棉花模拟采集系统

终期报告

c 语言课程设计

专业班级：自动化 2202
小组成员：冯天瑞（U202214953） 梁栢杰（U202290032）
指导老师：周纯杰、何顶新、汪国有、左峥嵘
周凯波、彭刚、高常鑫、陈忠
上交时间：2023 年 4 月 22 日

目录

一、编写背景

21 世纪初，互联网飞速发展并迅速普及到各行各业，而属于基层的农作物业也不再靠传统的纯人力耕种。收割机，运输车，智能仓库等近现代的工业机械也逐渐走进了中国的田地。

而棉花行业就是农业中最重要的产业之一，它产量大，生产成本低，使棉织品价格比较低廉。棉花能制造成多种规格的纺织物，衣服，布，棉签，手套，鞋子，口罩，被子，甚至钞票和医用绷带等都可以由棉花制造而成，可谓用途广泛，在我们的生活中随处可见了。棉花从三千多年前就已经传入中国，但一直到元朝时期才开始广泛种植，多数种植在西北地区，黄河、长江流域。

棉花在生活中的应用已非常广泛，棉花中的棉可以卖钱还能做成棉被，棉衣在冬季成为生活取暖的重要物品，而且棉籽还可以用来榨油满足生活的需要，剩下的秸秆还可以作为燃料生火取暖和做饭，棉花的皮还可以磨成肥料饲养牲畜。

棉花的种植一般分为播种，管理和采集三个部分，而本项目主要体现在采集和管理的部分。通过棉花种植园的面积和地理位置，计算棉花的收获方式和收获量，模拟棉花采集的全过程。

二、目标功能

本项目主要通过模拟棉花采集的过程。根据其投入的收割机数量和型号、种植棉花的地理位置、棉花田的面积和形状，计算出其产出的棉花量、收割时间和收割形式，并模拟和制作出棉花从采集到装库全过程的动画。以此提供用户对自己实际种植棉花情况的参考和建议收获方式。该项目通过鼠标与键盘直接进行控制。用户将鼠标移至需要操作的区域进行点击来显示不同界面，同时通过键盘来完成各种参数的输入功能。

三、运行环境和配置

一、硬件接口

处理器：Intel Pentium 166 MX 或以上。

硬盘：空间 500MB 以上。

屏幕适配器：VGA 接口。

系统运行内存：要求 32MB 以上。

二、软件接口

开发软件工具：Borland C++

文字编辑工具：visual Stdio Code

操作系统：DOS WINDOWS 9X/ME/2000/XP/WINDOWS 10/WINDOWS 11

四、需求分析

棉花自动采摘系统模拟

主要功能说明：根据农田面积进行采摘路径规划、模拟智能农机进行采摘作业、对采摘棉花进行统计、并运输到棉花集中站进行储存、盘点、出库和入库等功能。

根据本选题要求进行需求分析，可见要求制作的软件系统是一款棉花采摘农业实践的仿真模拟系统,需要最大程度地结合实际，为实际农业自动化生产服务，现根据题目要求及实际查阅资料，有以下核心用户需求，即软件核心功能：

1） 本软件分用户使用，用户登录进入操作主页面，便可开始模拟操作

2） 通过实际情况，用户可以根据不同地区需求设置棉花生产参数，土地参数，系统自动推荐棉花种植种类，农机类型

3） 进入模拟流程，系统根据土地类型，农机类型，自动规划采摘路径，模拟智能农机进行采摘，给出采摘用时。

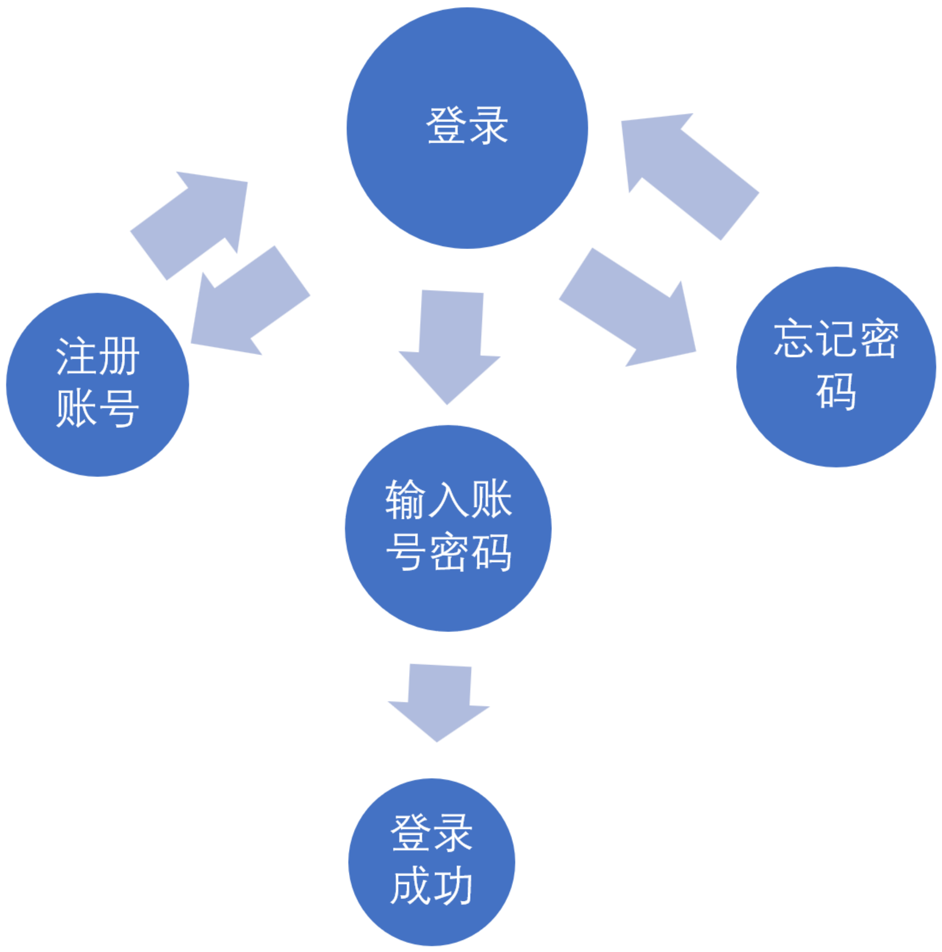
4） 进入仓储界面，自行操作仓库出入库流程

5） 进入参数列表，修改过往参数

五、系统设计

登录界面流程：

1）输入账号和密码完成登录并进入主界面，也可选“忘记密码”或注册新账号。忘记密码的验证通过注册账号时输入的手机号来确认。注册账号或者找回密码回就会返回登录界面。



主界面流程：

主界面中主要给用户进行选择。用户的功能主要分为：

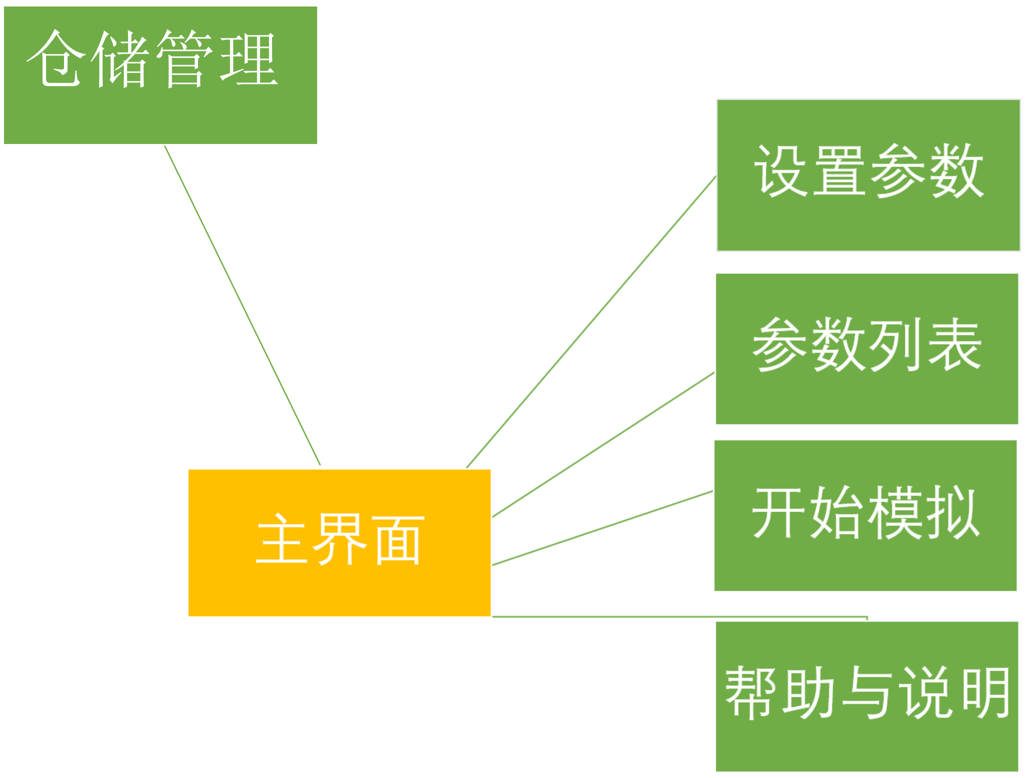
1）设置参数

2）参数列表

3）开始模拟

4）仓储管理

5）帮助与说明



设置参数流程：

需要设置的参数包括：

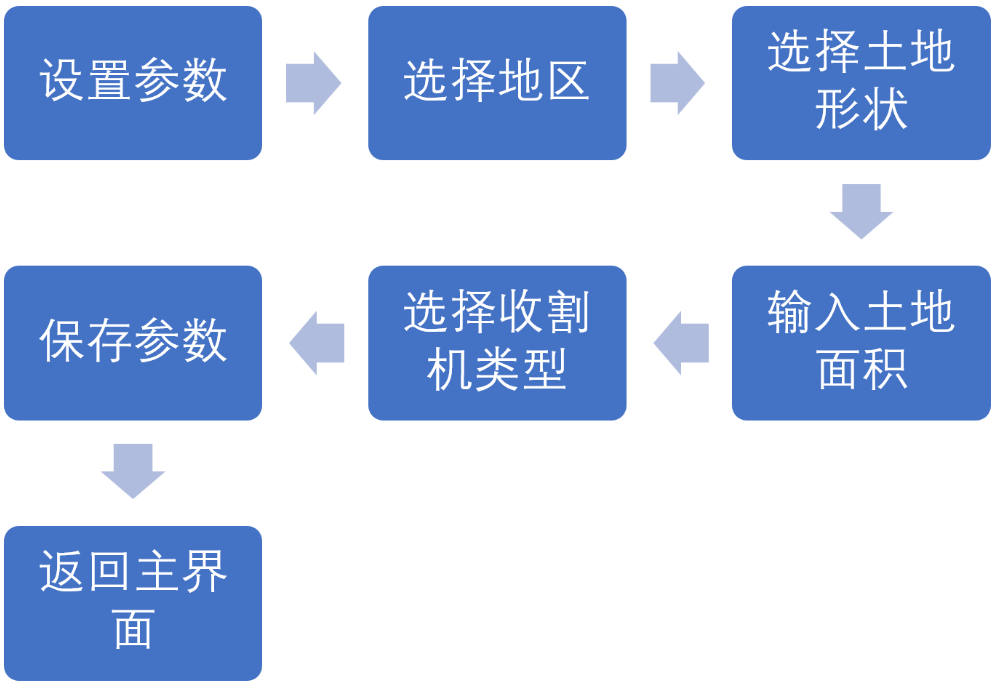
1）选择地区并输入参数名，本项目提供中国三个适合种植棉花的地区进行选择，包括西北地区，黄河地区和长江地区。

