浙江大学

程序设计专题

大程序报告

1. 学生姓名 ：杜祐陞　　学号：3180100186
2. 学生姓名 ：王晨宇　　学号：3180104919
3. 学生姓名 ：章可循　　学号：3180104920
4. 学生姓名 ：周屹赫　　学号：3180104922

指导老师：陈建海

2018~2019 春夏学期 2018 年 5 月 31 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

1. 题目描述和题目要求

设计和实现一款带有图形界面的游戏(数独,五子棋,9 宫格,小蜜蜂,拼图等)或其他应用。要求基于 libgraphics，设计和实现图形界面，至少运用以下技术和知识：菜单、按钮、鼠标、键盘、文件、链表。

选题名称：元素之战 War of Elements
特色：有三维血量值的怪物经过不同路径向玩家进攻，玩家需要建造功能各异的防御塔来消灭怪物，取得游戏胜利。

考评项		分数	说明
菜单系统		10	游戏开始界面的菜单，游戏进行中选择塔的菜单，暂停界面，选关卡界面的菜单。
图标工具栏		5	暂停游戏，选择对应的防御塔。
快捷键		5	游戏界面：ASDF 对应 1234 塔，空格暂停 暂停界面：ctrl+s 保存，（esc 或空格）回到游戏 胜利界面：回车或 esc 返回选择关卡界面 失败界面：回车重开 esc 退出
状态信息栏		5	窗口底部，即时显示玩家的 HP 和金钱
功能（35）（分项自行添加）	基本界面图形绘制	15	基本界面的生成，怪物、防御塔、地图、路径、血量、金钱、怪物 debuff 的绘制，检测按钮是否被点击
	业务功能实现	20	每个时间周期怪物移动、扣血、防御塔攻击、怪物变色、玩家造塔等内部逻辑的实现。
链表		5	设计链表用于系统数据的组织，用链表来完成怪物的遍历，删除，生成。
文件	文件读取	10	保存地图、路径到文本文件； 保存怪物、防御塔、HP、金钱到文本文件以存档；
	多文件组织	5	系统程序采用多文件组织的方式

大程序报告	分析和设计	5	软件简介，功能结构，全局、函数及重要算法说明，源程序中功能、函数、文件的组织关系
	部署和运行	5	编译安装、运行测试、用户使用手册
	分工	5	成员各自承担任务、挑战点和感言
	合作	5	合作记录、系统开发亮点和应用知识点总结
总分	100		

2. 需求分析

2.1. 游戏总体

正式开始游戏前，支持进入游戏、选择关卡、加载存档、查看游戏说明、退出。

开始游戏后，支持暂停游戏、退出游戏、保存游戏。

2.2. 关于怪物

每个怪物有三维血量，当怪物的三维血量均被攻击至 0 时怪物死亡，怪物的三维血量被转化为 RGB 值绘制在屏幕上，玩家可以通过查看怪物的颜色来了解其掉血情况，当怪物完全变黑时它死亡。

怪物会按照一定的波次，一波一波地进攻玩家。

怪物会按照特定的路线进攻玩家，当怪物走到其路径的终点时，玩家将会受到一点伤害。如果怪物在走到终点前死亡，则玩家会获得 10 元钱。

2.3. 关于防御塔

按照西方奇幻中的四元素火、风、水、土设计，不同的防御塔有不同的攻击范围、攻击频率、攻击对象限定和攻击效果，给游戏增添乐趣和多样性。建造防御塔需要消耗金钱。

2.3.1. Fire

单体攻击，被攻击到的怪物获得灼伤状态，在灼伤状态下的怪物会持续扣血，灼伤状态会持续若干回合，若在灼伤尚未消失前再次被 Fire 攻击，则灼伤时间又被重置。

2.3.2. Air

单体攻击，攻击频率较高，无特殊效果。

2.3.3. Water

攻击范围内的所有怪物，怪物不直接受到伤害，而是获得减速状态，在减速状态下的怪物速度会降低，减速状态会持续若干回合，若在减速尚未消失前再次被 Water 攻击，则减速时间又被重置。

