相应函数完成操作。

8) 函数原型：void drawCanvas()
功能描述：绘制画布
参数描述：无
返回值描述：无
重要局部变量用途描述：无
函数算法描述：绘制特定参数的白色矩形形成画布区域。

4 部署运行和使用说明

4.1 编译安装

在桌面右键->显示设置中确保分辨率为 1920 * 1080，更改文本、应用等大小为 125%。

安装 Dev-C++ V5.10,文件打开 DiagramDrawer-devProjects 文件夹，选择 DiagramDrawer.dev，经编译、运行后产生 DiagramDrawer.exe，首次编译运行可能会提示有很多有关库函数中 GenUIID 的 warning，可以忽略，不影响程序的运行，可再次编译运行。

4.2 运行测试

例 1：运用断点和变量查看进行测试：LoadFile 函数读取文件的测试

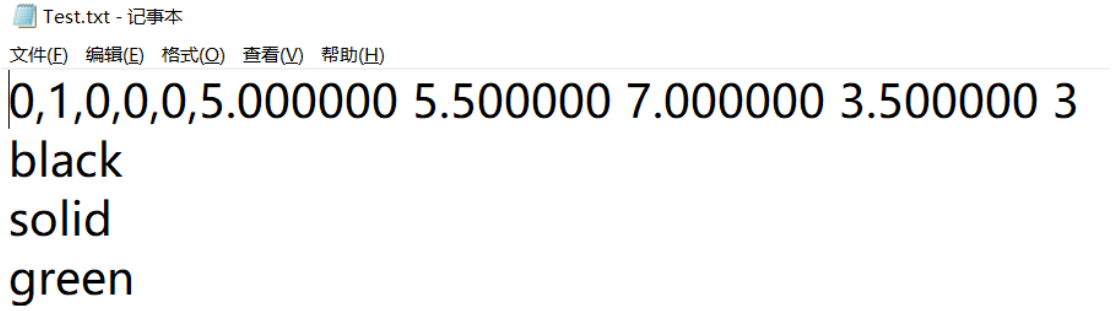
第一版 LoadFile 函数第一次写完时直接运行程序进行测试，发现程序未响应或直接闪退。首先分析出现这种问题可能出现的原因，一般有以下几种：死循环、内存越界、非法内存访问等。明确可能的错误原因后，还需大致确定错误发生的位置。涉及这次 bug 可能出现问题的操作只有两个，一是保存文件，二是读取文件。于是脱离鼠标相应函数，手动通过代码为一个结构体赋值，由于数据不存在特殊情况，以简单为好。插入链表后保存运用 SaveFile 保存至 txt 文件。

```
int curPenSize = 3; //当前画笔粗细
string curPenColor = "black";
string curFillColor = "green";
string curPenType = "solid";
```

```
double x = GetWindowWidth()/2, y = GetWindowHeight()/2;
```

```
curRect = (myRect)GetBlock(sizeof(*curRect));
          curRect->isSelected = FALSE;
          curRect->penColor = curPenColor;
          curRect->penSize = curPenSize;
          curRect->penType = curPenType;
          curRect->fillColor = curFillColor;
          InsertNode(list[RECT], NULL, curRect);
          curRect->x1 = MIN(x, x+2);
          curRect->y1 = MAX(y, y+2);
          curRect->x2 = MAX(x, x+2);
          curRect->y2 = MIN(y, y+2);
```

打开 txt 文件，发现生成的 txt 文件内的数据与手动赋值的数据相同，且格式符合 SaveFile 函数当中的输出格式。



于是基本可以确定错误实在 LoadFile 当中发生的。将断点设置于 LoadFile 函数，开始调试。

在单步执行的同时，保持查看变量。发现执行至 fscanf 函数调用时，发生段错误。在 fscanf 发生的段错误多为内存越界而发生。

仔细分析之后发现问题所在，我们先前将结构体内存储颜色、线型等变量定义为 string 类型，而在 fscanf 时直接将字符读取至这些变量当中。由于 string 类型是字符指针，它虽然可以指向字符串或表示字符串，但本身分配的到的内存空间只有一个字符指针的大小，不能用于存放字符串，需要为其动态分配内存空间。

```
Rect->penColor = (string)malloc(10 * sizeof(char));
Rect->penType = (string)malloc(10 * sizeof(char));
Rect->fillColor = (string)malloc(10 * sizeof(char));
```