2）选择土地形状，本项目提供了三种土地形状，包括矩形，圆形和三角形。

3）输入土地面积，利用键盘输入在该形状下的土地大致面积。

4）选择收割机类型，采集棉花常用的收割机类型分为垂直式和水平式，本项目亦提供这两种选择。

5）保存参数并返回主界面。

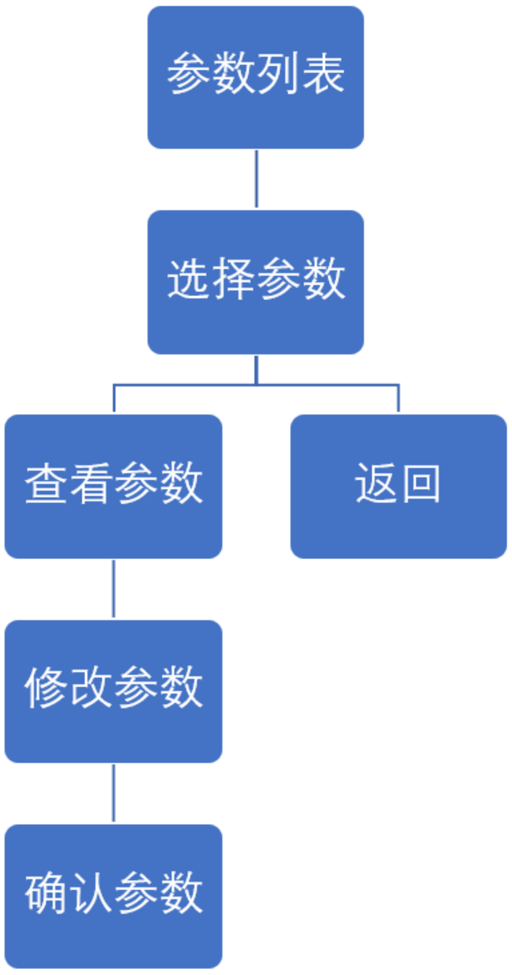


参数列表流程：

1）选择参数，选择之前保存过的参数。

2）查看参数，查看该参数具体内容。

3）修改参数，选择需要修改的参数内容，并进行修改。



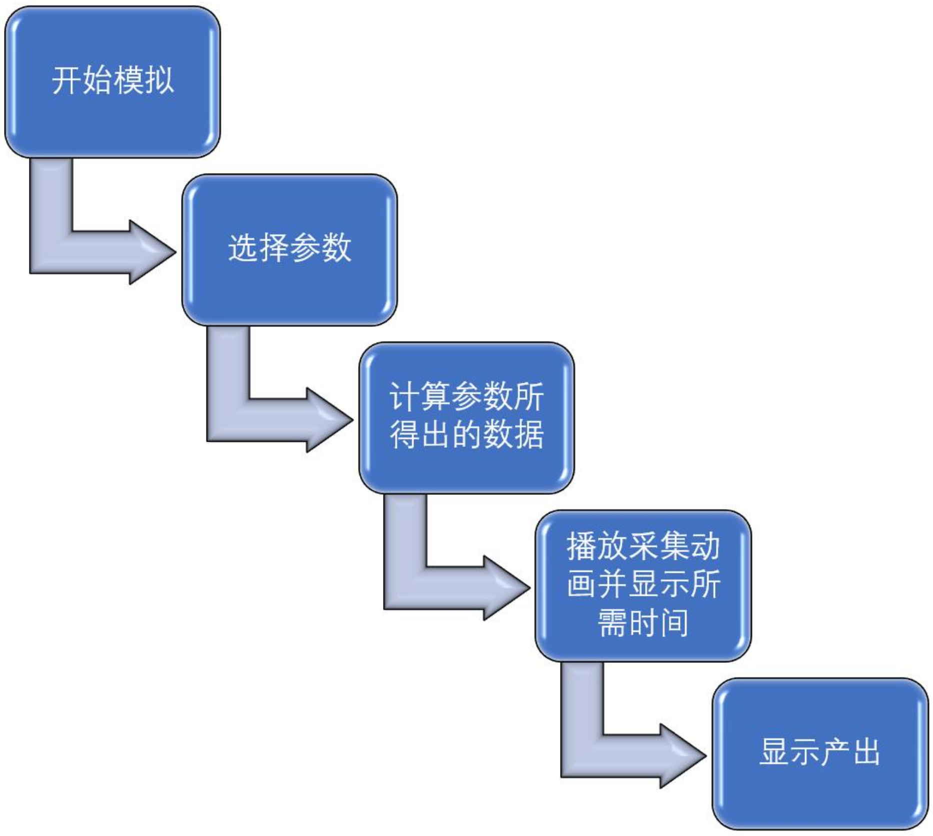
开始模拟流程：

1）根据已编辑的参数选择参数。

2）计算参数所得出的数据，根据选择的参数，计算出采集所需时间，采集方式和产出。

3）播放采集动画，根据土地面积和土地形状，播放相应的采集动画。

4）显示产出，显示计算出的棉花产出。

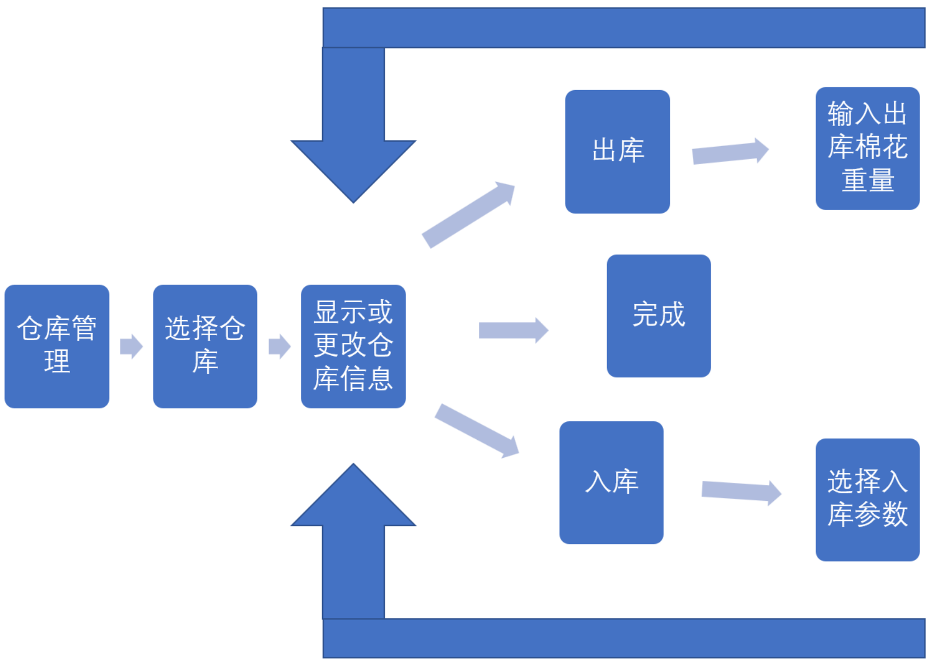


仓储管理流程：

1） 选择仓库名，或者新建仓库。

2） 盘点并显示该仓库的棉花总数，所剩仓库容量。并可进行出库看或入库的操作改变库存。

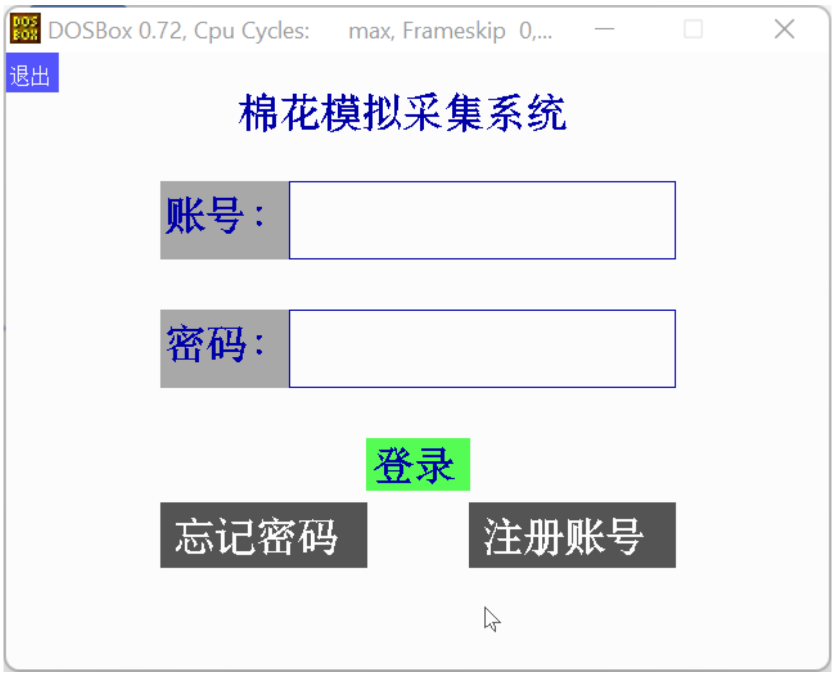
3） 完成操作后退出仓储管理，并返回主界面。

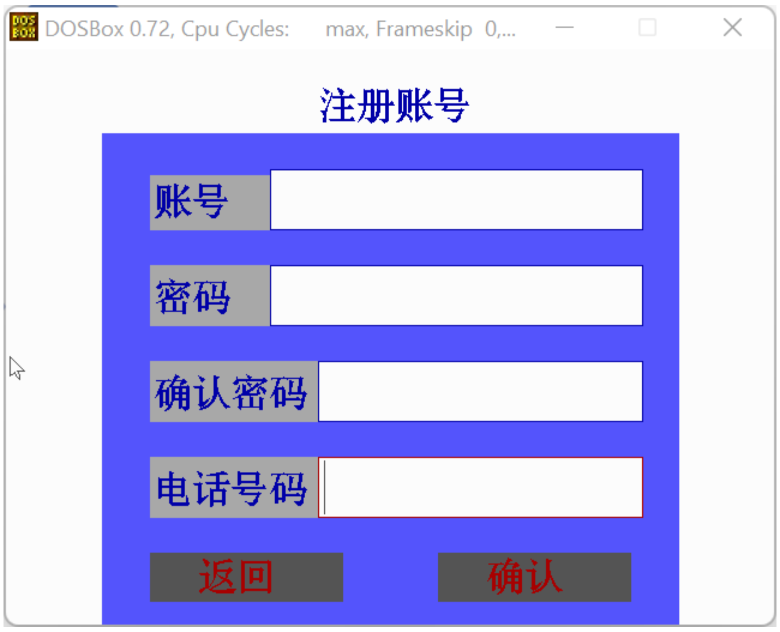


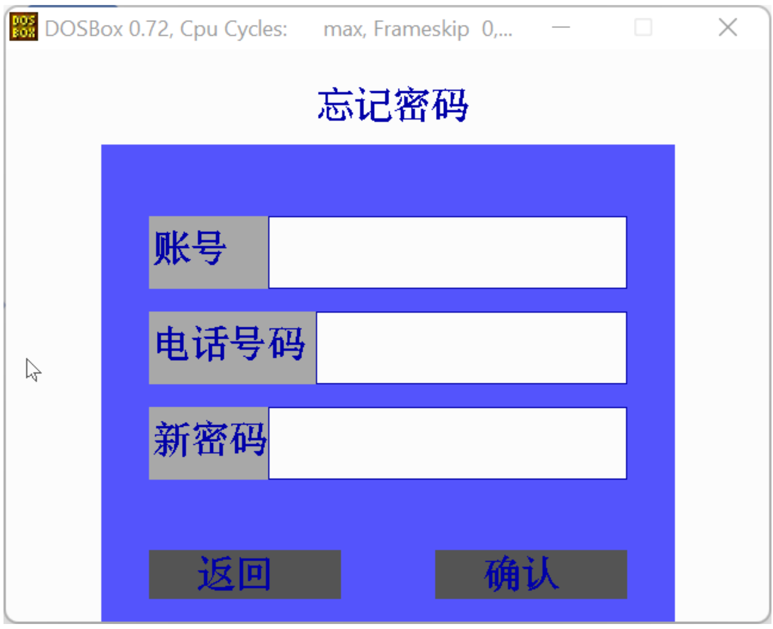
六、界面设计

登录界面：

输入账号和密码完成登录并进入主界面，也可选“忘记密码”或注册新账号。忘记密码的验证通过注册账号时输入的手机号来确认，验证成功则会提供其令其重新输入密码。

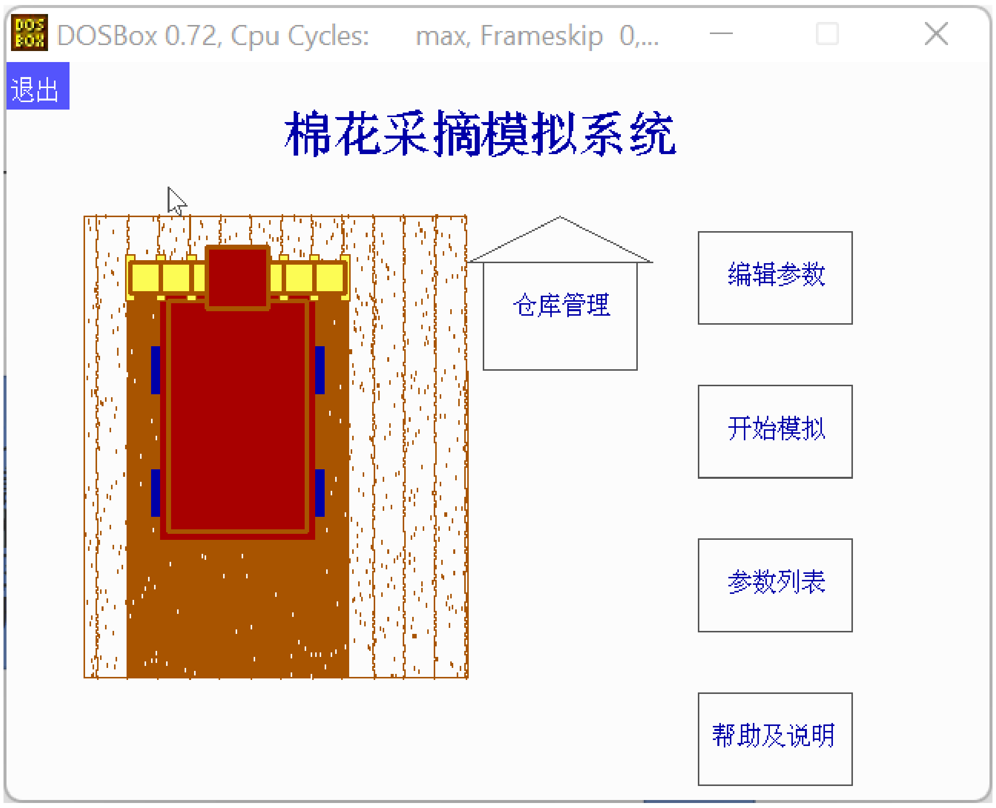