2.3.4. Earth

攻击范围内的所有怪物，造成一定量的伤害，无特殊效果。

2.4. 关于玩家

玩家主要有 HP 和金钱两个属性。HP 会因为怪物走到终点而减少，金钱会因为消灭怪物而增加。

当 HP 减少为 0 时，玩家会输掉游戏。

当某个关卡的所有怪物进攻结束，且玩家 HP 仍为正数时，玩家获得胜利。

3. 总体设计

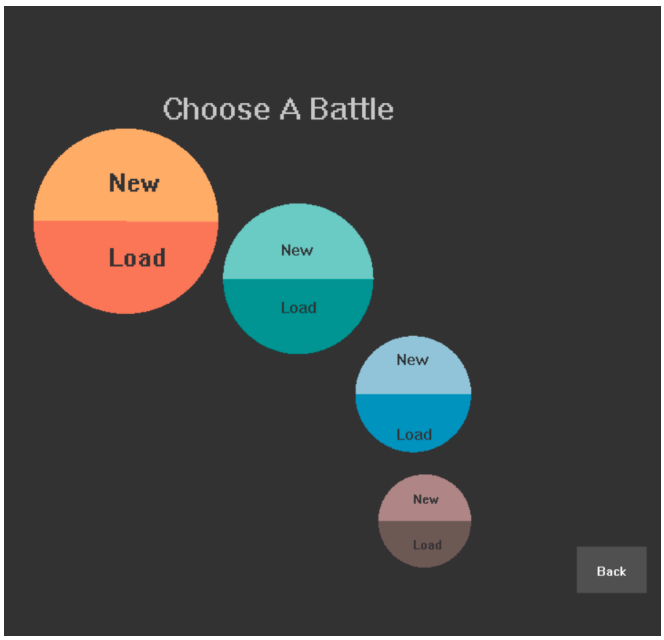
3.1. 功能模块设计

3.1.1. 用户界面

3.1.1.1. 开始界面

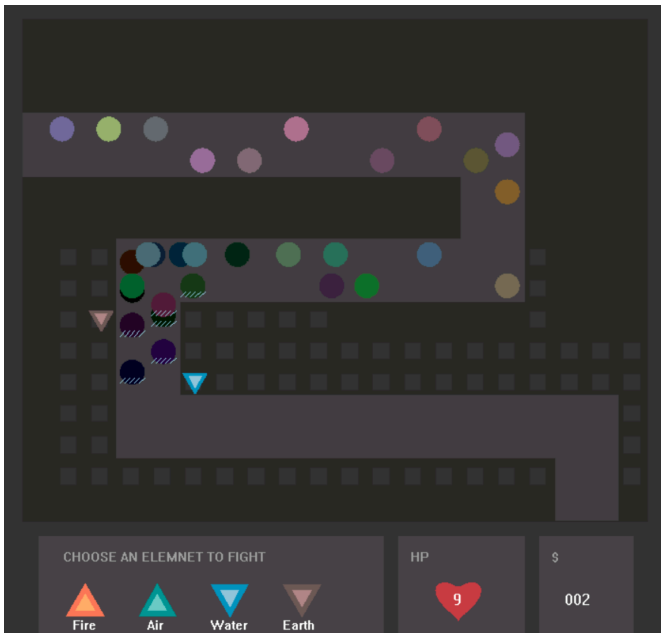
有开始游戏，游戏介绍，退出游戏三个按钮。

3.1.1.2. 选择关卡



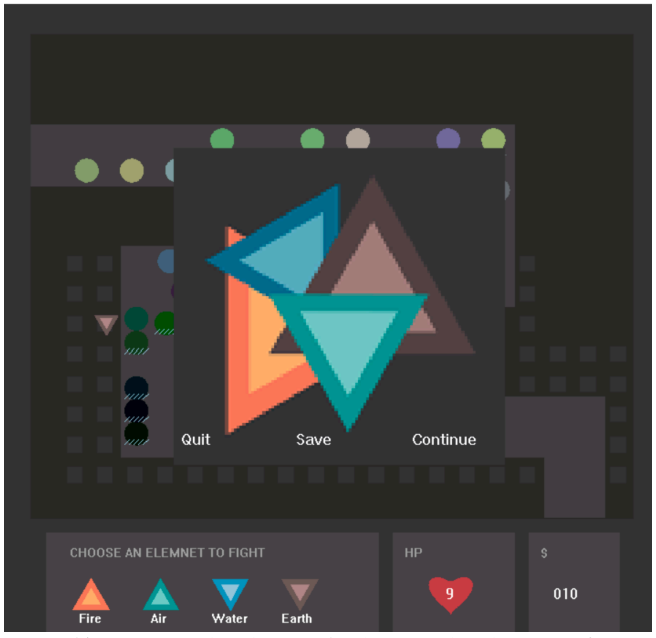
目前有四个关卡供玩家选择，点击 New 可以重新挑战该关卡，点击 Load 可以继续上次对该关卡的挑战。

3.1.1.3. 游戏界面



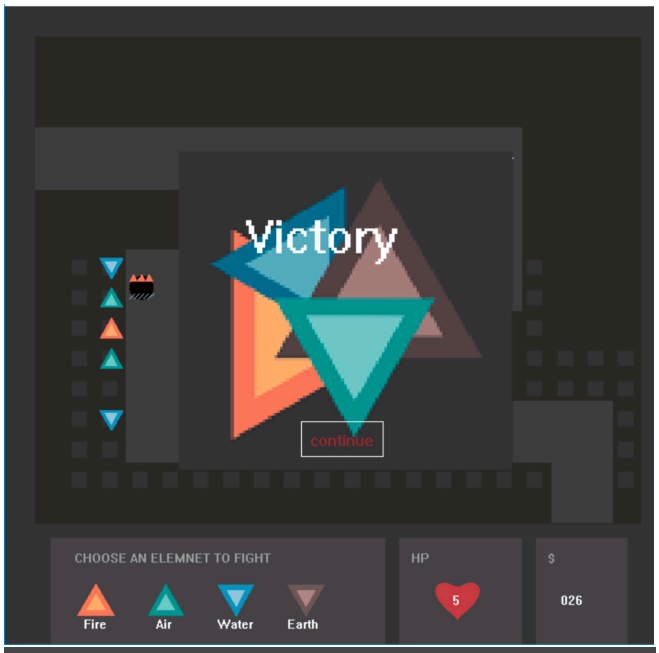
游戏界面的主要部分为地图，上面显示了怪物和塔，左下方的工具栏供玩家选择塔进行建造。右下方显示了玩家的 HP 和金钱。