动态分配内存空间后，段错误解决，但程序依旧秒退。继续进行断点调试，发现虽然读取正常执行，但通过变量查看观察到的读取到的字符串内容有问题。

```
Rect->penColor = (string) 0x5778dc0 "\r\360\255\272\r\360\255\272\020\034", "\2
```

此时基本可以明确是由于 fprintf 和 fscanf 的输出读入格式设置不太合理。由于单个结构体有多个字符串需要输出和读取，在使用%s 读取时若设置不合理，很容易读取失败，经过多次尝试后，我们中语更正了输出和读取的格式，完成了该项功能的调试。

例 2：特殊情况的测试：竖直箭头的选中

在设定好各类图形的选中范围后，尝试使用一些特殊的情况对图形选中进行测试。对于箭头，在尝试了四个象限方向的常规测试后，开始考虑完全水平和完全竖直的情况下的选中。发现水平情况下箭头线段可以正常选中，而竖直情况下的很难选中，有时可以选中有时不能。

于是开始进一步的测试，探究在怎样的情况下可以选中，怎样的情况下出现不能选中的 bug，以明确问题问题产生的原因和位置。在尝试中发现，只有当鼠标完全准确地刚刚好贴在线段上时才能够选中，选中范围只有竖直方向的一条线而不是一块区域，相当难以选中。

有了这样的情况，基本可以明确在选中函数中对于鼠标是否在选中区域内的判断可能存在逻辑上的错误。仔细查看代码后，发现选中区域的设置逻辑出现问题，在斜率不存在情况下，选中区域的 Δx 将等于 0。为了解决这个问题，加入一条 if 逻辑分支，专门判断斜率不存在的情况。

```
if (dx == 0)
{
    if (isBetween(dataptr->y1, dataptr->y2, y))
    {
        dataptr->isSelected = TRUE;  //将图形设置为已选中
        haveSelected = TRUE;
    }
}
```