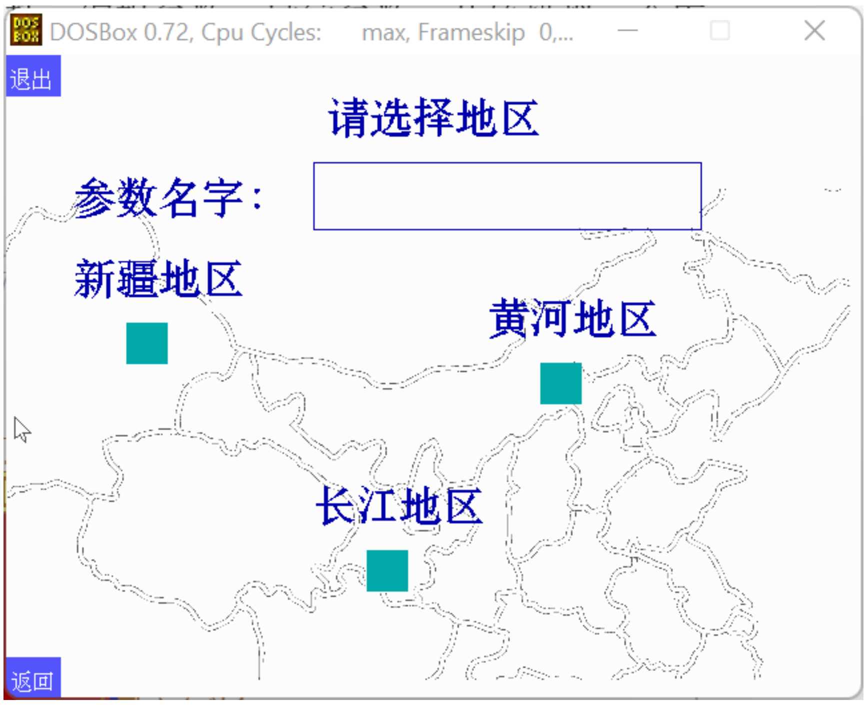
主界面：

运行之后的第一个界面，可以通过鼠标选择接下来的动作。可选择的动作包括：编辑参数，过往参数，开始模拟，仓库管理和帮助及说明。

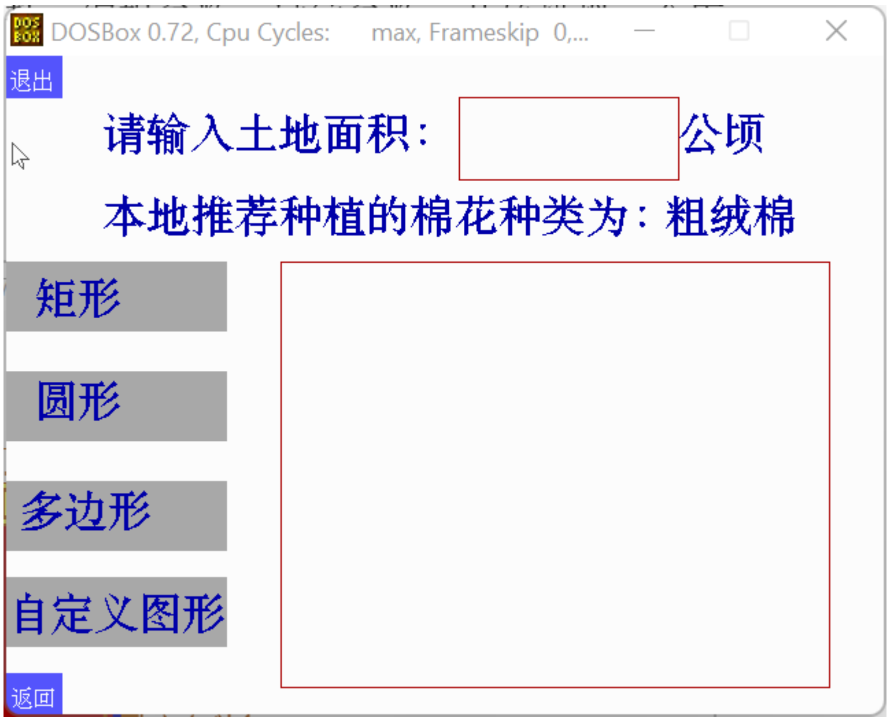


编辑参数界面：

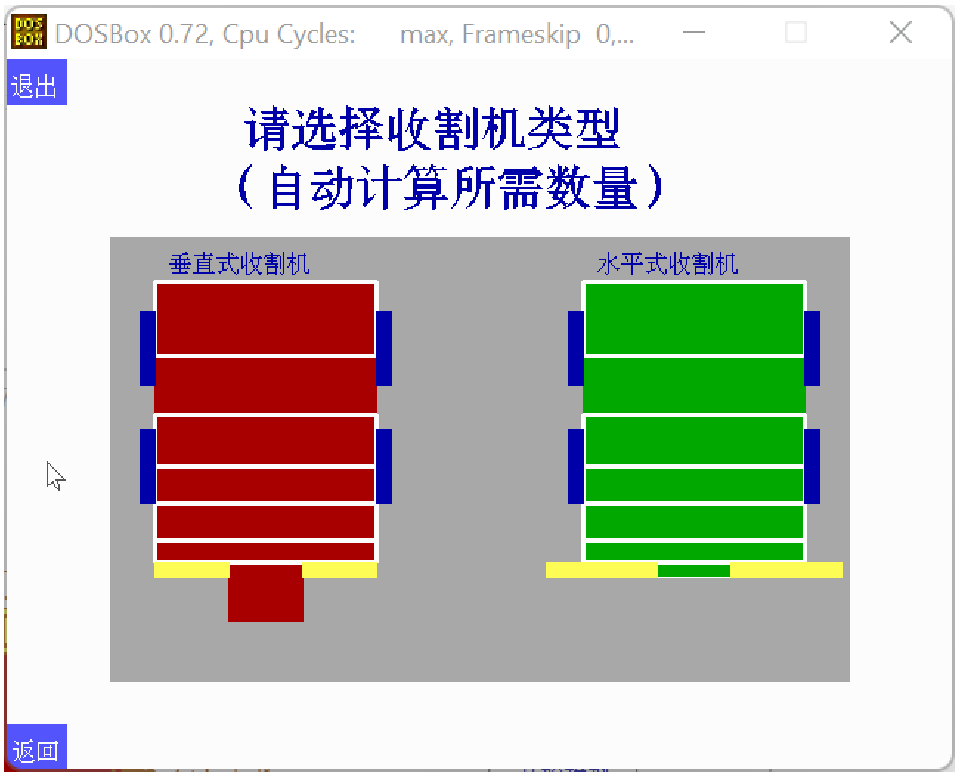
界面 1：选择产区，三个棉花产区，鼠标移动到地图上对应地区则该地区变为红色。



界面 2：通过键盘输入土地面积，并显示推荐种植的棉花类型。

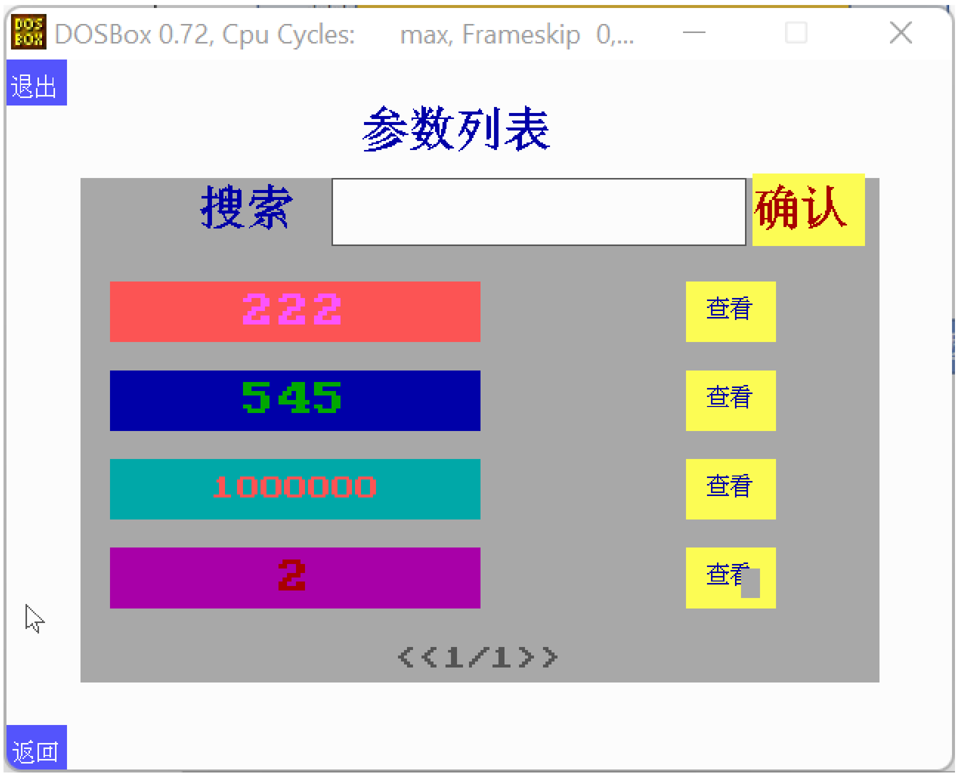

DOSBox 0.72, Cpu Cycles: max, Frameskip 0,...
退出
请输入土地面积：
公顷
本地推荐种植的棉花种类为：粗绒棉
矩形
圆形
多边形
自定义图形
返回

界面 3：通过鼠标选择收割机的类型。

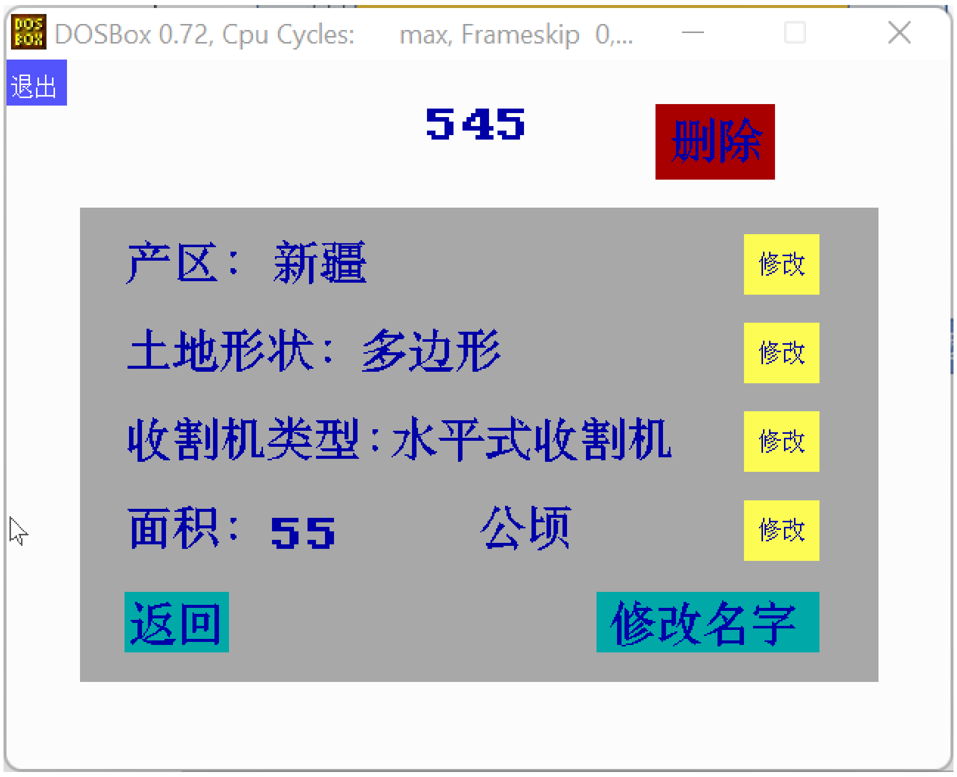


参数列表界面：

界面 1：选择参数

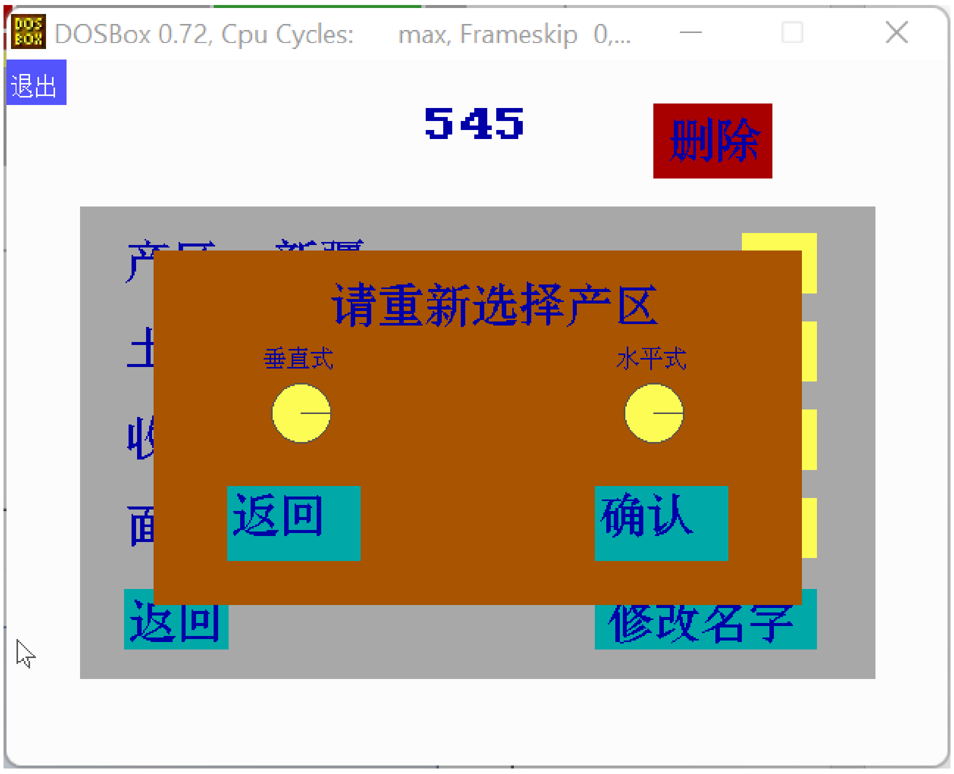


界面 2：选择参数后，选择返回可重新选择参数，查看可查看该参数的内容。


DOSBox 0.72, Cpu Cycles: max, Frameskip 0,...

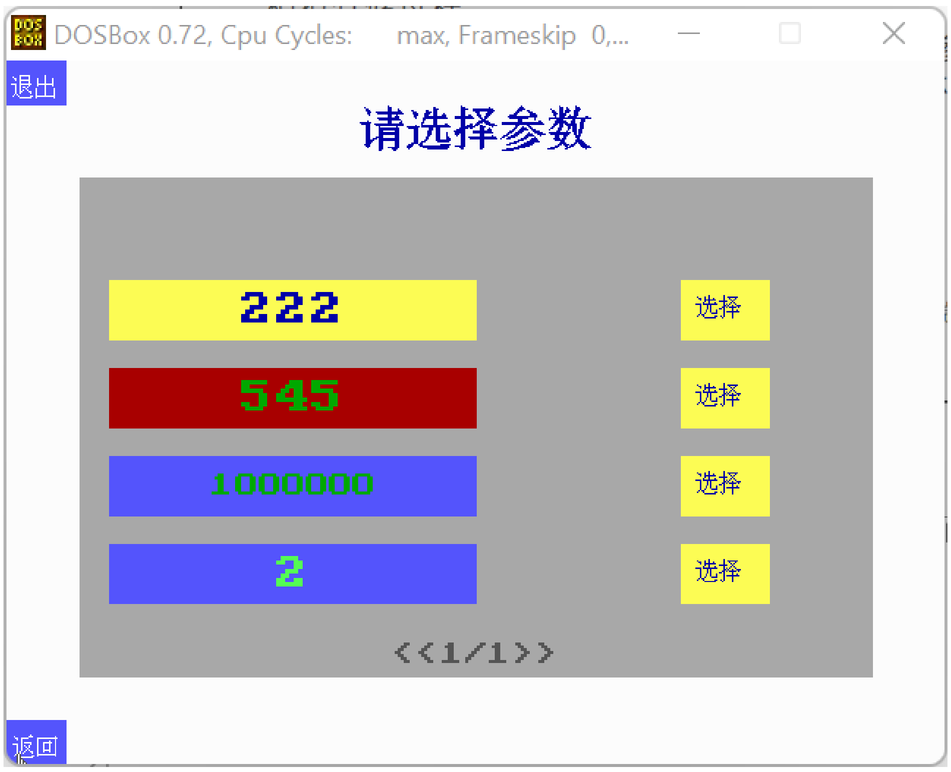
产区： 新疆
土地形状： 多边形
收割机类型：水平式收割机
面积： 55 公顷

界面 3：查看参数后，可以修改指定的参数。

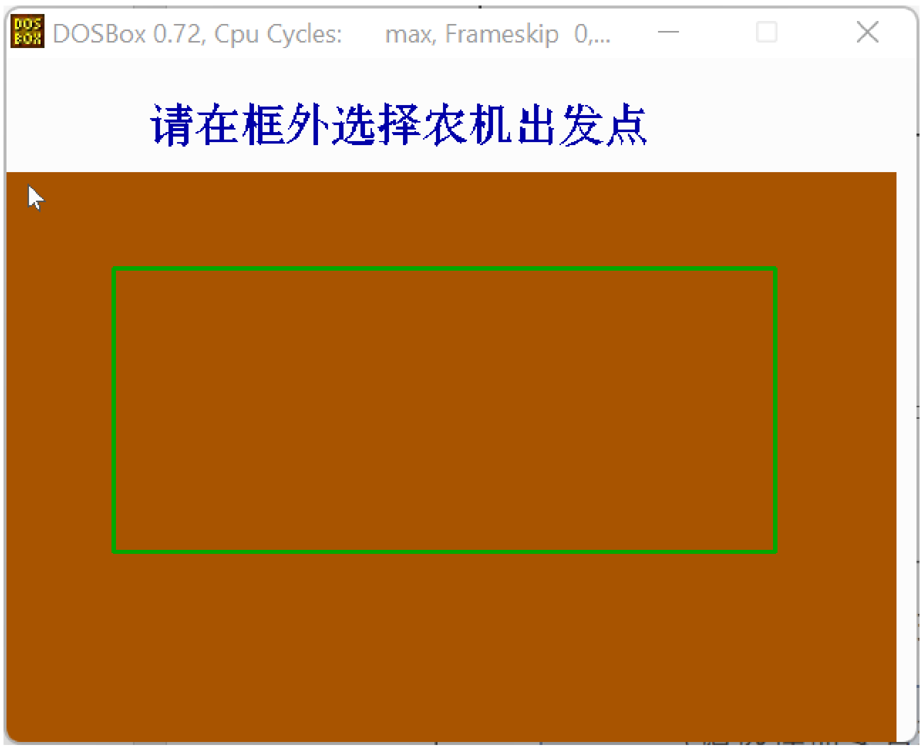

DOSBox 0.72, Cpu Cycles: max, Frameskip 0,...