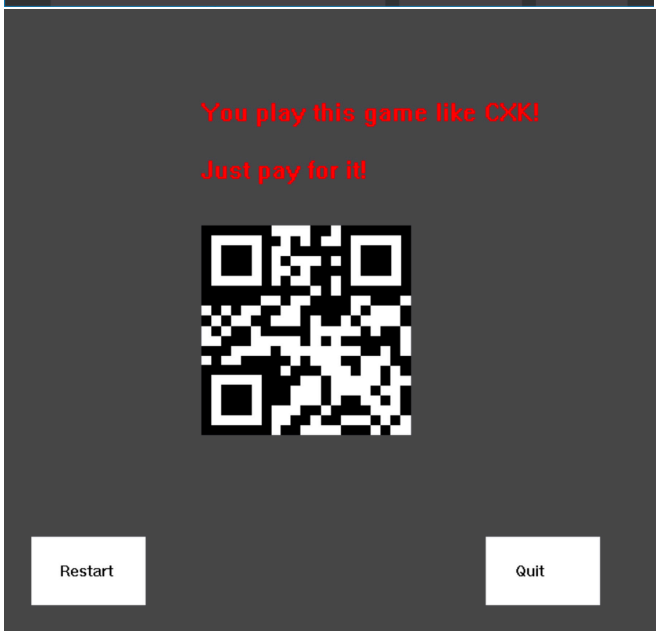
3.1.1.4. 暂停界面



按空格键可以暂停/开始游戏，暂停后玩家可以选择保存游戏，退出当前关卡，继续当前关卡。

3.1.1.5. 胜利和失败界面





3.1.2. 内部逻辑

3.1.2.1. 区分界面

设置全局变量 Status 来表明当前所在的是那个界面。

3.1.2.2. Timer

一个 Timer 用于向玩家展示怪物的行进路线。

一个 Timer 用于进行游戏。游戏中最小的时间单元为该 Timer 的一个周期，将其称之为游戏的一个回合。

每个回合中会发生如下事件：结算怪物身上的状态，怪物行进，防御塔攻击，检测怪物是否到达终点，重新绘制地图将新的怪物状态绘制上去等。所有和游戏相关的事件的时长都是回合时长的整数倍。

3.1.2.3. 鼠标事件

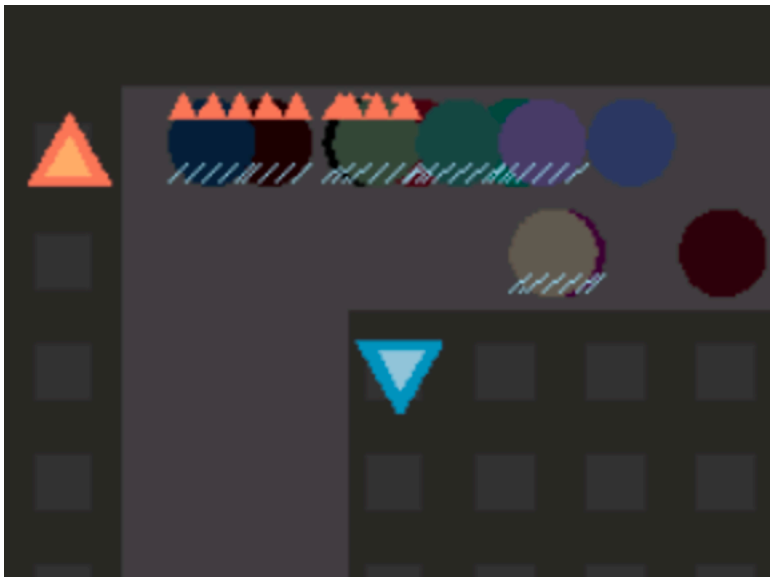
首先根据全局变量 Status 来判断当前位于哪个界面，然后根据鼠

标点击的位置和事件来判断玩家点击了哪个按钮，然后进行相应的操作。

3.1.2.4. 怪物

每个怪物都有其特定的前进路线，路线上标明了转弯的位置，怪物以一定速度前进（考虑减速状态的影响），当到达转弯位置时做出相应的转弯。

怪物可能有灼伤和减速两种效果，每一回合这个效果的剩余时间都会减少，每一回合灼伤效果都会对怪物造成一定的伤害。当怪物具有某种效果时，它身上将会出现该效果对应的标记。红色三角形代表灼伤，蓝色条纹代表减速。