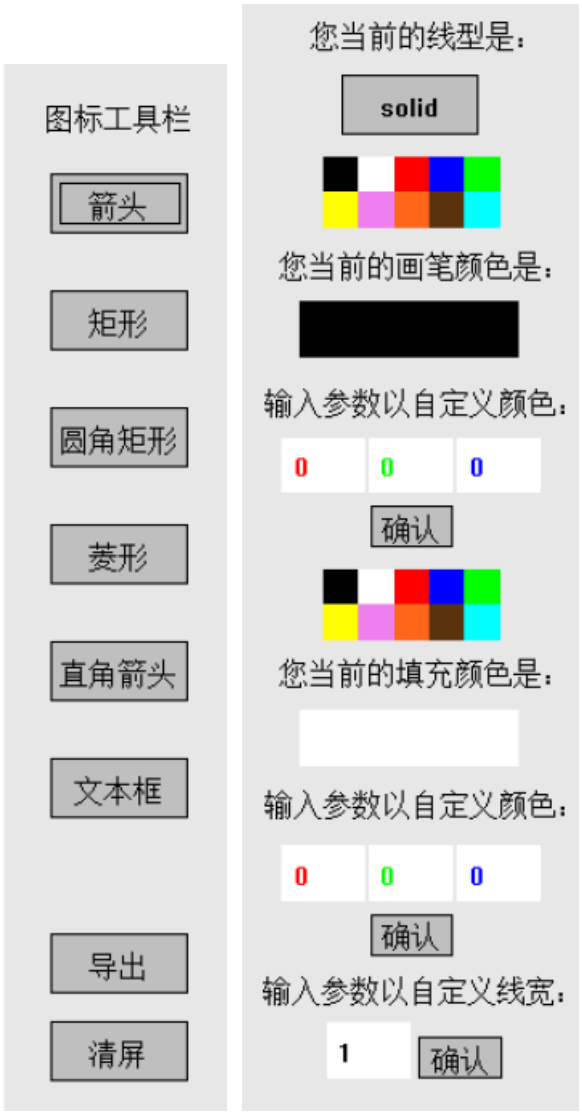
修改过后，问题解决。

4.3 使用操作

程序整体界面如下

打开软件后，首先点左上角文件菜单中的新建，出现白色画布，可以开始绘制。

点击左侧的工具栏可以选择不同的画图工具，右侧部分可以选择线型、粗细、颜色、填充颜色等。导出键可以将已完成的流程图导出成图片格式。

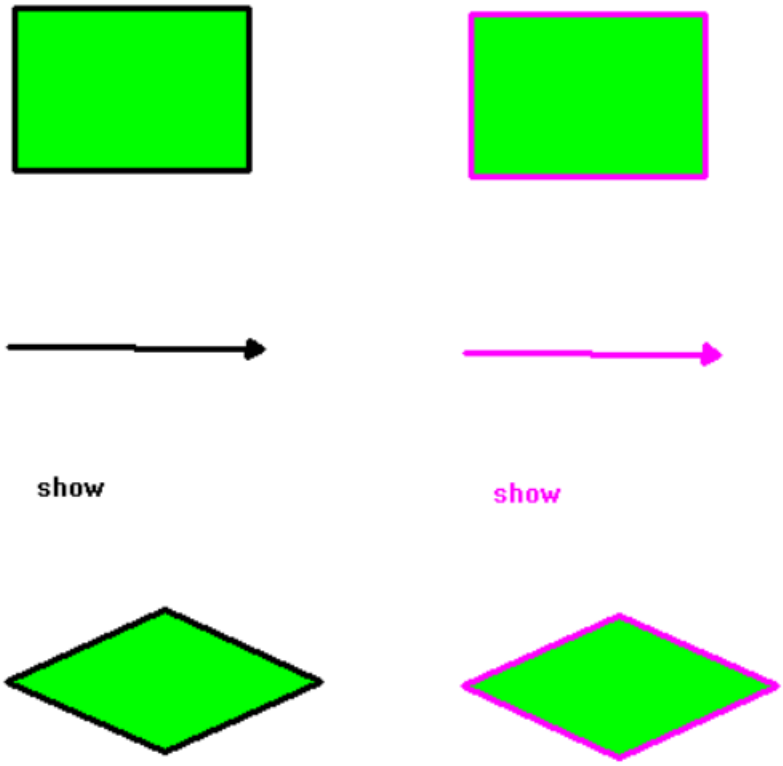


选择绘图工具后即可开始按住鼠标左键移动鼠标进行绘制。绘制直角箭头时，在保持按住鼠标左键绘制过程中按下鼠标右键可以变换直角箭头的顺逆时针方向。绘制完成后也可以再按鼠标右键变换刚画好的这个直角箭头的顺逆时针。

文本框不需要绘制范围，直接用鼠标左键在想要输入的位置上点一下，即可开始输入（没有输入内容时不会有光标显示），输入完成后需按回车完成输入。

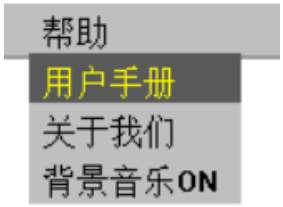
想要编辑已经绘制好的图形，需要先将图形选中，在编辑菜单栏中选择选中，即可通过鼠标左键单击选中图形，可以同时选中多个图形，图形被选中后边框会变为品红色。

按下 backspace 键可以删除全部已经选中的图形。按 enter 键取消选中。注意：由于原理原因，若鼠标所在位置上有两个图形重叠，不同种的的图形按箭头、直角箭头、矩形、圆角矩形、菱形、文本的顺序进行选中，同种类型的图形会先选中先画的。重叠情况下选中一个之后无法选中另一个，需要将已经选中的移开再选中另一个进行编辑，或者找到两者不重叠的位置进行点击选中。
下图中左侧为未选中状态，右侧为选中状态。



图形选中想要进行编辑，可以在编辑菜单中选择复制或者粘贴所有已经选中的图形，粘贴所得图形默认为选中状态。若想要拖动，再选择编辑菜单中的“改变”选项，此状态下按住鼠标左键可以进行拖移，按住鼠标右键可以进行缩放（只有已经选中的图形可以拖移和缩放）。可以同时缩放和拖移所有当前已选中的图形。选中文本框后，可以通过滚轮调整字体大小。