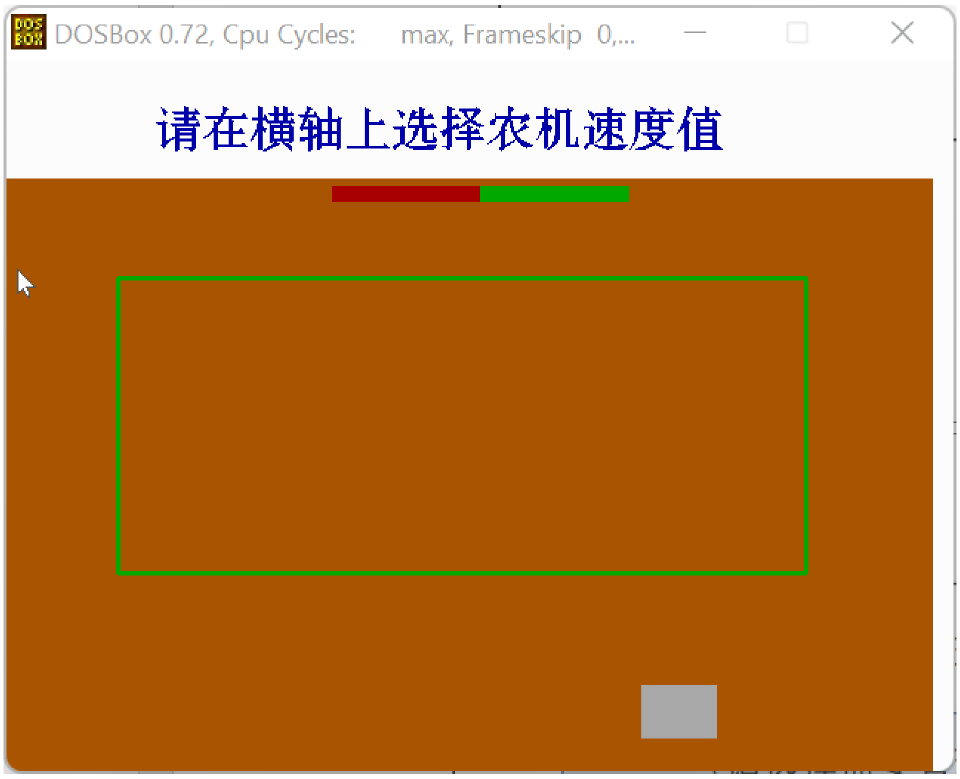
开始模拟界面：

界面 1：选择需要模拟的参数。



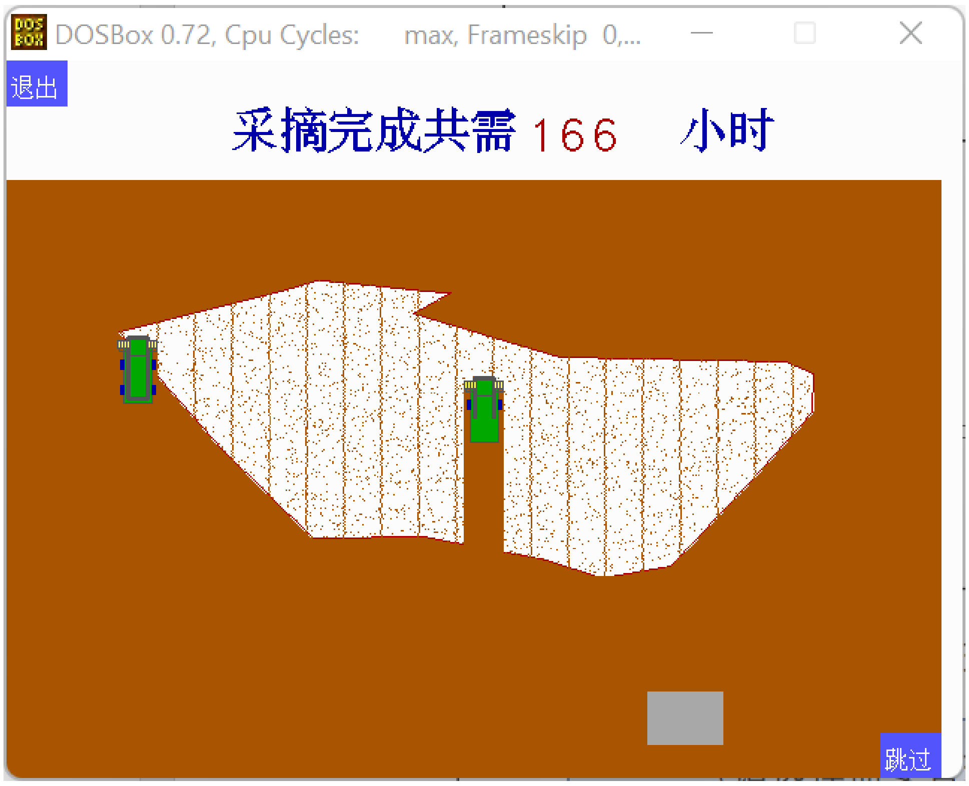
界面 4：选择出发点和农机设定速度



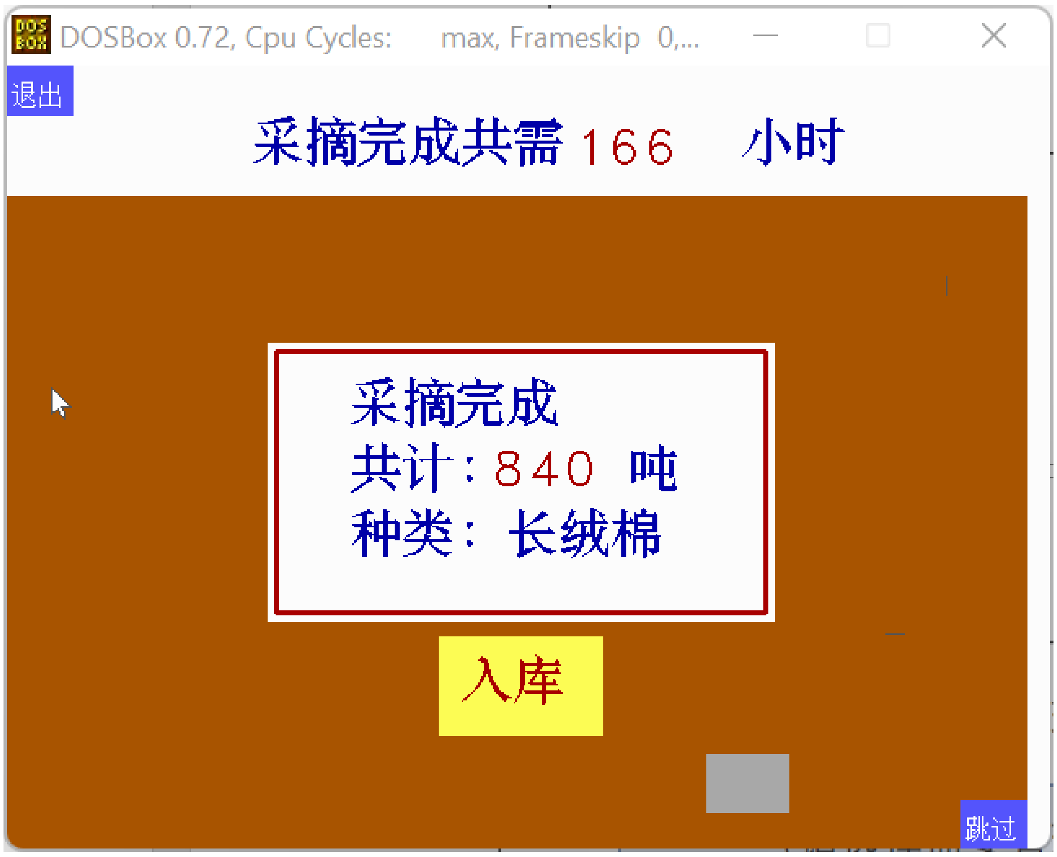


请在横轴上选择农机速度值

界面 3：播放采摘动画，并分别显示所需时间和收获量。



采摘完成共需 166 小时


DOSBox 0.72, Cpu Cycles: max, Frameskip 0,...

退出

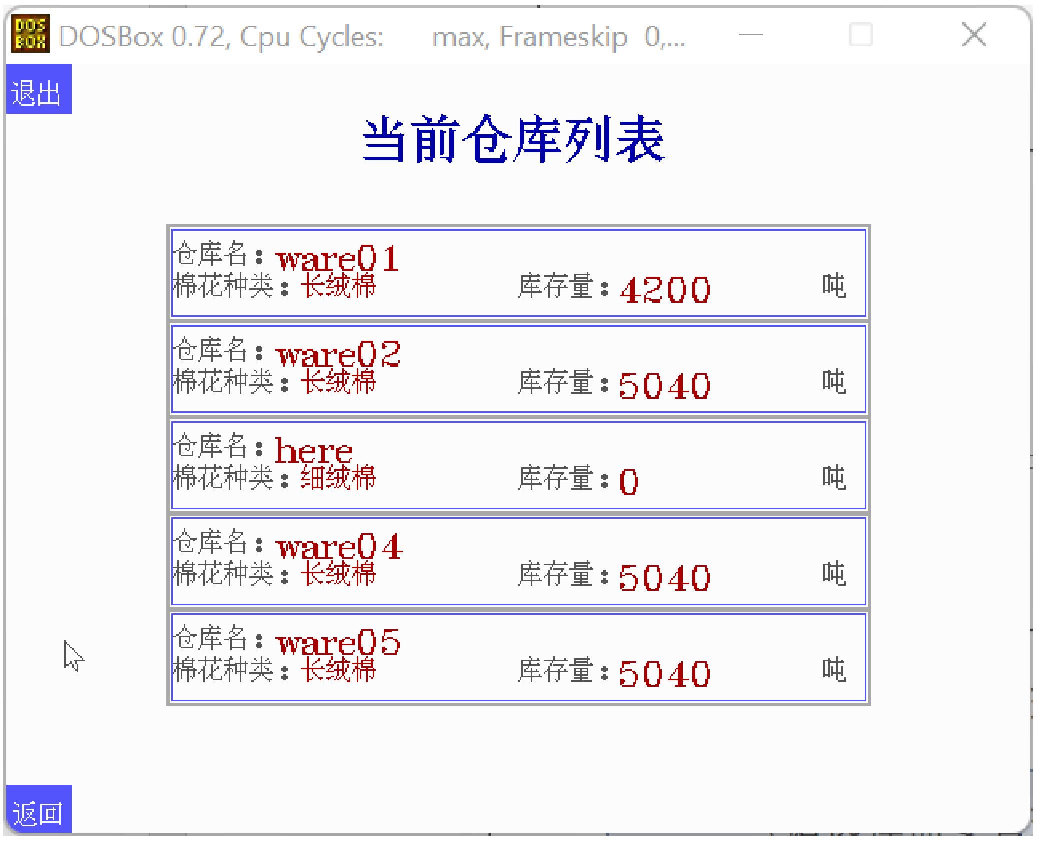
采摘完成共需 166 小时

采摘完成
共计：840 吨
种类：长绒棉

入库

跳过

界面 4：选择入库仓库并入库


DOSBox 0.72, Cpu Cycles: max, Frameskip 0,...