怪物的血量被改变后要将其重新按照新的颜色绘制到地图上。

3.1.2.5. 塔

玩家选择想要使用的塔然后将其放置在地图上可放置的位置。
第一次点击某位置，将会用绿色圆圈标记该塔的攻击范围。第二次点击，则在该位置摆放塔。

塔有一定的冷却时间（两次攻击的间隔），冷却时间是回合的整数

倍，每个回合对每个塔检验其是否冷却完成，若完成则进行攻击。每个塔有其特定的选择攻击对象的规则。塔会对于其攻击范围内的每个目标进行估价，估价最高的将会被攻击。

3.1.3.存档与文件交互

3.1.3.1.地图文件

地图以文本文件的形式存储在相对目录 map 下，游戏运行时从对应的目录读取地图信息。地图信息主要包括：怪物所经过的道路对应的格点，可放置防御塔的格点，怪物行进的道路以及上面转弯的位置等。

3.1.3.2.存档文件

为每个关卡设置一个存档文件，玩家可以在中途暂停某个关卡进行保存，并在之后加载存档继续游戏。存档中存储的主要信息为：当前怪物的位置，剩余怪物的序列，怪物的血量，怪物的状态，塔的位置，玩家剩余的 HP 和金钱。

3.2.数据结构设计

3.2.1.MONSTER

```
struct MONSTER{
        double x, y; // 所处的格点坐标
        double v; // 移动速度
        double hpr, hpg, hpb; // 剩余的三维生命值
        int slowdown_cnt; // 减速效果剩余回合数，0 代表未被减速
        int burnt_cnt; // 灼伤效果剩余回合数，0 代表未被灼伤
        int road; // 该怪物的前进路线
        int swerve_cnt; // 该怪物进行了多少次转弯
        bool is_alive; // 该怪物是否存活
        struct MONSTER* nxt; // 指向下一个存活的怪物的指针
};
```

此处使用链表存储怪物，方便快速遍历以及在怪物死亡时快速将其删除。

3.2.2.TOWER

```
typedef struct{
        double x, y; // 塔的格点坐标
        int cool_down; // 塔的冷却时间，冷却到 0 时塔可以射击
        int type; // 塔的类型，1-4 分别为 Fire, Air, Water, Earth
}TOWER;
```

3.2.3.PATH

```
struct PATH{
    double STx, STy, Endx, Endy; // 道路的起点和终点
    double Crossroadx[10]; // 道路的转折点横坐标
    double Crossroady[10]; // 道路的转折点纵坐标
    int Direction[20]; // 在转折点处转弯后的方向
    int Crossroad_cnt; // 转折点的个数
```

};

3. 2. 4. 主要全局常数说明

Atkr[], Atkg[], Atkb[] – 各个类型的塔对于怪三维血量的攻击力
Period[] – 塔的冷却时间
Atk_range[] – 塔的攻击范围
P – 一个回合的时长
Monster_limit – 怪物个数上限

3.2.5.主要全局变量说明

Status - 代表当前于哪个界面。
Map – 代表哪些位置是供怪物行走的，哪些位置是可放置防御塔的。
Monster_cnt, tower_cnt, path_cnt – 怪物、防御塔、路径的个数。
Monster_head – 怪物链表的头指针（哨兵节点，其下一个节点为第一个存活的怪物）
Monster_tail – 怪物链表的尾指针
Tower[] – 存放塔的数组
Path[] – 存放道路的数组
Is_Pause – 游戏是否暂停
Sequencer[], sequenceg[], sequenceb[] – 怪物的血量序列

3.3. 函数功能描述

3.3.1.内部逻辑和文件交互

1. 怪兽结构体定义：

```
struct MONSTER{
  double x, y; // 所处的格点坐标
  double v; // 移动速度
  double hpr, hpg, hpb; // 剩余的维生命值
  int slowdown_cnt; // 减速效果剩余回合数，0 代表未被减速
  int burnt_cnt; // 灼伤效果剩余回合数，0 代表未被灼伤
  int road; // 该怪物的前进路线
  int swerve_cnt; // 该怪物进行了多少次转弯
  bool is_alive; // 该怪物是否存活
  struct MONSTER* nxt; // 指向下一个存活的怪物的指针
};
```

2. 塔结构体定义

```
typedef struct{
  double x, y; // 塔的格点坐标
  int cool_down; // 塔的冷却时间，当冷却到 0 时塔可以进行射击
  int type; // 塔的类型，1-4 分别为 Fire, Air, Water, Earth
}TOWER;
```

3. void Refresh();
重画整个游戏界面

4. bool Load_Information(int mapid);
接受关卡编号 mapid，从相应的目录读取存档信息。
若存档不存在，则返回 0。

5. bool Load_Information(int mapid);
接受关卡编号 mapid，将怪物、塔、HP、金钱等信息写入相应目录的存档。

6. void Load_Map();
从存储地图信息的文件中读取地图信息。包括地图中的每一条怪物行进的路线以及转弯的位置。
将地图的道路绘制到游戏界面上。

7. void Load_Path();
将怪物行进的路线转换为怪物转弯的位置。

8. void init(int Gameid, bool if_load)();
初始化游戏，Gameid 表示当前进行的是第几个关卡，if_load 表示当前是读取存档还是进行新游戏。

9. void Tower_Attack();
遍历所有塔，让每个塔进行攻击。

10. void Monster_Generator();
按照怪物出现波次的规律随机生成一个怪。

11. void Tower_Generator(double x, double y, int type);
在位置(x, y)生成一个类型为 type 的防御塔。