在帮助菜单中可以选择播放背景音乐，查看用户手册等。



5 团队合作

5.1 任务分工

程序员甲：

主要负责文件：main.c，figure.h，Rect.c，RndRect.c，String.c，UI.h，UI.c

代码量：约 1800 行

担任组长，统筹规划整组的进度，前期对产品进行需求分析，确定主要功能和实现方式，中期负责完成主函数的编写，实现矩形、圆角矩形和字符串的绘制和编辑操作，给产品增加了导出 BMP 文件功能，后期测试产品，寻找漏洞并解决，汇总最终结题报告并排版。与程序员乙共同完成鼠标消息回调函数以及共同实现 UI 画面刷新和画布画面刷新的分离。

难点 1：绘制圆角矩形需要用到画弧函数，而画弧函数与区域填充函数的匹配度不好。解决方法：将圆角矩形的绘制分解成分解成四步，先绘制出直线框并填充，再绘制圆角并填充，擦掉现有边框保留区域填充颜色，最后更改颜色重新绘制边框。

难点 2：课程提供的库无法实现导出 BMP 文件功能。解决方法：利用网络资源学习 WinAPI 函数库中的相关函数，了解相关函数的接口和用法以便调用进行 Windows 编程。

难点 3：图形用户界面需要提供优良的用户体验。解决方法：咨询设计专业方面的同学寻求配色、排版的建议，完成用户界面的美工。

程序员乙：

主要负责文件：Arrow.c，Edit.c，Edit.h，File.c，File.h

代码量（最终成品）：约 1500 行，大小约 41KB

负责完成的绘图功能有：箭头线段和直角箭头的绘制，为小组实现提供画虚线功能。负责完成编辑功能：选中区域设计划定，选中功能实现，复制和粘贴操作的实现，删除功能的实现。负责完成文件相关操作：实现将工程保存至 txt 文件的功能，实现读取 txt 文件保存内容并在画布上显示的功能。程序调试测试。与程序员甲共同完成鼠标消息回调函数以及共同实现 UI 画面刷新和画布画面刷新的分离。

难点 1：不同图形的选中区域判定。解决方法：画好平面几何示意图，根据示意图逐步判断。

难点 2：画图时不断刷新 UI 会卡顿，但是没有部分区域清屏的函数来实现刷新分离。解决方法：更换思路，如果不能清屏，可以通过在画布范围内画一块白色的矩形将先前所有的内容盖掉，再在此基础上显示，即可实现画布内部屏幕“刷新”，实现 UI 刷新与画布刷新分离。

程序员丙：

Diamond.c 文件以及背景音乐添加

5.2 开发计划

DiagramDrawer产品开发时间表

	事项
2020年5月1日	确定选题
2020年5月1日~6日	产品需求分析，确定主要功能
2020年5月7日	完成figure.h文件，统一宏定义、结构体定义以及函数接口
2020年5月8日~15日	编写图形的绘制函数，实现各类图形的绘制功能
2020年5月16日~25日	编写图形的编辑函数，实现各类图形的编辑功能
2020年5月26日~31日	完成图形用户界面的设计和功能实现
2020年6月1日~2日	添加文本框输入和编辑功能
2020年6月3日~4日	添加文件保存和读取功能；添加背景音乐功能
2020年6月5日	更改屏幕刷新原理，单独刷新画布区域以保证执行速度
2020年6月6日	基本功能全部实现，开始测试程序寻找漏洞并解决
2020年6月7日	添加导出BMP文件功能；编写用户手册
2020年6月8日~9日	撰写报告，复盘，总结产品开发经验，思考不足之处

5.3 编码规范

学在浙大课件—代码规范示例—C 编码规范.pdf

5.4 合作总结

本次合作开发的总代码规模在 100KB 左右，开发过程碰到过一些困难，但小组成员勇于攻克难题，积极讨论交流，在共同努力下成功解决了一系列问题，开发过程较为顺利。

本次开发运用到了以下知识点：

1. 模块化程序设计，具体包括递归函数，预处理命令，全局变量 extern 引用，static 全局变量和多文件组织等。
2. 高级指针，具体包括二级指针的概念和变量定义，指针数组，指针数组和二级指针的关系以及函数指针等。
3. 链表，具体包括动态内存分配，链表的定义、创建和基本操作等。
4. 图形程序设计，具体包括第三方图形库基本图形函数，编程模型以及回调函数等。