退出

当前仓库列表

仓库名：ware01
棉花种类：长绒棉　　库存量：4200　吨

仓库名：ware02
棉花种类：长绒棉　　库存量：5040　吨

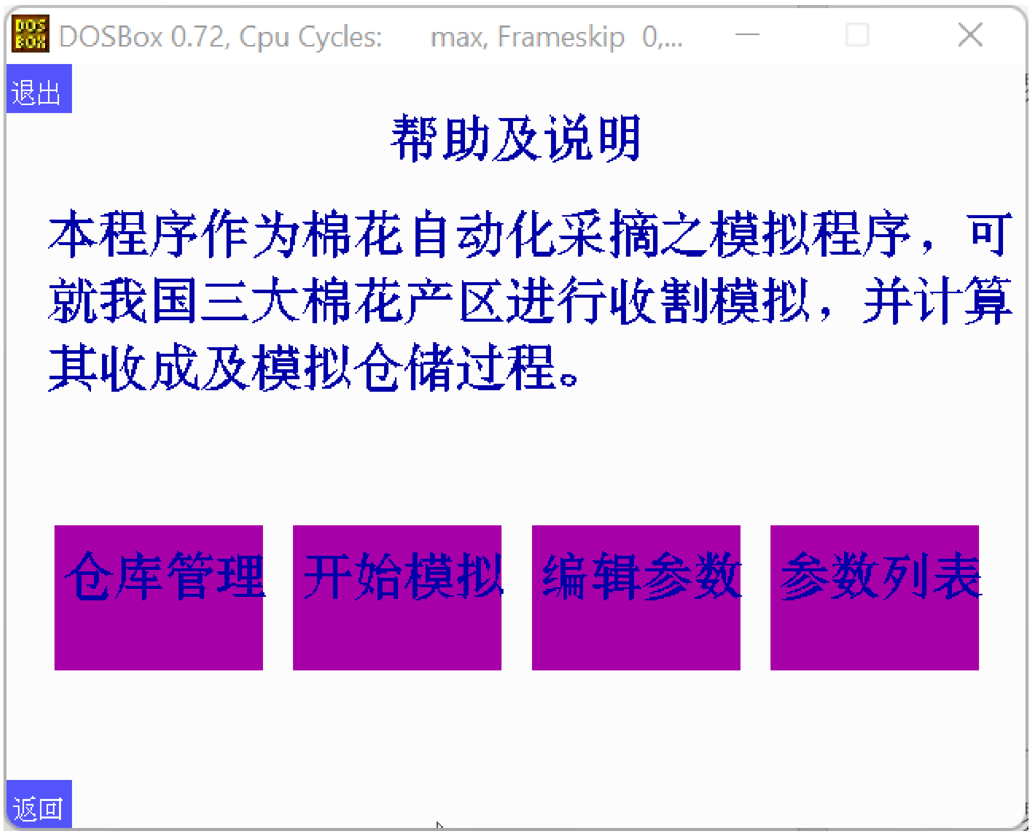
仓库名：here
棉花种类：细绒棉　　库存量：0　吨

仓库名：ware04
棉花种类：长绒棉　　库存量：5040　吨

仓库名：ware05
棉花种类：长绒棉　　库存量：5040　吨

返回

帮助与说明页：将鼠标置于各个按钮可以查看操作说明



DOSBox 0.72, Cpu Cycles: max, Frameskip 0,...

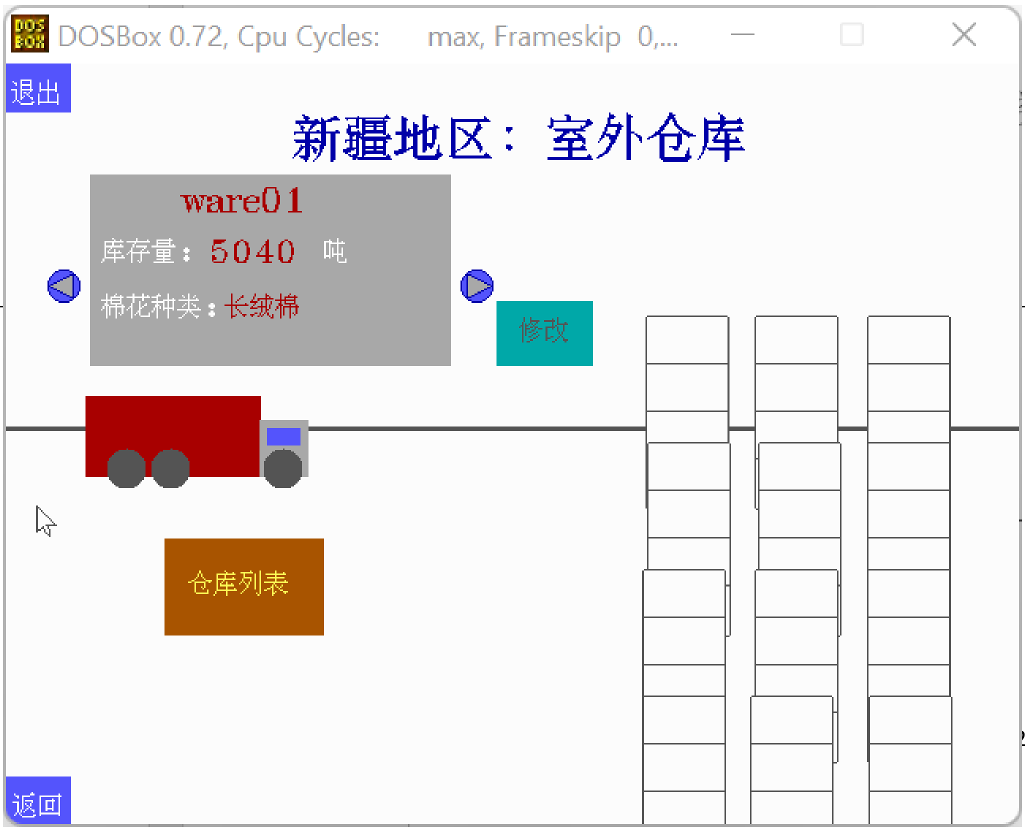
退出

帮助及说明

本程序作为棉花自动化采摘之模拟程序，可就我国三大棉花产区进行收割模拟，并计算其收成及模拟仓储过程。

仓储管理页：

界面 1：仓库主界面，显示当前参数设置地区及棉花种类对应库存量



DOSBox 0.72, Cpu Cycles: max, Frameskip 0,...

退出

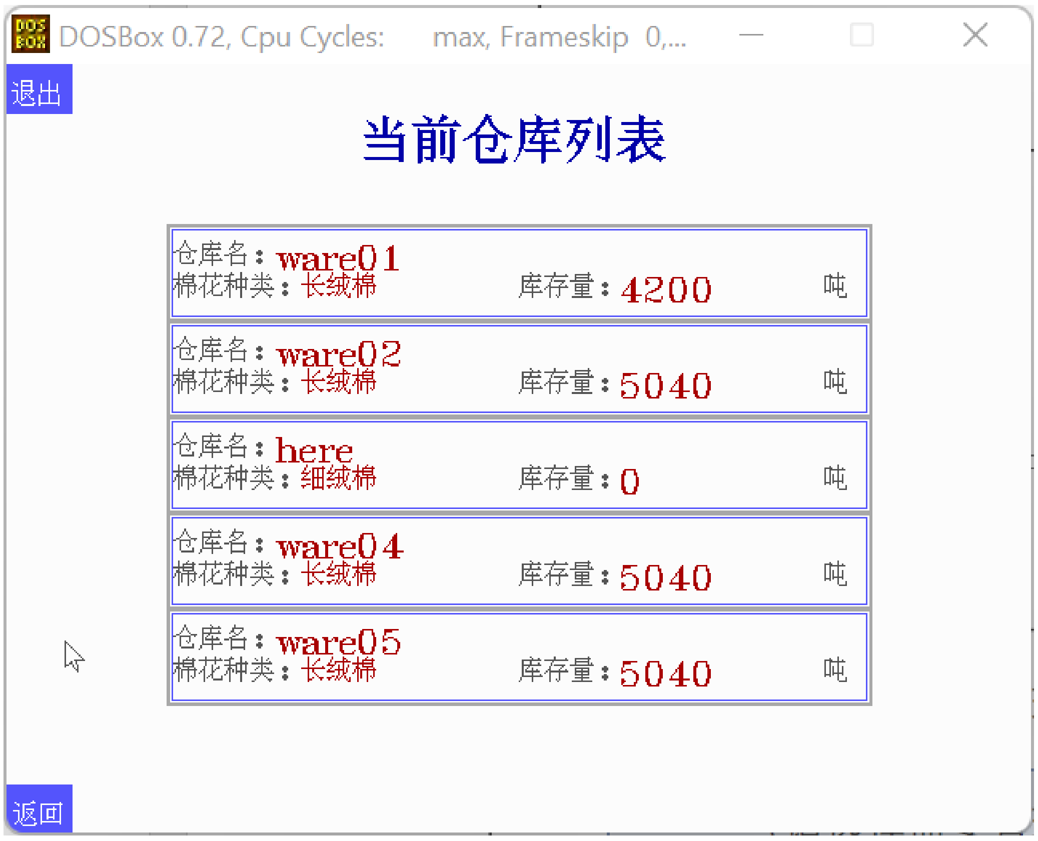
新疆地区：室外仓库

ware01

库存量：5040 吨

棉花种类：长绒棉

界面 2：仓库列表，可以更改当前仓库



DOSBox 0.72, Cpu Cycles: max, Frameskip 0,...

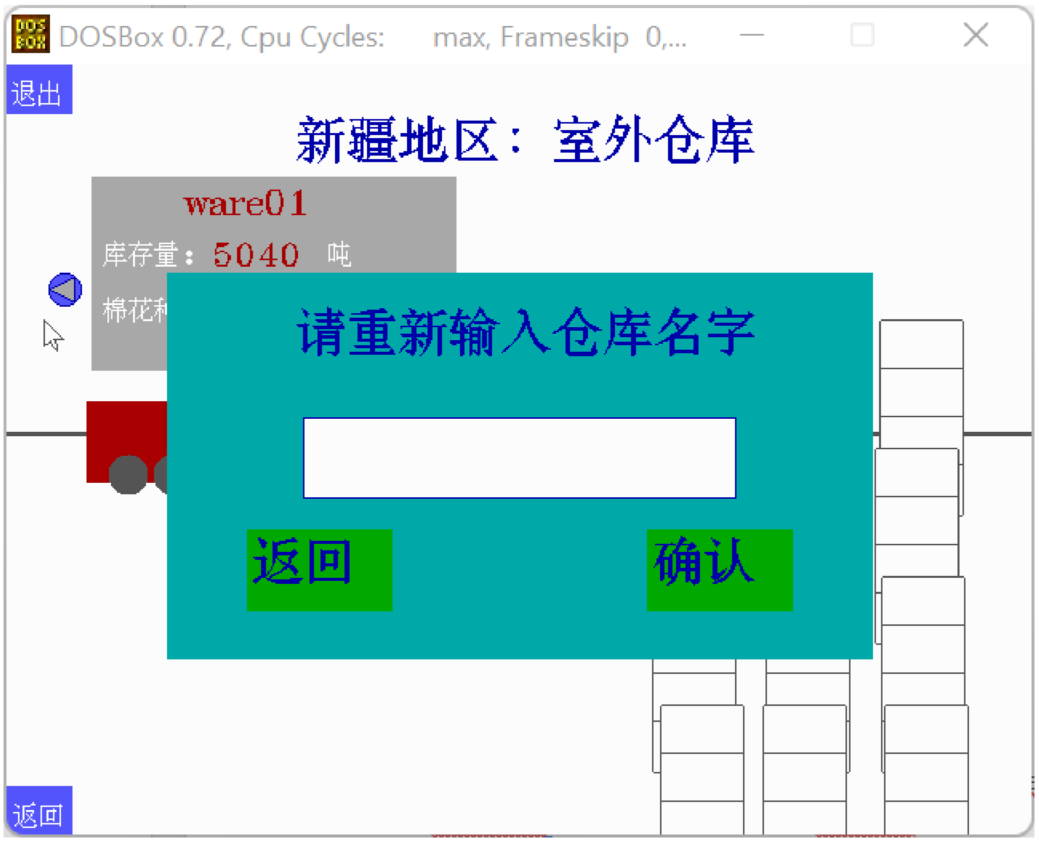
退出

当前仓库列表

仓库名：ware01 棉花种类：长绒棉	库存量：4200	吨
仓库名：ware02 棉花种类：长绒棉	库存量：5040	吨
仓库名：here 棉花种类：细绒棉	库存量：0	吨
仓库名：ware04 棉花种类：长绒棉	库存量：5040	吨
仓库名：ware05 棉花种类：长绒棉	库存量：5040	吨

返回

界面 3：仓库名字修改



DOSBox 0.72, Cpu Cycles: max, Frameskip 0,...