12. double velo(MONSTER m);
接受怪，返回它经过可能的减速之后的速度。若被减速，则速度变为原来的 slowdown_coef 倍。

13. bool At_EndPoint(MONSTER t);
接受怪，检查它是否到达了终点。

14. int Pass_Crossroad(MONSTER t);
接受怪，返回 0，代表怪物下一步路程中不经过转弯点；返回 1，代表怪物此时恰好处在转弯点上；返回 2，代表怪物的下一步路径种跨过转弯点

15. void Monster_Move();
让所有怪物进行一轮移动。

16. bool At_Crossroad(MONSTER t);
判断路径小块是否到达转弯点。

17. void Track_Move();
让路径小块沿着怪物行进路线画出地图路径。

18. void one_to_one_atk(TOWER *pt, MONSTER *pm);
让*pt 对应的塔攻击*pm 对应的怪。若怪去世，则将其删除。

19. void LosingHPCheck();
检查所有怪是否到达终点，若到达，则减少玩家的 HP 值。

20. double sqr(double x);
返回 x 的平方。

21. double dist_tm(TOWER t, MONSTER m);
返回怪和塔的距离。

22. double calc_weight(TOWER t, MONSTER m);
计算塔 t 攻击怪 m 的权重。对于单体攻击的塔而言，它会选择攻击权重最大的怪。
权重取决于该怪还有多久离开该塔的攻击范围（利用二分法求解方程求出时间），还取决于该怪剩余的血量。

23. void tower1_attack(TOWER *pt);
让 pt 指向的 Fire 塔攻击，它会先选择权重最大且没被灼伤的怪。
若不存在没被灼伤的怪，则它会攻击权重最大的怪。
被它攻击的怪的 burnt_cnt 会被赋值为 BURNT_PERIOD。

23. void tower2_attack(TOWER *pt);
让 pt 指向的 Air 塔攻击，它会攻击权重最大的怪。

24. void tower3_attack(TOWER *pt);
让 pt 指向的 Water 塔攻击，它会让范围内所有的怪获得减速效果。
即 slowdown_cnt 被赋值为 SLOWDOWN_PERIOD。

25. void tower4_attack(TOWER *pt)
让 pt 指向的 Earth 塔攻击，它会攻击范围内的所有怪。

26. void Monster_Burn();
计算怪物身上的状态对于怪的影响，灼伤会减少其血量。
灼伤和减速的回合数减一。

27. void TimerEventProcess(int timerID);
根据 TIMERID 来判断是路径生成进程还是游戏进程。
根据调用的时间间隔来生成怪物以及处理塔的攻击与怪物的移动。

28. void MouseEventProcess(int x, int y, int button, int event);
先根据 Status 来判断属于哪个界面。

然后根据鼠标点击的事件和位置判断点在哪个按钮上，然后调用相应的功能。

3.3.2.用户界面绘制

3.3.2.1.about

1.void draw_about()
画出 about 界面
2.int About_Back_Available(double x,double y)
返回鼠标按键是否按到 back 按钮 返回 1/0 表示 按到/没按到

3.3.2.2.battle

1.void draw_Battle()
画出 battle 界面
2.static void draw_circle1/2/3/4()
画出关卡 1/2/3/4
3.int choose_circle(double x,double y)
判断鼠标在哪个圆内
4.int choose_new_load(double x,double y,int fl)
判断在 fl 圆上方还是下方

3.3.2.3.begin

1.static void Tower1/2/3/4(double x,double y,double Scale)
画出开始界面的塔 1/2/3/4 x,y 为其坐标， Scale 为其大小
2.int Begin_Available(double x,double y)// start 1, about 2, exit 3
返回鼠标按键是否按到按钮 返回 1/2/3 表示 按到/start/about/exit
3.void draw_begin()
画出开始界面

3.3.2.4.fail

1.void fail_surface()
画出失败界面
2.static void QRcode_generator()
生成一个（假的）二维码
3.static Draw_Tri_Rec(double sx,double sy)
画二维码的三个定点正方形
4.int fail_available(double x,double y)
返回 鼠标是否按到按钮上 返回 0/1/2 表示没按到/按到 restart/按到 quit

3.3.2.5.imgui

1.void drawRectangle(double x, double y, double w, double h, int fillflag)
画一个左下角顶点为 x,y，宽 w, 高 h 的矩形， fillflag=0/1 表示填充/不填充颜色
2.void drawBox(double x, double y, double w, double h, int fillflag, char *label, char labelAlignment, char *labelColor)
画一个左下角顶点为 x,y，宽 w, 高 h 的矩形，并在其内部显示一个字符，fillflag=0/1 表示填充/不填充颜色， label 为内部的字符，labelAlignment 表示字符居左/居右/居中，labelColor 表示填充的颜色

