附件1：实验报告中的诚信设计

数据结构与算法导论

实验报告

实验题目：\_\_\_电梯控制演示程序\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

姓 名：\_\_\_\_\_\_耿翊中\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

学 号：\_\_\_\_2021213382\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

日 期：\_\_\_2022.6.26\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

自我评分：\_\_\_\_\_\_\_\_\_【 A+ 】\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

自我评分说明：A+，A，B+，B，B-，C，D，分别对应分数95、90、85、80、75、70、60

诚信声明

本人郑重承诺：本实验程序和实验报告均是本人独立学习和工作所获得的成果。尽我所知，实验报告中除特别标注的地方外，不包含其他同学已经发表或撰写过的成果；实验程序中对代码工作的任何帮助者所作的贡献均做了明确的说明，并表达了谢意。

如有抄袭，本人原因承担因此而造成的任何后果。

特此声明。

签名：\_\_\_\_耿翊中\_\_\_\_\_\_\_\_\_\_\_\_

日期：\_2022.6.26\_\_\_\_\_\_\_\_\_\_\_

程序引用说明

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 引用项 | 来源 | 代码引用行数 |
| 1 | 彩蛇方块闪烁 | <https://blog.csdn.net/qq_52670477/article/details/119284366>  CSDN | 1 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
| 小计 | | | 1 |

总代码行数\_\_\_\_500\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_; 引用占比\_\_\_\_\_0.002%\_\_\_\_\_\_\_\_\_\_\_

1、实验简介

功能要求：

* 编写算法根据不同楼层按键控制电梯运行；
* 能够演示电梯上下行运行状态

程序效果展示请老师观看附件的展示视频

2、程序框架

头文件game.h中列出了所用的函数，和相对应的功能，展示如下：

//1.数据属性：基本数据成员分析

//坐标属性，电梯属性，按键属性

//坐标属性

typedef struct pointXY

{

int x;

int y;

}MYPOINT;

//电梯属性

struct Lift

{

int num;//电梯长度

MYPOINT xy[length]; //电梯最多有10节

char postion;//电梯的方向

int curnum = 0; //梯中人数（待启用）

};

//按键的属性

struct Button

{

MYPOINT fdxy; //按键的位置

int nature ; //按键的属性（1为上按键，2为下按键，3为电梯内按键）

int flag = 0; //按键是否存在，0表示存在，1表示不存在

};

//方向

enum movePostion { right = 77, left = 75, down = 80, up = 72, freeze = 3 };

//主窗口

static HWND hwnd = NULL;

//2.过程分析：函数分析

//初始化主电梯

void initmaiLift(struct Lift\*lift);

//初始化副电梯

void initsubLift(struct Lift\*lift);

//画电梯

void drawLift(struct Lift\* lift);

//移动电梯

void moveLift(struct Lift\* lift);

//初始化按键

void initButton(struct Button\* button, int x, int y, int na);

//画电梯内部按键

void drawButton(struct Button\* button);

//画下按键

void drawdoButton(struct Button\* button);

//画上按键

void drawupButton(struct Button\* button);

//消去按键

void eatButton(struct Button\* button, struct Lift\* lift);

//展示提示

void showwords();

//按空格键暂停

void pauseMoment();

//画楼层框架

void drawfloor();

//显示电梯有几人

void showbutton(struct Lift\* lift);

//调错函数

void showerror();

//绘制副电梯是否启用

void drawifuse(int x);

3、关键代码实现

3.1 实现电梯移动void moveLift(struct Lift\* lift)

//移动电梯

void moveLift(struct Lift\* lift)

{

//除了第一节之外，后面每一节都是前面一节的坐标

for (int i = lift->num; i > 0; i--)

{

lift->xy[i].x = lift->xy[i - 1].x;

lift->xy[i].y = lift->xy[i - 1].y;

}

//第一节的处理

switch (lift->postion)