退出

新疆地区：室外仓库

ware01

库存量：5040 吨

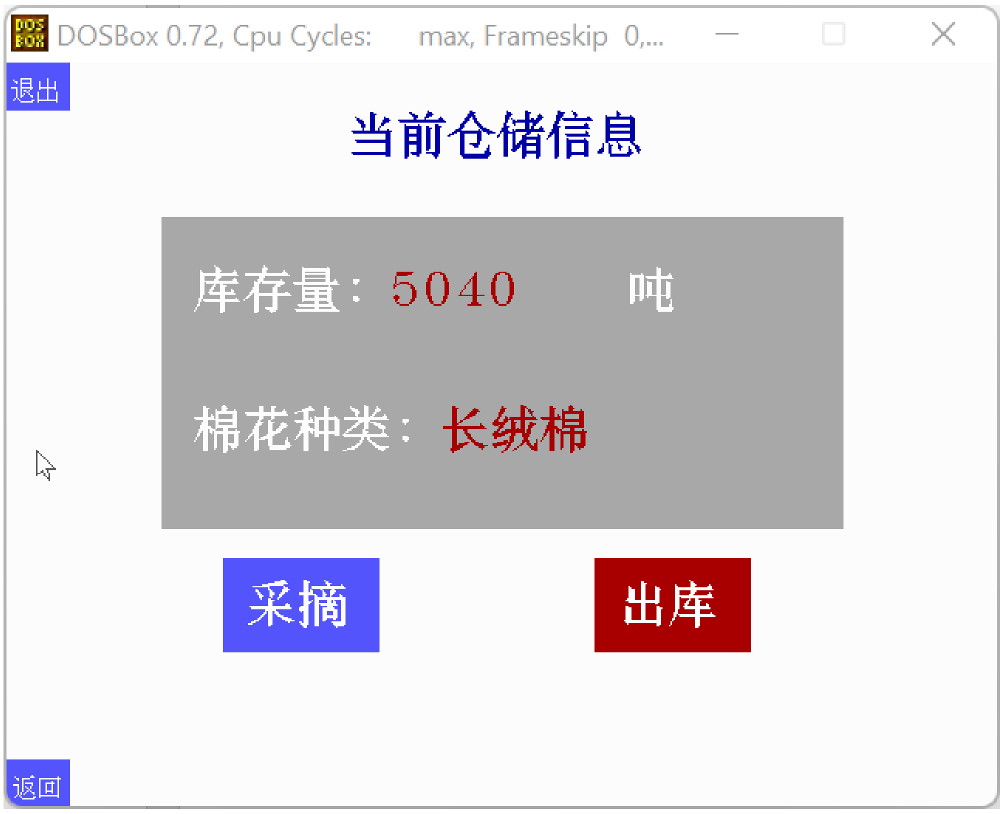
棉花种

请重新输入仓库名字

返回 确认

返回

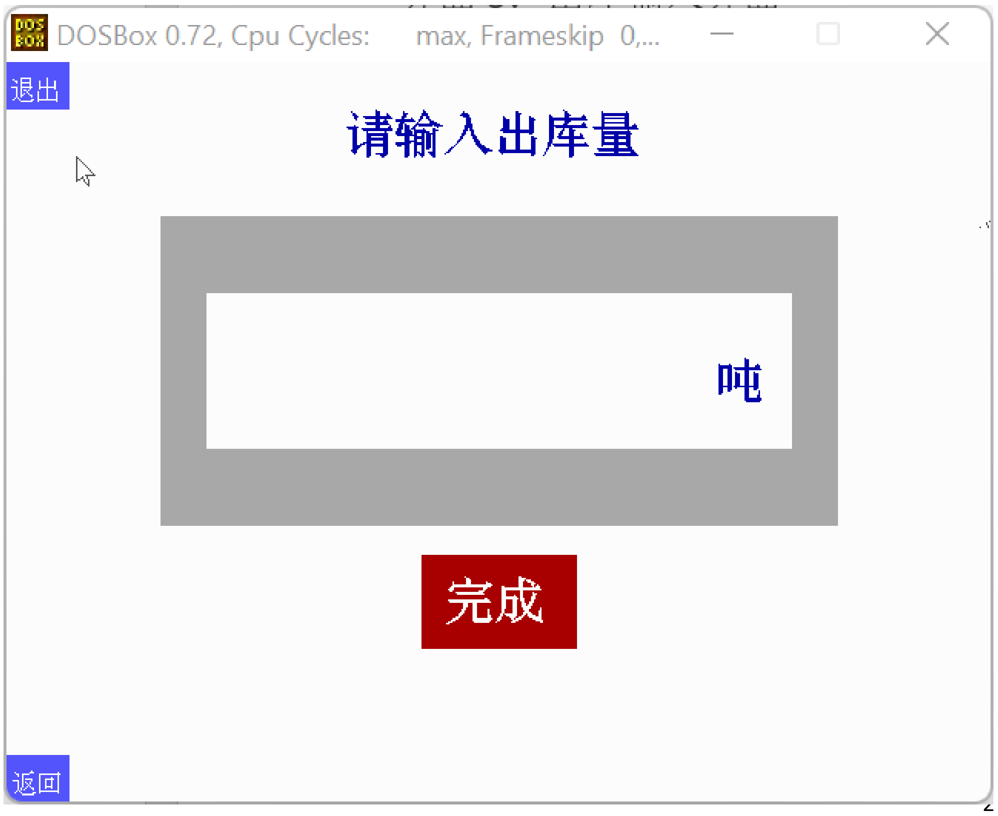
界面 4 详细仓库信息及操作，采摘则进入模拟界面，出库则进入出库输入界面



库存量：5040　　吨

棉花种类：长绒棉

界面 5：出库输入界面



七、 函数代码

<1>头文件部分

1、 COMMON.H

```
#ifndef _COMMON_H_
#define _COMMON_H_

#include<stdio.h>
#include<stdlib.h>
#include<graphics.h>
#include<dos.h>
#include<malloc.h>
#include<bios.h>
#include<math.h>
#include<string.h>
#include "mouse.h"
#include<conio.h>
#include ".\\HZK\\HZ.H"

//extern U_ware here[5];
extern int k;
extern int mode;
extern int mode1;
//extern char str[15];
extern struct User*h;
void quit(void);//the leave button in every page
void skip(void);
void next(void);
void last(void);
//void text_input(char *str,int x1,int y1,int x2,int y2,int t_x,int t_y,int t_size);//the input function for English characters and numbers
//int ch_to_int(char *str);
void input_text(char *id, int x, int y, int charnum, int color, int flag);

#endif
```

2、 DSTART.H

```
#ifndef _DSTART_H_
#define _DSTART_H_

#include "COMMON.H"
#include "LOGFUN.H"
#include "START.H"

#define x_max 550
#define y_max 320
#define x_start 50
#define y_start 120
#define point_max 500
#define dense_points_max 500

#define tracktor_num_max 5
//#define delaytime 20
#define tra_time 80
#define tracktor_w 25
#define co_pak_w 10
#define pick_bar 600
#define tracktor_l 30
#define tra_start_l 50
#define tra_start_d 35
#define pick_ph 2.4
#define Xinjinag_har 2.1
#define Huanghe_har 1.3
#define Chnagjiang_har 1.1
#define p_Up_arrow 0x4800
#define p_Donw_arrow 0x5000
#define p_Enter 0x1C0D

void init_tracktor01_f(int x,int y);//initialize the tarcker 01
void init_tracktor01_b(int x,int y);
void init_tracktor02_f(int x,int y);
void init_tracktor02_b(int x,int y);

void earth_fill01(int x,int y);
void earth_fill02(int x,int y);
void earth_fill03(int x,int y);
void earth_cover01(int x,int y);
void earth_cover02(int x,int y);
void earth_cover03(int x,int y);

void init_tracktor01_r(int x,int y);
void init_tracktor01_l(int x,int y);
void init_tracktor02_r(int x,int y);
void init_tracktor02_l(int x,int y);

void init_picker_f(int x,int y);
void init_picker_b(int x,int y);
void init_picker_r(int x,int y);
void init_picker_l(int x,int y);

void init_picker01_f(int x,int y);
void init_picker01_b(int x,int y);
void init_picker01_r(int x,int y);
void init_picker01_l(int x,int y);

#endif
```

3、 EDIT.H

```
#ifndef _EDIT_H_
#define _EDIT_H_

void edit01_screen(void);
int edit01(struct Parameter *abc);//The first in edit:choose area
void edit02_screen(struct Parameter *abc);
int edit02(struct Parameter *abc);//The Second in edit:define size and shape of the field
void edit03_screen(void);
int edit03(struct Parameter *abc);//Choose the type of tractors and cotton

void edit(void);

#endif
```

4、 START.H

5、 PAST.H

6、 HOME.H

```
#ifndef _HOME_H_
#define _HOME_H_

#define ware_full 29999
#define num_ware 5

void draw_home01(void);//the page of n-w warehouse
//void draw_home00(void);//the page of Y-H warehouse
void warehouse_list(struct user_warehouse * w);
void press_warelist(int *re);
void draw_warehouse(void);//draw the picture of warehouse
void cot_mount(int x,int y);//draw the cotton in warehouse
void draw_trunk(void);//draw the trunk int the page
void in_warehouse(struct user_warehouse *now);//draw the board which show the cotton in warehouse
void detailed_warehouse(long int count);//detail message of warehouse
void press_detwarehouse(void);
void press_home(int *c_t);//add the press moudule
void out_warehouse(void);//page of the cotton out
void press_outware(void);
void out_finish(void);

typedef struct user_warehouse
{
 char ware_name[15];
 int cotton_type;
 long int total[3];
}U_ware;

#endif
```

7、 HELP.H

```
#ifndef _HELP_H_
#define _HELP_H_


void draw_help(void);
void help01(void);//Draw the Help and
```

```
Explanation page
        void edit_help(void);
        void start_help(void);
        void past_help(void);
        void ware_help(void);
        void help(void);

        #endif
```

8、 LOGFUN.H

```
        #ifndef _LOGFUN_H_
        #define _LOGFUN_H_

        #include "HOME.H"
        #define PAR 10
        #define dense_points_max 500

        typedef struct Input
        {
         int x1;//框的坐标
         int y1;
         int x2;
         int y2;
         char string[19];
         char length;//输入的最大长度
         char cursor;//光标位置
         char flag;//0 代表不接受输入，1 代表可以接受
输入
        }INPUT;




        // 产区，面积，形状（坐标），收割机类型，
名字

        // 产区 place：a--新疆；b--黄河；c--长江；分
别对应：长绒棉，细绒棉，粗绒棉
        // 面积 S：123（数字）
        // 形状 shape：a--矩形；b--圆形；c--多边形；
d 自定义形状
        // 坐标 x 和坐标 y：c/d 才用
        // 收割机类型 type：a 垂直/b 水平
        // 名字 name：abcdefg

        typedef struct Parameter
        {
            char name[10];//名字
            char place;//产区
         char shape;//土地形状
            char type;//收割机类型
         char S[10];//面积
         int x[dense_points_max];
         int y[dense_points_max];//坐标
         int lenxy;//坐标数量
        }parameter;

        typedef struct User
        {
            char username[10];
            char password[10];
            char phonenumber[12];
         struct Parameter parameter[PAR];//参数列表
         int lenpar;//参数个数
         struct user_warehouse here[5];//仓库
        }user;



        void wr_user(char username[],char
password[],char phonenumber[]);
        int changepassword(char username0[],char
newpassword0[],char phonenumber0[]);
        int logg(char username0[],char password0[]);


#endif
```

9、 LOGIN.H

```
        #ifndef _LOGUSER_H_
        #define _LOGUSER_H_

        void loginit_screen(void);
        void loginit(void);
        int logenter(void);



#endif
```

10、 PARAMETER.H

```
        #ifndef _PARAMETE_H_
        #define _PARAMETE_H_

        //键盘输入函数
        int input_s(int x,int y,struct Input *word,int
size,int mode);

	//判断名字是否为空 and 重复
	int judgename(char name[]);

	//判断土地面积是否为数字和空
	int judgeS(char name[]);

	//把参数写进 h，然后把 h 写进文件里面
	void wr_parameter(struct Parameter* abc);

	//修改产区
	int changeplace(int par);

	//修改土地形状
	int changeshape(int par);

	//修改收割机类型
	int changetype(int par);

	//修改面积
	int changeS(int par);

	//修改名字
	void changeparname(int par);

	//删除参数
	void deletepar(int par);

	//把当前登录的用户重新写进文件
	void wr_h(void);

	//把第二个参数赋给第一个参数
	void parcpy(struct Parameter *a,struct Parameter
*b);

	//选择参数
	int choosepar(void);

	//搜索
	int search(char name[]);

	//改仓库名字
	void changewarename(int wi);
#endif
```

11、 REGISTER.H

```
	#ifndef _REGISTER_H_
	#define _REGISTER_H_

	void log_register(void);//注册
	void register_screen(void);
	int username_same(char *name,char
*number);//判断注册账号是否存在和电话号码是否
正确


#endif
```

12、 RESET.H

```
	#ifndef _RESET_H_
	#define _RESET_H_


	void reset_screen(void);
	void reset(void);





	 #endif
```