本次合作开发中有着一下开发上的亮点：

1. UI 界面与画布刷新分离。本小组最早采用全窗口实时刷新的方式进行屏幕刷新，然而在之后的调试当中，发现在画图过程中出现明显的卡顿，图形跟不住鼠标的移动速度，严重影响使用体验。研究后发现是由于 libgraphics 库的画图函数效率较低，实时刷新屏幕需要大量重画 UI，效率较低，但 UI 界面若不实时刷新便无法使用菜单和按钮。本小组通过分析，通过一定方法判断用户想要操作 UI 还是进行绘制，在不同情况下对进行 UI 界面与画布的刷新分离，有效减少了 UI 的刷新次数，显著提高了程序的效率，解决了卡顿问题。
2. 导出 bmp 图片功能。小组同学为完善功能，使软件更加便捷强大，查询资料学习课外知识，利用 WinAPI 函数进行 Windows 编程，实现了通过截屏方式将绘制好的流程图保存成 bmp 图片并导出的功能。
3. 自由的文件保存与读取。贴合人们日常的软件使用习惯，实现了将流程

图信息暂时保存至 txt 文档，以供用户随时进行修改，有利于流程图的分步骤绘制和合作绘制。

4. 背景音乐。

在开发的过程中，小组同学积极交流，互帮互助解决问题，制定统一标准规则，提高了合作的效率和有效性。一下是一些交流记录：

2020 年 5 月 10 日：

2020 年 5 月 17 日（最后砍掉了这项功能）：

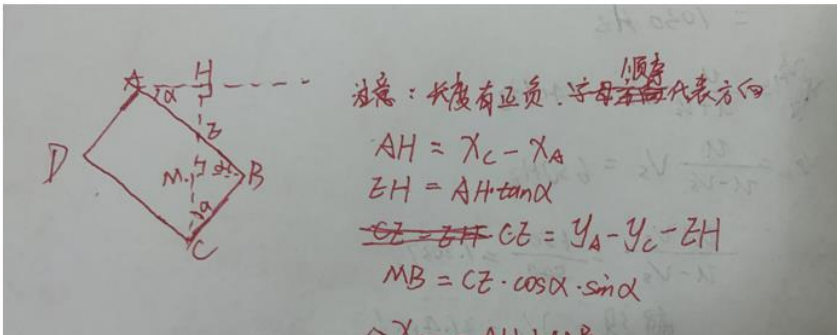
5 月 17 日晚↵

今天开始尝试写绘制直角箭头的函数。基本作图原理与矩形类似，画出半边矩形再画上一个箭头即可，方向有顺逆时针两种。↵

难点在于，该函数要能够画出旋转过一定角度的图形，存储图形的结构体当中记录的是旋转后当前图形的左上角坐标、左下角坐标以及将这个图形补全所得的矩形与 x 轴正方向的夹角 angle。↵

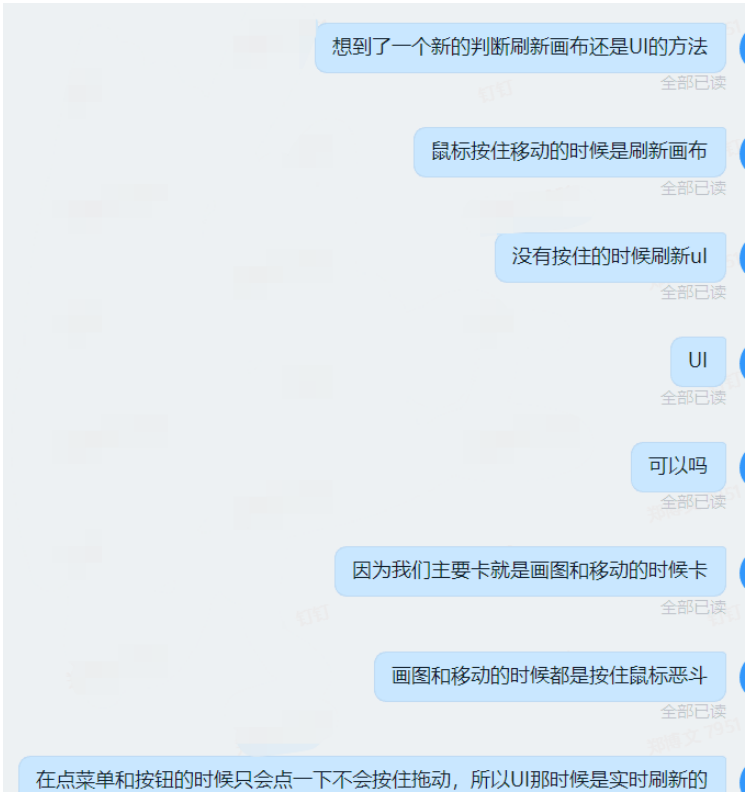
于是这就变成了一道数学题，已知矩形两对顶点坐标以及矩形的一条边和 x 轴正方向的夹角，作出这个矩形。过程比较复杂，为了防止以后遗忘这个方法，难以处理之后可能出现的 bug，故记录这种方法，以顺时针方向为例，逆时针方法类似，方法如下图。注意，所有长度都是有正负的。↵