3.void drawLabel(double x, double y, char *label)
显示字符串标签
4.drawTower1/2/3/4(double x,double y,double Scale)
在 x,y 处画防御塔 1/2/3/4， Scale 为该防御塔的尺寸大小
5.void drawmonster1(double x,double y,double hpr,double hpg,double hpb)
在 x,y 处画怪物，hpr,hpg,hpb 为其三维血量值
6.void drawheart(double x,double y)
在 x,y 处以点阵图方式画一颗心
7.void drawpath(int a[][22])
画该关卡的游戏地图
8.void drawmenu()
画出下方菜单选项栏
9.void drawshadow()
画出地图上的阴影（表示此处可以放塔）
10.void Repaint(MONSTER *monster,int number_M,TOWER *tower,int number_T,int HP,int MONEY)
刷新游戏界面 monster 怪物序列，number_M 怪物个数 TOWER *tower 防御塔序列 number_防御塔个数
HP 玩家当前血量，MONEY 玩家当前金币数
11.int Calc_Idx(double x)
返回坐标 x 对应在那个网格上
12.int Calc_Idy(double y)
返回坐标 y 对应在那个网格上
13.int Available(double x,double y)
返回鼠标点击处是否能够造防御塔 1/0 能/不能
14.int Choose_Tower(double x,double y)
返回鼠标点击选择了菜单栏的哪个防御塔 0/1/2/3/4 没选择/选塔 1/选塔 2/选塔 3/选塔 4
15.void Out_Line(int fl)
若 fl=1/2/3/4 在菜单栏的防御塔 1/2/3/4 处画一个方框 表示已选择此防御塔
16.void Re_Brush(int a[][22])
重刷一遍游戏道路
17.void Draw_Tower_Scope(double x,double y)
标出位于 x,y 处的塔的攻击范围

3.3.2.6.pause

1.void draw_save_successfully()
画出 保存成功 字样
2.int Pause_available(double x,double y)
返回鼠标按键 x,y 处是否按到按钮,返回 1/2/3 表示按到 quit/continue/save
3.void draw_pause()
画出暂停界面

3.3.2.7.shoot

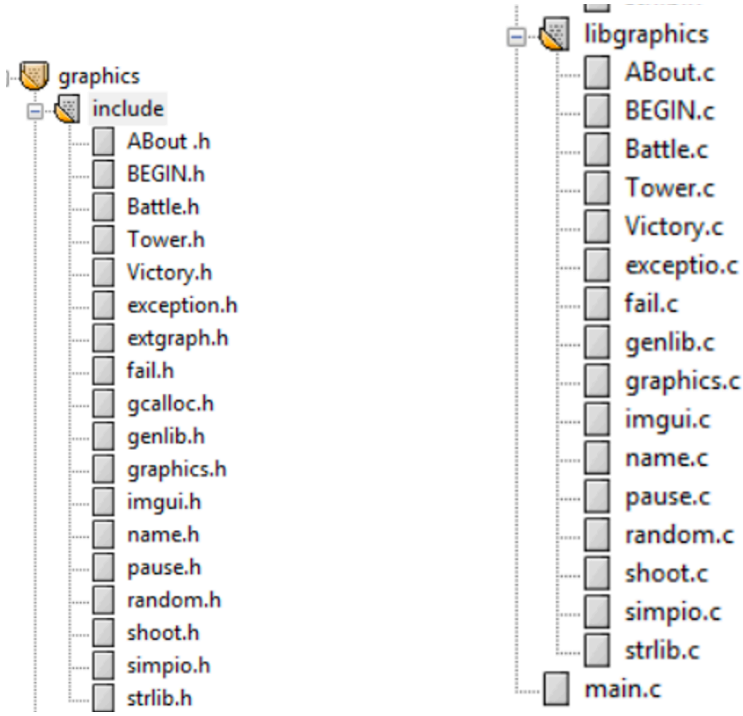
1.void Air_Shoot(double sx,double sy,double tx,double ty)

画出风属性防御塔（sx,sy) 对怪物（tx,ty) 的攻击
2.void Earth_shoot(double sx,double sy)
画出地属性防御塔（sx，sy) 的攻击（范围攻击）
3.void Water_Shoot(double sx,double sy)
画出地属性防御塔（sx，sy) 的攻击（范围攻击）
4.void Fire_Shoot(double sx,double sy,double tx,double ty)
画出火属性防御塔（sx,sy) 对怪物（tx,ty) 的攻击
5.static void Draw_Tri(double tx,double ty)
画出一簇火苗
6.void Draw_firing(double tx,double ty)
画出灼烧效果
7.void Draw_frozing(double tx,double ty)
画出冰冻减速效果

3.3.2.8.victory

1.void draw_victory()
画出本关胜利界面
2.int victory_continue_available(double x,double y)
返回用户是否点击 conitnue 按钮 1/0 表示点击/为点击

3. 4. 程序文件结构



3. 4. 1. 分文件说明

main.c – 主程序
imgui.c – 绘制地图、塔、怪物、血量、攻击范围等基本单位的函数
name.c – 数据结构的定义
shoot.c – 绘制塔的射击效果
fail.c – 绘制失败界面，检测点击事件。
About.c – 绘制游戏说明界面，检测点击事件。
Begin.c – 绘制开始界面，检测点击事件。

Battle.c – 绘制关卡选择界面，检测点击事件。
Victory.c – 绘制胜利界面，检测点击事件。
Pause.c – 绘制暂停界面，检测点击事件。

4. 部署与运行

4.1. 编译说明

使用 DevC++打开根目录下的项目文件 graphics.dev，点击全部重新编译按钮或按 F12 键进行编译，点击运行或按 F10 运行。
会被编译成 graphics.exe 的可执行文件。