<2>源文件部分

1、 DSTART.C

```
	#include "DSTART.H"

	// initialize the tarcker 01
	void init_tracktor01_f(int x, int y)
	{
	 int i;
	 setlinestyle(0, 0, 1);
	 setcolor(DARKGRAY);
	 // The main rectangle
	 setfillstyle(1, RED);
	 bar(x + 3, y - 3, x + 22, y + 38);
	 rectangle(x + 3, y - 3, x + 22, y + 38);
	 setlinestyle(0, 0, 3);
	 rectangle(x + 6, y - 6, x + 19, y + 35);

	 // The samll bars
	 setcolor(DARKGRAY);
	 setlinestyle(0, 0, 1);
 setfillstyle(1, YELLOW);
 bar(x, y, x + 25, y - 3);
 rectangle(x, y, x + 25, y + 3);
 for (i = x; i <= 25 + x; i += 2)
 {
        bar(i - 1, y + 1, i + 1, y - 4);
        rectangle(i - 1, y + 1, i + 1, y - 4);
 }

 // The driver site
 setfillstyle(1, RED);
 bar(x + 7, y - 5, x + 18, y + 6);
 rectangle(x + 7, y - 5, x + 18, y + 6);

 // The wheels
 setfillstyle(1, BLUE);
 bar(x + 1, y + 9, x + 3, y + 15);
 bar(x + 22, y + 9, x + 24, y + 15);
 bar(x + 1, y + 26, x + 3, y + 32);
 bar(x + 22, y + 26, x + 24, y + 32);
}

// initialize the tracktor type 1
void init_tracktor01_b(int x, int y)
{
 int i;
 setlinestyle(0, 0, 1);
 setcolor(DARKGRAY);
 // The main rectangle
 setfillstyle(1, RED);
 bar(x + 3, y, x + 22, y + 41);
 rectangle(x + 3, y, x + 22, y + 41);
 setlinestyle(0, 0, 3);
 rectangle(x + 6, y + 3, x + 19, y + 38);

 // The samll bars
 setcolor(DARKGRAY);
 setlinestyle(0, 0, 1);
 setfillstyle(1, YELLOW);
 bar(x, y + 38, x + 25, y + 41);
 rectangle(x, y + 38, x + 25, y + 41);
 for (i = x; i <= 25 + x; i += 2)
 {
        bar(i - 1, y + 41, i + 1, y + 37);
        rectangle(i - 1, y + 42, i + 1, y + 37);
 }

 // The driver site
 setfillstyle(1, RED);
 bar(x + 7, y + 43, x + 18, y + 32);
 rectangle(x + 7, y + 43, x + 18, y + 32);

 // The wheels
 setfillstyle(1, BLUE);
 bar(x + 1, y + 29, x + 3, y + 23);
 bar(x + 22, y + 29, x + 24, y + 23);
 bar(x + 1, y + 12, x + 3, y + 6);
 bar(x + 22, y + 12, x + 24, y + 6);
}

void init_tracktor01_r(int x, int y)
{
 int i;
 setlinestyle(0, 0, 1);
 setcolor(DARKGRAY);
 // The main rectangle
 setfillstyle(1, RED);
 bar(x - 3, y + 3, x - 38, y + 22);
 rectangle(x - 3, y + 3, x - 38, y + 22);
 setlinestyle(0, 0, 3);
 rectangle(x - 6, y + 6, x - 35, y + 19);

 // The samll bars
 setcolor(DARKGRAY);
 setlinestyle(0, 0, 1);
 setfillstyle(1, YELLOW);
 bar(x, y, x - 3, y + 25);
 rectangle(x, y, x - 3, y + 25);
 for (i = y; i <= 25 + y; i += 2)
 {
        bar(x + 1, i - 1, x - 4, i + 1);
        rectangle(x + 1, i - 1, x - 4, i + 1);
 }

 // The driver site
 setfillstyle(1, RED);
 bar(x - 5, y + 7, x + 6, y + 18);
 rectangle(x - 5, y + 7, x + 6, y + 18);

 // The wheels
 setfillstyle(1, BLUE);
 bar(x - 9, y + 1, x - 15, y + 3);
 bar(x - 9, y + 22, x - 15, y + 24);
 bar(x - 26, y + 1, x - 32, y + 3);
 bar(x - 26, y + 22, x - 32, y + 24);
}

void init_tracktor01_l(int x, int y)
{
 int i;
 setlinestyle(0, 0, 1);
 setcolor(DARKGRAY);
 // The main rectangle
 setfillstyle(1, RED);
 bar(x, y + 3, x - 41, y + 22);
 rectangle(x, y + 3, x - 41, y + 22);
 setlinestyle(0, 0, 3);
 rectangle(x - 3, y + 6, x - 38, y + 19);

 // The samll bars
 setcolor(DARKGRAY);
 setlinestyle(0, 0, 1);
 setfillstyle(1, YELLOW);
 bar(x - 38, y, x - 41, y + 25);
 rectangle(x - 38, y, x - 41, y + 25);
 for (i = y; i <= 25 + y; i += 2)
 {
        bar(x - 41, i - 1, x - 37, i + 1);
        rectangle(x - 42, i - 1, x - 37, i + 1);
 }

 // The driver site
 setfillstyle(1, RED);
 bar(x - 43, y + 7, x - 32, y + 18);
 rectangle(x - 43, y + 7, x - 32, y + 18);

 // The wheels
 setfillstyle(1, BLUE);
 bar(x - 29, y + 1, x - 23, y + 3);
 bar(x - 29, y + 22, x - 23, y + 24);
 bar(x - 12, y + 1, x - 6, y + 3);
 bar(x - 12, y + 22, x - 6, y + 24);
}

// initialize the tarcker 02 in front
void init_tracktor02_f(int x, int y)
{
 int i;
 setlinestyle(0, 0, 1);
 setcolor(DARKGRAY);
 // The main rectangle
 setfillstyle(1, GREEN);
 bar(x + 3, y - 3, x + 22, y + 38);
 rectangle(x + 3, y - 3, x + 22, y + 38);
 setlinestyle(0, 0, 3);
 rectangle(x + 6, y - 6, x + 19, y + 35);

 // The samll bars
 setcolor(DARKGRAY);
 setlinestyle(0, 0, 1);
 setfillstyle(1, YELLOW);
 bar(x, y, x + 25, y - 3);
 rectangle(x, y, x + 25, y + 3);
 for (i = x; i <= 25 + x; i += 2)
 {
        bar(i - 1, y + 1, i + 1, y - 4);
        rectangle(i - 1, y + 1, i + 1, y - 4);
 }

 // The driver site
 setfillstyle(1, GREEN);
 bar(x + 7, y - 5, x + 18, y + 6);
 rectangle(x + 7, y - 5, x + 18, y + 6);

 // The wheels
 setfillstyle(1, BLUE);
 bar(x + 1, y + 9, x + 3, y + 15);
 bar(x + 22, y + 9, x + 24, y + 15);
 bar(x + 1, y + 26, x + 3, y + 32);
 bar(x + 22, y + 26, x + 24, y + 32);
}

// initialize the tracktor type 2 in backward
void init_tracktor02_b(int x, int y)
{
 int i;
 setlinestyle(0, 0, 1);
 setcolor(DARKGRAY);
 // The main rectangle
 setfillstyle(1, GREEN);
 bar(x + 3, y, x + 22, y + 41);
 rectangle(x + 3, y, x + 22, y + 41);
 setlinestyle(0, 0, 3);
 rectangle(x + 6, y + 3, x + 19, y + 38);

 // The samll bars
 setcolor(DARKGRAY);
 setlinestyle(0, 0, 1);
 setfillstyle(1, YELLOW);
 bar(x, y + 38, x + 25, y + 41);
 rectangle(x, y + 38, x + 25, y + 41);
 for (i = x; i <= 25 + x; i += 2)
 {
        bar(i - 1, y + 41, i + 1, y + 37);
        rectangle(i - 1, y + 42, i + 1, y + 37);
 }

 // The driver site
 setfillstyle(1, GREEN);
 bar(x + 7, y + 43, x + 18, y + 32);
 rectangle(x + 7, y + 43, x + 18, y + 32);

 // The wheels
 setfillstyle(1, BLUE);
 bar(x + 1, y + 29, x + 3, y + 23);
 bar(x + 22, y + 29, x + 24, y + 23);
 bar(x + 1, y + 12, x + 3, y + 6);
 bar(x + 22, y + 12, x + 24, y + 6);
}

void init_tracktor02_r(int x, int y)
{
 int i;
 setlinestyle(0, 0, 1);
 setcolor(DARKGRAY);
 // The main rectangle
 setfillstyle(1, GREEN);
 bar(x - 3, y + 3, x - 38, y + 22);
 rectangle(x - 3, y + 3, x - 38, y + 22);
 setlinestyle(0, 0, 3);
 rectangle(x - 6, y + 6, x - 35, y + 19);

 // The samll bars
 setcolor(DARKGRAY);
 setlinestyle(0, 0, 1);
 setfillstyle(1, YELLOW);
 bar(x, y, x - 3, y + 25);
 rectangle(x, y, x - 3, y + 25);
 for (i = y; i <= 25 + y; i += 2)
 {
        bar(x + 1, i - 1, x - 4, i + 1);
        rectangle(x + 1, i - 1, x - 4, i + 1);
 }

 // The driver site
 setfillstyle(1, GREEN);
 bar(x - 5, y + 7, x + 6, y + 18);
 rectangle(x - 5, y + 7, x + 6, y + 18);

 // The wheels
 setfillstyle(1, BLUE);
 bar(x - 9, y + 1, x - 15, y + 3);
 bar(x - 9, y + 22, x - 15, y + 24);
 bar(x - 26, y + 1, x - 32, y + 3);
 bar(x - 26, y + 22, x - 32, y + 24);
}

void init_tracktor02_l(int x, int y)
{
 int i;
 setlinestyle(0, 0, 1);
 setcolor(DARKGRAY);
 // The main rectangle
 setfillstyle(1, GREEN);
 bar(x, y + 3, x - 41, y + 22);
 rectangle(x, y + 3, x - 41, y + 22);
 setlinestyle(0, 0, 3);
 rectangle(x - 3, y + 6, x - 38, y + 19);

 // The samll bars
 setcolor(DARKGRAY);
 setlinestyle(0, 0, 1);
 setfillstyle(1, YELLOW);
 bar(x - 38, y, x - 41, y + 25);
 rectangle(x - 38, y, x - 41, y + 25);
 for (i = y; i <= 25 + y; i += 2)
 {
        bar(x - 41, i - 1, x - 37, i + 1);
        rectangle(x - 42, i - 1, x - 37, i + 1);
 }

 // The driver site
 setfillstyle(1, GREEN);
 bar(x - 43, y + 7, x - 32, y + 18);
 rectangle(x - 43, y + 7, x - 32, y + 18);

 // The wheels
 setfillstyle(1, BLUE);
 bar(x - 29, y + 1, x - 23, y + 3);
 bar(x - 29, y + 22, x - 23, y + 24);
 bar(x - 12, y + 1, x - 6, y + 3);
 bar(x - 12, y + 22, x - 6, y + 24);
```

```
}

// earth filling after pick ,front
void earth_fill01(int x, int y)
{
 setfillstyle(1, BROWN);
 setcolor(WHITE);
 bar(x - 1, y, x + 25, y + 41);
 // if (y % 4 == 0)
 // {
 //     int x_temp = rand() % 25;
 //     line(x + x_temp, y + 41, x + x_temp, y + 41);
 // }
}

// earth filling after pick,back
void earth_fill02(int x, int y)
{
 setfillstyle(1, BROWN);
 setcolor(WHITE);
 bar(x - 1, y - 1, x + 25, y + 41);
 // if (y % 4 == 0)
 // {
 //     int x_temp = rand() % 25;
 //     line(x + x_temp, y - 1, x + x_temp, y - 1);
 // }
}

// earth filling after pick, turn direction
void earth_fill03(int x, int y)
{
 // int i;
 setfillstyle(1, BROWN);
 setcolor(WHITE);
 bar(x - 1, y, x + 25, y + 50);
 // for (i = 0; i < 10; i++)
 // {
 //  int x_temp = rand() % 25, y_temp = rand() %
50;
 //     line(x_temp + x, y_temp + y, x_temp + x,
y_temp + y);
 // }
}

// earth filling    ,front and back
void earth_cover01(int x, int y)
{
 setfillstyle(1, BROWN);
 setcolor(WHITE);
 bar(x - 1, y, x + 26, y + 41);
 // if(y%4==0)
 // {
 //     int x_temp=rand()%25;
 //     line(x+x_temp,y+41,x+x_temp,y+41);
 // }
}

// earth filling ,left and right
void earth_cover02(int x, int y)
{
 setfillstyle(1, BROWN);
 setcolor(WHITE);
 bar(x - 1, y - 1, x - 41, y + 26);
 // if(y%4==0)
 // {
 //     int x_temp=rand()%25;
 //     line(x+x_temp,y-1,x+x_temp,y-1);
 // }
}

// earth filling , turn direction
void earth_cover03(int x, int y)
{
 int i;
 setfillstyle(1, BROWN);
 setcolor(WHITE);
 bar(x - 1, y, x + 25, y + 50);
 // for(i=0;i<10;i++)
 // {
 //     int x_temp=rand()%25,y_temp=rand()%50;
 //
line(x_temp+x,y_temp+y,x_temp+x,y_temp+y);
 // }
}


// initialize the picker 01
void init_picker_f(int x,int y)
{
 int i;
 setlinestyle(0, 0, 1);
 setcolor(DARKGRAY);
 // The main rectangle
 setfillstyle(1, LIGHTGRAY);
 bar(x + 3, y - 3, x + 22, y + 38);
 rectangle(x + 3, y - 3, x + 22, y + 38);
 setlinestyle(0, 0, 3);
 rectangle(x + 6, y - 6, x + 19, y + 35);

 // The samll bars
 setcolor(DARKGRAY);
 setlinestyle(0, 0, 1);
 setfillstyle(1, CYAN);
 bar(x, y, x + 25, y - 3);
 rectangle(x, y, x + 25, y + 3);
 for (i = x; i <= 25 + x; i += 5)
 {
        bar(i - 1, y + 1, i + 1, y - 4);
        rectangle(i - 1, y + 1, i + 1, y - 4);
 }

 // The driver site
 setfillstyle(1, LIGHTBLUE);
 bar(x + 7, y - 5, x + 18, y + 6);
 rectangle(x + 7, y - 5, x + 18, y + 6);

 // The wheels
 setfillstyle(1, BLUE);
 bar(x + 1, y + 9, x + 3, y + 15);
 bar(x + 22, y + 9, x + 24, y + 15);
 bar(x + 1, y + 26, x + 3, y + 32);
 bar(x + 22, y + 26, x + 24, y + 32);
}

void    init_picker_b(int x, int y)
{
 int i;
 setlinestyle(0, 0, 1);
 setcolor(DARKGRAY);
 // The main rectangle
 setfillstyle(1, LIGHTGRAY);
 bar(x + 3, y, x + 22, y + 41);
 rectangle(x + 3, y, x + 22, y + 41);
 setlinestyle(0, 0, 3);
 rectangle(x + 6, y + 3, x + 19, y + 38);

 // The samll bars
 setcolor(DARKGRAY);
 setlinestyle(0, 0, 1);
 setfillstyle(1, CYAN);
 bar(x, y + 38, x + 25, y + 41);
 rectangle(x, y + 38, x + 25, y + 41);
 for (i = x; i <= 25 + x; i += 5)
 {
        bar(i - 1, y + 41, i + 1, y + 37);
        rectangle(i - 1, y + 42, i + 1, y + 37);
 }

 // The driver site
 setfillstyle(1, LIGHTBLUE);
 bar(x + 7, y + 43, x + 18, y + 32);
 rectangle(x + 7, y + 43, x + 18, y + 32);

 // The wheels
 setfillstyle(1, BLUE);
 bar(x + 1, y + 29, x + 3, y + 23);
 bar(x + 22, y + 29, x + 24, y + 23);
 bar(x + 1, y + 12, x + 3, y + 6);
 bar(x + 22, y + 12, x + 24, y + 6);
}

void init_picker_r(int x, int y)
{
 int i;
 setlinestyle(0, 0, 1);
 setcolor(DARKGRAY);
 // The main rectangle
 setfillstyle(1, LIGHTGRAY);
 bar(x - 3, y + 3, x - 38, y + 22);
 rectangle(x - 3, y + 3, x - 38, y + 22);
 setlinestyle(0, 0, 3);
 rectangle(x - 6, y + 6, x - 35, y + 19);

 // The samll bars
 setcolor(DARKGRAY);
 setlinestyle(0, 0, 1);
 setfillstyle(1, CYAN);
 bar(x, y, x - 3, y + 25);
 rectangle(x, y, x - 3, y + 25);
 for (i = y; i <= 25 + y; i += 5)
 {
        bar(x + 1, i - 1, x - 4, i + 1);
        rectangle(x + 1, i - 1, x - 4, i + 1);
 }

 // The driver site
 setfillstyle(1, LIGHTBLUE);
 bar(x - 5, y + 7, x + 6, y + 18);
 rectangle(x - 5, y + 7, x + 6, y + 18);

 // The wheels
 setfillstyle(1, BLUE);
 bar(x - 9, y + 1, x - 15, y + 3);
 bar(x - 9, y + 22, x - 15, y + 24);
 bar(x - 26, y + 1, x - 32, y + 3);
 bar(x - 26, y + 22, x - 32, y + 24);
}

void init_picker_l(int x, int y)
{
 int i;
 setlinestyle(0, 0, 1);
 setcolor(DARKGRAY);
 // The main rectangle
 setfillstyle(1, LIGHTGRAY);
 bar(x, y + 3, x - 41, y + 22);
 rectangle(x, y + 3, x - 41, y + 22);
 setlinestyle(0, 0, 3);
 rectangle(x - 3, y + 6, x - 38, y + 19);

 // The samll bars
 setcolor(DARKGRAY);
 setlinestyle(0, 0, 1);
 setfillstyle(1, CYAN);
 bar(x - 38, y, x - 41, y + 25);
 rectangle(x - 38, y, x - 41, y + 25);
 for (i = y; i <= 25 + y; i += 5)
 {
        bar(x - 41, i - 1, x - 37, i + 1);
        rectangle(x - 42, i - 1, x - 37, i + 1);
 }

 // The driver site
 setfillstyle(1, LIGHTBLUE);
 bar(x - 43, y + 7, x - 32, y + 18);
 rectangle(x - 43, y + 7, x - 32, y + 18);

 // The wheels
 setfillstyle(1, BLUE);
 bar(x - 29, y + 1, x - 23, y + 3);
 bar(x - 29, y + 22, x - 23, y + 24);
 bar(x - 12, y + 1, x - 6, y + 3);
 bar(x - 12, y + 22, x - 6, y + 24);
}

// initialize the picker after pick
void init_picker01_f(int x,int y)
{
 int i;
 setlinestyle(0, 0, 1);
 setcolor(DARKGRAY);
 //The cotton package
 setfillstyle(1,WHITE);
 bar(x,y-1,x+tracktor_w,y-1+co_pak_w);

 // The main rectangle
 setfillstyle(1, LIGHTGRAY);
 bar(x + 3, y - 3, x + 22, y + 38);
 rectangle(x + 3, y - 3, x + 22, y + 38);
 setlinestyle(0, 0, 3);
 rectangle(x + 6, y - 6, x + 19, y + 35);

 // The samll bars
 setcolor(DARKGRAY);
 setlinestyle(0, 0, 1);
 setfillstyle(1, CYAN);
 bar(x, y, x + 25, y - 3);
 rectangle(x, y, x + 25, y + 3);
 for (i = x; i <= 25 + x; i += 5)
 {
        bar(i - 1, y + 1, i + 1, y - 4);
        rectangle(i - 1, y + 1, i + 1, y - 4);
 }

 // The driver site
 setfillstyle(1, LIGHTBLUE);
 bar(x + 7, y - 5, x + 18, y + 6);
 rectangle(x + 7, y - 5, x + 18, y + 6);

 // The wheels
 setfillstyle(1, BLUE);
 bar(x + 1, y + 9, x + 3, y + 15);
 bar(x + 22, y + 9, x + 24, y + 15);
 bar(x + 1, y + 26, x + 3, y + 32);
 bar(x + 22, y + 26, x + 24, y + 32);
}

void    init_picker01_b(int x, int y)
{
 int i;
 setlinestyle(0, 0, 1);
 setcolor(DARKGRAY);
 //The cotton package
 setfillstyle(1,WHITE);
 bar(x,y+42,x+tracktor_w,y+42-co_pak_w);

 // The main rectangle
 setfillstyle(1, LIGHTGRAY);
 bar(x + 3, y, x + 22, y + 41);
 rectangle(x + 3, y, x + 22, y + 41);
 setlinestyle(0, 0, 3);
 rectangle(x + 6, y + 3, x + 19, y + 38);

 // The samll bars
 setcolor(DARKGRAY);
 setlinestyle(0, 0, 1);
 setfillstyle(1, CYAN);
 bar(x, y + 38, x + 25, y + 41);
 rectangle(x, y + 38, x + 25, y + 41);
 for (i = x; i <= 25 + x; i += 5)
 {
        bar(i - 1, y + 41, i + 1, y + 37);
        rectangle(i - 1, y + 42, i + 1, y + 37);
 }

 // The driver site
 setfillstyle(1, LIGHTBLUE);
 bar(x + 7, y + 43, x + 18, y + 32);
 rectangle(x + 7, y + 43, x + 18, y + 32);

 // The wheels
 setfillstyle(1, BLUE);
 bar(x + 1, y + 29, x + 3, y + 23);
 bar(x + 22, y + 29, x + 24, y + 23);
 bar(x + 1, y + 12, x + 3, y + 6);
 bar(x + 22, y + 12, x + 24, y + 6);
}

void init_picker01_r(int x, int y)
{
 int i;
 setlinestyle(0, 0, 1);
 setcolor(DARKGRAY);
 //The cotton package
 setfillstyle(1,WHITE);
 bar(x+1,y,x+1-co_pak_w,y+tracktor_w);

 // The main rectangle
 setfillstyle(1, LIGHTGRAY);
 bar(x - 3, y + 3, x - 38, y + 22);
 rectangle(x - 3, y + 3, x - 38, y + 22);
 setlinestyle(0, 0, 3);
 rectangle(x - 6, y + 6, x - 35, y + 19);

 // The samll bars
 setcolor(DARKGRAY);
 setlinestyle(0, 0, 1);
 setfillstyle(1, CYAN);
 bar(x, y, x - 3, y + 25);
 rectangle(x, y, x - 3, y + 25);
 for (i = y; i <= 25 + y; i += 5)
 {
        bar(x + 1, i - 1, x - 4, i + 1);
        rectangle(x + 1, i - 1, x - 4, i + 1);
 }

 // The driver site
 setfillstyle(1, LIGHTBLUE);
 bar(x - 5, y + 7, x + 6, y + 18);
 rectangle(x - 5, y + 7, x + 6, y + 18);

 // The wheels
 setfillstyle(1, BLUE);
 bar(x - 9, y + 1, x - 15, y + 3);
 bar(x - 9, y + 22, x - 15, y + 24);
 bar(x - 26, y + 1, x - 32, y + 3);
 bar(x - 26, y + 22, x - 32, y + 24);
}

void init_picker01_l(int x, int y)
{
 int i;
 setlinestyle(0, 0, 1);
 setcolor(DARKGRAY);
 //The cotton package
 setfillstyle(1,WHITE);
 bar(x-42,y,x-42+co_pak_w,y+tracktor_w);

 // The main rectangle
 setfillstyle(1, LIGHTGRAY);
 bar(x, y + 3, x - 41, y + 22);
 rectangle(x, y + 3, x - 41, y + 22);
 setlinestyle(0, 0, 3);
 rectangle(x - 3, y + 6, x - 38, y + 19);

 // The samll bars
 setcolor(DARKGRAY);
 setlinestyle(0, 0, 1);
 setfillstyle(1, CYAN);
 bar(x - 38, y, x - 41, y + 25);
 rectangle(x - 38, y, x - 41, y + 25);
 for (i = y; i <= 25 + y; i += 5)
 {
     bar(x - 41, i - 1, x - 37, i + 1);
     rectangle(x - 42, i - 1, x - 37, i + 1);
 }

 // The driver site
 setfillstyle(1, LIGHTBLUE);
 bar(x - 43, y + 7, x - 32, y + 18);
 rectangle(x - 43, y + 7, x - 32, y + 18);

 // The wheels
 setfillstyle(1, BLUE);
 bar(x - 29, y + 1, x - 23, y + 3);
 bar(x - 29, y + 22, x - 23, y + 24);
 bar(x - 12, y + 1, x - 6, y + 3);
 bar(x - 12, y + 22, x - 6, y + 24);
}
```