α 为顺时针转过的角度，经检验，当-90° <= α <= 0° 时也成立，猜测所有情况成立。↵

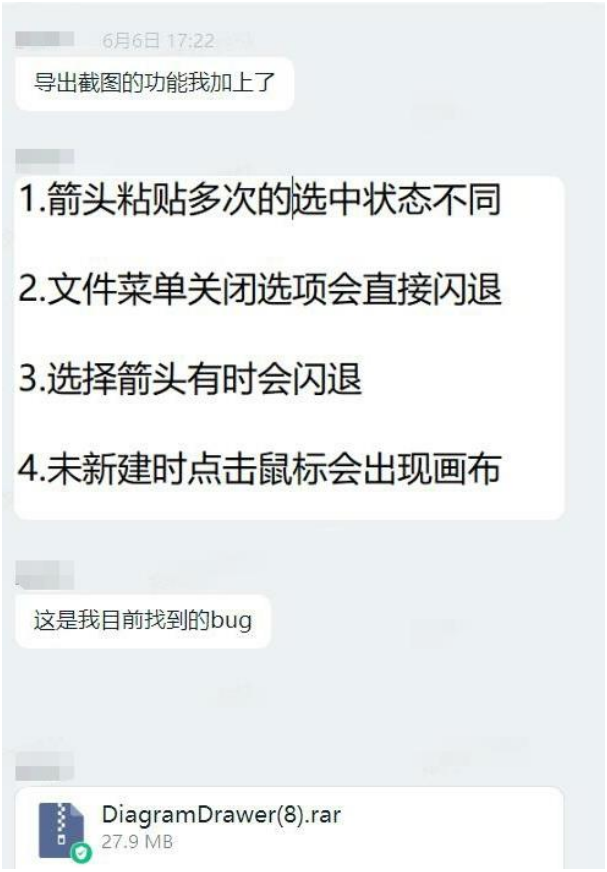


2020 年 6 月 5 日


音乐应该是循环播放的
关了再开是从头放
话说你们那里应该能用的吧
能
我改了鼠标事件那块，现在出现了画到外面擦不掉的情况
我把判断加上吧，不让图形出画布
而且也有划的快有残留的问题


想到了一个新的判断刷新画布还是UI的方法
全部已读
鼠标按住移动的时候是刷新画布
全部已读
没有按住的时候刷新ui
全部已读
UI
全部已读
可以吗
全部已读
因为我们主要卡就是画图和移动的时候卡
全部已读
画图和移动的时候都是按住鼠标恶斗
全部已读
在点菜单和按钮的时候只会点一下不会按住拖动，所以UI那时候是实时刷新的

2020 年 6 月 6 日


6月6日 17:22
导出截图的功能我加上了
1.箭头粘贴多次的选中状态不同
2.文件菜单关闭选项会直接闪退
3.选择箭头有时会闪退
4.未新建时点击鼠标会出现画布
这是我目前找到的bug
DiagramDrawer(8).rar
27.9 MB

5.5 收获感言

程序员甲：

本次开发过程中，我被组员们推选为组长，在完成代码任务的同时认真履行组长职责，统筹规划全组进度，定期组织会议讨论，分配任务。本次开发任务让我明白，一个团队的成功与他的每一个成员都是分不开的。我们在为期一个月左右的开发过程中，互相鼓励，共同进步，更是结下了良好的友谊，增进了默契。