4.2. 游戏手册

本游戏上手简单，若有疑问请参考根目录下的游戏手册.docx。

5. 组内分工

5.1. 分工情况

杜祐陞：关卡设计，制作游戏手册，游戏测试，聊天记录整理。
王晨宇：实现游戏的主要程序框架。实现各状态页面间的切换。统一 UI 与程序间的接口，实现塔的攻击和怪物所受的伤害和 debuff。提供绘图建议，修复若干 bug。
章可循：帮助构建程序框架，实现怪物的链表存储，一小部分绘制工作。
周屹赫：游戏创意，美术设计。实现怪物、塔、血量、菜单、界面等绘制工作，检测玩家对按钮的点击。

5.2. 难点和挑战

5.2.1. 杜祐陞

(1) 在绘制地图的时候，由于对于塔防游戏的不熟悉，画了很多地图然后作废，然后后期想要做改动时，又因时间和地图大小跟怪物生成路径的原因，所以只好放弃更精致的路径。
(2) 由于在代码能力上的不足，在分工上我一直都是帮忙居多，看着其他人在添加功能后，有时我也会想多进行一些改动和补强，但每每都以失败收场，希望往后能化这次的经验为动力，制作出更优秀的游戏。

5.2.2. 王晨宇

(1) 在选题前轻视了了解并熟练使用图形库的难度，也想挑战一下自己，所以我们选择了一个相较于回合制固定判断局面相对简单的棋牌类游戏来说更为陌生、细节更多的塔防类游戏。
(2) 前期由于组员对自己的分工不太明确、对整个项目构成的规划有限，导致工作进度被拖延。后期分工明确后还是有了一定效率，但是因为代码正确性与调试手段的缺乏，在 debug 上花了过多精力和时间。同时在课外分配在项目的时间上也可以优化，提高效率。
(3) 在画塔的攻击范围的时候，攻击范围的圆弧不能画到地图之外，为此要计算圆和上下左右四个边界的交点，求出实际上要画的弧度区间。

5.2.3. 章可循

(1) 在画塔的攻击范围的时候，攻击范围的圆弧不能画到地图之外，为此要计算圆和上下左右四个边界的交点，求出实际上要画的弧度区间。

(2) 在用链表存储防御塔时，遇到了一些访问非法内存导致程序崩溃的情况，在队友的帮助下调试出来了，以后使用链表时应该多加注意。

5.2.4. 周屹赫

(1) 我至今没有找到如何改变一个像素点的函数（用了一下 setpixel 但是不是想要的效果），无奈之下只好用画一个小正方形和画一条长度为 0 的线段的方法来模拟画一个点，但是效率极低下，本来尝试着把一张位图通过二维矩阵的形式读进来的，结果发现画一张像素点 4w 的图竟然要花 1s 的时间，所以又打消了导入精致图片的念头。

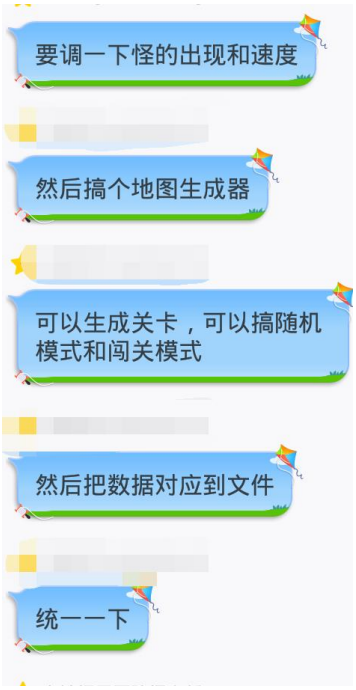
(2) 该图形库没有图层的概念，并且我也没能找到获取当前像素点颜色的办法所以涉及到擦除的部分就经常不得不重画整张图存位图的代价又太高，为此我不得不在游戏设计的交互性和美观性方面做了很大的牺牲。比如单体攻击的弹道和范围攻击的圆圈，我最后都没能很好的实现出来，总之在诸多限制之下，勉强完成了一个还算美观的图形界面。