{

case down:

lift->xy[0].y += 10;

break;

case up:

lift->xy[0].y -= 10;

case freeze:

break;

default:

break;

}

}

用结构体的方式制作电梯，此函数实现了电梯坐标的改变，来实现电梯移动。

3.2实现按键消去 void eatButton(struct Button\* button, struct Lift\* lift)

//消去按键

void eatButton(struct Button\* button, struct Lift\* lift)

{

if ((lift->xy[0].y+5 == button->fdxy.y|| lift->xy[0].y - 5 == button->fdxy.y) &&

( (lift->postion==down&&button->nature==2)

|| (lift->postion == up && button->nature == 1)

|| button->nature==3 ) )

{

button->flag = 0;

}

}

判断何时消去适当的按键，当电梯下行时消去向下请求的按键，当电梯上行的时候消去向上请求的按键。

3.3主电梯移动调度原理

if (lift[1].postion == freeze)

{

for (int i = 1; i <= N; i++)

{

if (button[i].flag == 1)

{

//showerror();

if (button[i].fdxy.y < lift[1].xy[0].y)

lift[1].postion = up;

else

lift[1].postion = down;

}

}

}

else if (lift[1].postion == up)

{

int maxheight = -1;

int jud = 1;//判断当前是否为最高，如果不是最高，则值修改为0

for (int i = 1; i <= N; i++)

{

if (button[i].fdxy.y - 2 < lift[1].xy[0].y&&button[i].flag)

jud = 0;

}

if (jud)

{

lift[1].postion = freeze;

}

}

else if (lift[1].postion == down)

{

int maxheight = -1;

int jud = 1;//判断当前是否为最低，如果不是最低，则值修改为0

for (int i = 1; i <= N; i++)

{

if (button[i].fdxy.y + 2 > lift[1].xy[0].y&&button[i].flag)

jud = 0;

}

if (jud)