此外，在本次开发任务中，我们将课堂上学习到的知识运用到了实际，不仅加深了各个知识点的理解，还延申到了很多课堂上未曾讨论的新领域，为了解决一个需求学习了很多新知识，自学能力和实践能力都得到了极大锻炼。

总的来说，这是我们第一次以团队的形式完成一个产品的开发，也是第一次让理论落地的尝试，在这个过程中我们收获了很多，希望自己以后能将这份对团队协作的热情一直保持下去。

程序员乙：

本次合作开发，我收益颇丰，充分认识到了团队合作的重要性，也理解了合理分工的重要性。在这次的程序开发中，我认识到了在开发过程中，软件的架构框架设计是非常重要的。如果没有好的框架设计，那就是在建造一栋地基不稳、岌岌可危的高楼，一些错误的产生可能会因为不合理的框架而需要极大规模的改动，而好的框架设计能够有效规避一些错误带来的后果。

其次，我也认识到了在程序开发的过程中，团队内部有效的交流是极其重要的。由于代码开发工作是非常抽象化的，如果没有良好的沟通技巧，就无法与团队成员交接工作、交流问题，在调试和改错阶段也会碰到非常大的阻力，从而影响整个团队的开发效率。为了实现高效的交流，所有人都应该熟练掌握各种术语，并且在团队内部定下一定的规则和标准。

我自以为在本次合作开发过程中有着较为良好的表现，积极与同学交流，及时完成自己的任务，对于自己的代码认真负责，对所有的错误负责到底，不给团队里的其他成员带来困扰，也热心帮助同学共同寻找问题所在，共同解决问题。然而，在开发最后阶段我也发现了在先前开发的代码当中存在的一些过于繁复的问题，若时间充足，还可以继续进行改良简化。在今后的学习过程中，也应该多多注意这样的问题。

程序员丙：

在完成这次大作业的实践过程中，我通过查找资料以及组员间的交流学

到了很多，对于一学期下来学习的知识也有了更多理解和运用经验。反思下来，自己在合作过程中有许多做的不足之处，比如在尝试实现更多内容时主动性不足；在过程中遇到困难且没有头绪时，缺少了与组内或者其他同学的交流，浪费了很多时间，最后也因为时间与能力的限制而放弃了一些想法等。很抱歉对于组内没有提供很多的帮助，也非常感谢小组以及课程中老师同学带来的帮助。

6 参考文献资料

https://blog.csdn.net/foreverhuylee/article/details/22753315?ops_request_misc=&request_id=&biz_id=102&utm_term=WINAPI%20StretchBlt&utm_medium=distribute.pc_search_result.none-task-blog-2~all~sobaiduweb~default-2-22753315
（理解 Bitblt、StretchBlt 与 SetDIBitsToDevice、StretchDibits）
https://blog.csdn.net/jarodlau/article/details/339759?ops_request_misc=%257B%2522request%255Fid%2522%253A%252215914319051972484831288l%2522%252C%2522scm%2522%253A%252220140713.130102334.pc%255Fall.%2522%257D&request_id=159143190519724848312881&biz_id=0&utm_medium=distribute.pc_search_result.none-task-blog-2~all~first_rank_v2~rank_v25-8-339759.pc_search_back_js&utm_term=winapi%E4%BF%9D%E5%AD%98bmp%E5%9B%BE%E7%89%87
（捕捉屏幕并保存位图文件）
https://blog.csdn.net/Valerian0/article/details/23276231?utm_source=blogxgwz8?utm_medium=distribute.pc_relevant.none-task-blog-baidujs-4
（C 语言实现 BMP 图片生成）
https://blog.csdn.net/fengxianghui01/article/details/82937642?utm_medium=distribute.pc_relevant.none-task-blog-BlogCommendFromBaidu-4.nonecase&depth_1-utm_source=distribute.pc_relevant.none-task-blog-BlogCommendFromBaidu-4.nonecase
（C 语言实现 BMP 图像处理（读取与保存））
https://blog.csdn.net/qq_21237549/article/details/104264621
（DEV C++设置背景音乐）
https://blog.csdn.net/QQ1910084514/article/details/80754123
（PlaySound 函数用法）