6. 合作纪要

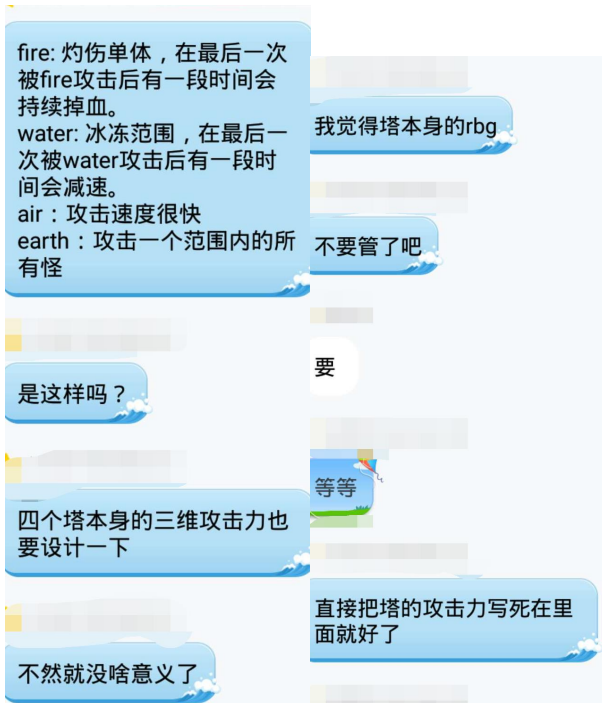
第一次小组会议


你应该没重画吧
只会重画跑道
然后画怪
那可能会局部闪？...
那等我能画出来再说好了
不能每次重新画整块画布，不然会有很严重的频闪？
还是应该重画需要修改的小块而不动大背景
攻击表示感觉可以花在怪物的圆里
画
这样的话塔的范围用颜色叠加这种感觉也还行
05-23 09:27
画在怪物的圆里是啥子意思？
频闪可太难受了...
每次画所有怪物都不能接受的吗。。
所有怪物的确是每次都会被更新的
不可能不重画
其实把游戏调的慢一点我觉得会好一点
不重画背景？...
不的呀
你应该没重画吧

解决频闪问题


要调一下怪的出现和速度
然后搞个地图生成器
可以生成关卡，可以搞随机模式和闯关模式
然后把数据对应到文件
统一一下

文件交互



塔的设计

还有很多很多…

7. 总结

7.1. 程序亮点或创新之处

1. 相对于传统的塔防游戏，我们的项目采用了 rgb 三个参数作为怪物的生命值，不仅使怪物血量变化具有一致性，玩家能用肉眼直观地观察场上形势，丰富了怪物的状态变化；也便于 graphics 库能够以较快的速度画出图像，提高了程序运行效率。

2. 相对于一般用 graphics 库实现的程序，我们跳出框架，自己实现封面、按钮、菜单等界面，图形、颜色、功能更丰富，让程序变得更美观。同时也创造性地在怪物身上增加由简单线条或图形构成的状态，使游戏的情况更丰富。

3. 由于怪物移动、塔的攻击是一个动态、连续的过程，而 graphics 库中没有图层的概念，重构图层会极大影响玩家体验。所以我们在表现上做了许多权衡与取舍，在实现中做了很多优化与改进。如局部重画、利用计算几何知识减少冗余线条、利用二分方法加快对象选取、利用链表减少遍历时间等。最大程度上平衡了游戏的视觉丰富性与性能与效果的要求。

7.2. 应用知识点

1. 菜单：开始界面的各种功能选取及之间的来回切换
2. 链表：新建怪物、更新怪物状态、删除怪物
3. 按钮：在界面、游戏中均有大量自行实现的按钮模块
4. 鼠标：交互界面的切换以及游戏过程中的主要操作
5. 键盘：暂停、选择防御塔等快捷键的实现
6. 文件：从文件中读取地图信息或存档信息，并将玩家存档记录

到文件中

7. 算法：二分、计算几何

7.3. 心得体会

7.3.1. 杜祐陞

这次的大作业真的让我获益良多，它让我深刻的体认到，先前所学的基础知识要实际运用并整合有多么困难，让我明白自己个人能力的不足，还好有大家的分工合作，才能完成这次的作业。在此我要特别感谢我的组员，他们每个人都能快速并有效率地提出方案和解决问题，带领没有头绪的我，明确指出我的工作范围并不厌其烦的指导我该如何走下一步，让我能在我能力所及之处尽力和大家一同完成这次的大作业，我觉得这种大家一起 debug 的经验会让我未来更向往去加入团队中，和大家一同完成一个项目。

7.3.2. 王晨宇

在整个工作中还是有很多收获，例如章可循让我对回调函数、链表等知识有了更清楚的认识，并且在我构建程序框架的过程中给了我很大帮助和启发；周屹赫和杜祐陞在 UI、游戏策划上的付出和精彩成果让我眼前一亮，给我们队伍很大鼓舞。同时一队队员一起寻找代码出现的问题的经历还是令人感动。

同时也有一些教训。前期由于组员对自己的分工不太明确、对整个项目构成的规划有限，导致工作进度被拖延。后期分工明确后还是有了一定效率，但是因为代码正确性与调试手段的缺乏，在 debug 上花了过多精力和时间。同时在课外分配在项目的时间上也可以优化，提高效率。

7.3.3.章可循

本次大作业由于对组内同学的实力比较了解，所以我们选择了自选项目，希望能够挑战自我，做出不错的作品。事实上效果也确实不错，我学到了多文件代码的制作，对于图形库的使用，也和队友进行了分工合作，一起制作出一个作品的感觉十分令人感动。

7.3.4.周屹赫

本次大作业总的来说还是收获颇多，懂得了如何分工合作写项目，懂得了如何写多个文件的大程序，也学到了很多图形方面的知识，队友都很靠谱，就是库不太靠谱（效率低就算了，甚至有明显的 bug），以后希望能够继续努力，在合作中尝试其他的角色。

8. 参考文献和资料

陈建海老师课程 PPT《多文件程序》

陈建海老师课程 PPT《libgraphics 使用》

《C 语言程序设计》