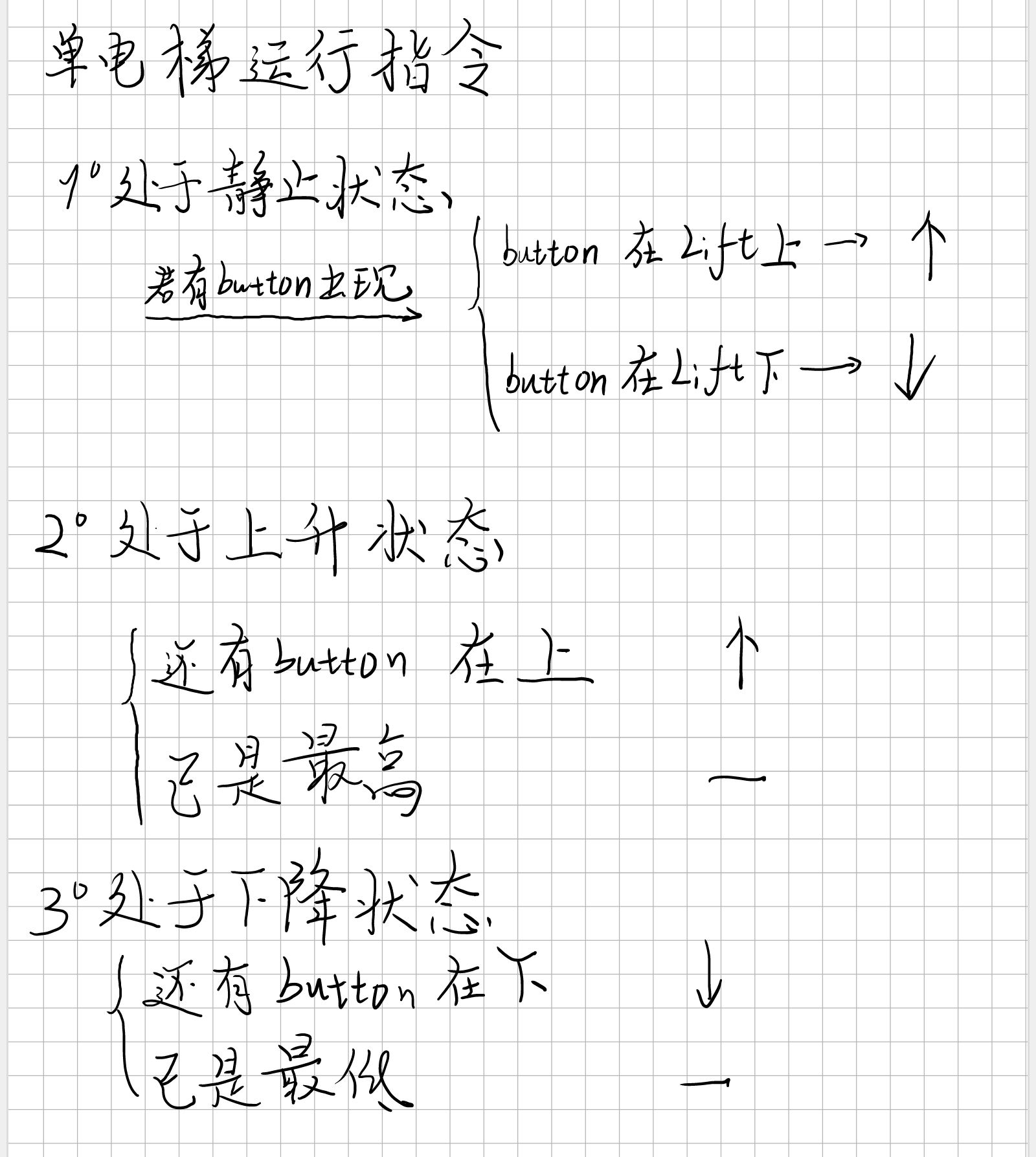
{

lift[1].postion = freeze;

}

}

调度命令如下：



4、不足

虽然做了双电梯的设计，但没能较合理的实现双电梯接人调度，要考虑的情况太多了。同时虽然设计了每个楼层的上下键和电梯内按键，但看起来图里比较繁琐，给同学看了有看不懂的情况。

5、心得体会

程序效果展示请老师观看附件的展示视频

对于电梯要求展示运行，开始属实有点头疼，但后来想到要实时展示电梯，其实与写过的贪吃蛇程序非常相像。那么我不妨将蛇看作电梯，实物看作乘客，这样一来，贪吃蛇吃食物就变成了电梯载乘客。在此启发下，我用贪吃蛇程序的easyx框架写了电梯的框架，并沿用了坐标和结构体设定，因此我的电梯由两节组成，其实在设定里我有想过将电梯的载人量用长度表示出来，也写了一个纯载人的程序，但是最后考虑到题目要求是要体现调度，遂放弃了这个想法。

再然后就是对电梯运行的理解很有意思，因为一度住过顶层的房子，所以对电梯有尤为丰富的观察经验（上学快迟到的抓狂）。所以我一开始就确信要用状态来界定电梯的运行，也就是分电梯此时状态来调度，当电梯分别是静止，上行，下行以不同对策来改变电梯运行。

在按键上本来想对电梯内外按键分别处理，后来我发现内按键和外按键本质上是一样的，你在电梯里上楼的时候电梯依然可以被小伙伴使坏在其它楼层用上键拦住，所以便都采用了一种结构体，只是添加了标记，方便消去按键。

最后在两个电梯的调度上属实有些复杂，需要满足诸多条件，并且我的程序受限于楼层少，速度快，不停顿，只有更多的楼层才有更清楚的看法和应用价值。于是最后只是选择了一种特殊情况来调用第二个副电梯。有更多的时间的话我还想再尝试。

毕竟，做一个完善的电梯系统，再给自己开一个后门让电梯先接自己，是多酷的一件事情啊。