2、 EDIT.C

```
#include "EDIT.H"
#include "COMMON.H"
#include "IMAGE.h"
#include "PARAMETE.H"
#include "LOGFUN.H"
#include "START.H"

void edit01_screen()
{
  clrmous(MouseX, MouseY);
  cleardevice();
  setbkcolor(WHITE);
  bmp_convert(".\\photo\\map.bmp",
".\\photo\\map.dbm");
  show_dbm(5, 100, ".\\photo\\map.dbm");
  puthz(240, 30, "请选择地区", 32, 32,
BLUE);
  quit();
  last();
  setcolor(12); // 淡红色

  // 参数名字
  puthz(50, 90, "参数名字：", 32, 32, BLUE);

  // 按钮
  puthz(80 - 30, 150, "新疆地区", 32, 32,
BLUE);
  puthz(400 - 40, 150 + 30, "黄河地区", 32,
32, BLUE);
  puthz(250 - 20, 300 + 20, "长江地区", 32,
32, BLUE);
  setcolor(CYAN);
  bar(120 - 30, 200, 150 - 30, 230);
  bar(440 - 40, 200 + 30, 470 - 40, 230 + 30);
  bar(290 - 20, 350 + 20, 320 - 20, 380 + 20);
  rectangle(230, 80, 520, 130);
}

int edit01(struct Parameter *abc)
{
  INPUT name = {230, 80, 520, 130, "", 10, 0,
0};
  edit01_screen();
  setfillstyle(1, MAGENTA); // 洋红色

  for (;;)
  {
    newmouse(&MouseX, &MouseY,
&press);
    input_s(233, 80, &name, 16, 0);
    if (mouse_press(100 - 30, 180, 170 - 30,
250) == 2) // 新疆
    {
        setfillstyle(1, MAGENTA);
        bar(120 - 30, 200, 150 - 30, 230);
    }
    else
    {
        setfillstyle(1, CYAN);
        bar(120 - 30, 200, 150 - 30, 230);
    }
    if (mouse_press(100 - 30, 180, 170 - 30,
250) == 1)
    {
        if (judgename(name.string) == 1)
        {
            strcpy(abc->name,
name.string);
            abc->place = 'a';
```

```
                return 0;
            }
        }

        if (mouse_press(420 - 40, 180 + 30,
490 - 40, 250 + 30) == 2) // 黄河
        {
            setfillstyle(1, MAGENTA);
            bar(440 - 40, 200 + 30, 470 - 40,
230 + 30);
        }
        else
        {
            setfillstyle(1, CYAN);
            bar(440 - 40, 200 + 30, 470 - 40,
230 + 30);
        }
        if (mouse_press(420 - 40, 180 + 30,
490 - 40, 250 + 30) == 1)
        {
            if (judgename(name.string) == 1)
            {
                strcpy(abc->name,
name.string);
                abc->place = 'b';
                return 0;
            }
        }

        if (mouse_press(270 - 20, 330 + 20,
340 - 20, 400 + 20) == 2) // 长江
        {
            setfillstyle(1, MAGENTA);
            bar(290 - 20, 350 + 20, 320 - 20,
380 + 20);
        }
        else
        {
            setfillstyle(1, CYAN);
            bar(290 - 20, 350 + 20, 320 - 20,
380 + 20);
        }
        if (mouse_press(270 - 20, 330 + 20,
340 - 20, 400 + 20) == 1)
        {
            if (judgename(name.string) == 1)
            {
                strcpy(abc->name,
name.string);
                abc->place = 'c';
                return 0;
            }
        }

        // quit
        if (mouse_press(0, 0, 40, 30) == 0 ||
mouse_press(0, 450, 40, 480) == 0)
        {
            MouseS = 0;
        }
        if (mouse_press(0, 0, 40, 30) == 2 ||
mouse_press(0, 450, 40, 480) == 2)
        {
            MouseS = 1;
        }
        if (mouse_press(0, 0, 40, 30) == 1)
        {
            exit(0);
        }

        // last
        if (mouse_press(0, 450, 40, 480) == 1)
        {
            return 1;
        }
        delay(15);
    }
    // getchar();
    // closegraph();
}

void edit02_screen(struct Parameter *abc)
{
    clrmous(MouseX, MouseY);
    cleardevice();
    setbkcolor(WHITE);

    quit();
    last();

    puthz(70, 40, "请输入土地面积：", 32, 32,
BLUE);
    puthz(490, 40, "公顷", 32, 32, BLUE);
    puthz(70, 100, "本地推荐种植的棉花种类
为：", 32, 32, BLUE);

        if (abc->place == 'a')
        {
            puthz(480, 100, "粗绒棉", 32, 32,
BLUE);
        }
        else if (abc->place == 'b')
        {
            puthz(480, 100, "长绒棉", 32, 32,
BLUE);
        }
        else
        {
            puthz(480, 100, "细绒棉", 32, 32,
BLUE);
        }

        // 土地形状
        setfillstyle(1, LIGHTGRAY);
        bar(0, 150, 160, 200);
        bar(0, 230, 160, 280);
        bar(0, 310, 160, 360);
        bar(0, 380, 160, 430);
        puthz(20, 160, "矩形", 32, 32, BLUE);
        puthz(20, 235, "圆形", 32, 32, BLUE);
        puthz(10, 315, "多边形", 32, 32, BLUE);
        puthz(0, 390, "自定义形状", 32, 32, BLUE);

        setfillstyle(1, WHITE);
        bar(200, 150, 600, 460);
        setcolor(RED);
        rectangle(200, 150, 600, 460);
        rectangle(330, 30, 490, 90);
    }

    int edit02(struct Parameter *abc)
    {
        // int flag = 0; // 返回键判断
        int flagcan = 0, flagcan1 = 1;
        INPUT S = {330, 30, 490, 90, "", 6, 0, 0};

        edit02_screen(abc);

        for (;;)
        {
            newmouse(&MouseX,      &MouseY,
&press);
            delay(15);
            input_s(333, 30, &S, 16, 0);

            // 土地形状按钮，停留在上面
            if (mouse_press(0, 150, 160, 200) == 2)
// 矩形
            {
                if (flagcan1 == 1)
                {
                    clrmous(MouseX, MouseY);
                    MouseS = 1;
                    setfillstyle(1, MAGENTA);
                    bar(0, 150, 160, 200);
                   puthz(20, 160, "矩形", 32, 32,
BLUE);
                    setfillstyle(10, BROWN);
                    rectangle(240,   190,   560,
420);
                    bar(241, 191, 559, 419);

                    flagcan = 0;
                    flagcan1 = 0;
                }
            }
            else if (mouse_press(0, 230, 160, 280)
== 2) // 圆形
            {
                if (flagcan1 == 1)
                {
                    clrmous(MouseX, MouseY);
                    MouseS = 1;
                    setfillstyle(1, MAGENTA);
                    bar(0, 230, 160, 280);
                   puthz(20, 235, "圆形", 32, 32,
BLUE);
                    setfillstyle(10, LIGHTGRAY);
                    circle(400, 305, 121);
                    pieslice(400,   305,   0,   360,
120);
                    line(280, 305, 520, 305);
                    line(400, 185, 400, 425);

                    flagcan = 0;
                    flagcan1 = 0;
                }
            }
```

```
		else if (mouse_press(0, 310, 160, 360) == 2) // 多边形
		{
			if (flagcan1 == 1)
			{
				int dindian[8] = {220, 270, 350, 270, 285, 170, 220, 270}, dindian2[10] = {560, 420, 560, 330, 400, 330, 400, 400, 560, 420}; // 200,150,600,460
				clrmous(MouseX, MouseY);
				MouseS = 1;
				setfillstyle(1, MAGENTA);
				bar(0, 310, 160, 360);
				puthz(10, 315, "多边形", 32, 32, BLUE);
				setfillstyle(10, LIGHTGRAY);
				fillpoly(4, dindian);
				fillpoly(5, dindian2);
				setlinestyle(0, 0, 3);
				line(220, 440, 580, 170);
				flagcan = 0;
				flagcan1 = 0;
			}
		}
		else if (mouse_press(0, 380, 160, 430) == 2) // 自定义图形
		{
			if (flagcan1 == 1)
			{
				clrmous(MouseX, MouseY);
				MouseS = 1;
				setlinestyle(0, 0, 15);
				setfillstyle(1, MAGENTA);
				bar(0, 380, 160, 430);
				puthz(0, 390, "自定义图形", 32, 32, BLUE);
				setfillstyle(1, LIGHTGRAY);
				arc(400, 230, -90, 180, 60);
				line(400, 290, 400, 370);
				setfillstyle(1, BLACK);
				circle(400, 390, 10);

				flagcan = 0;
				flagcan1 = 0;
			}
		}
		else
		{
			if (flagcan == 0)
			{
				clrmous(MouseX, MouseY);
				MouseS = 0;
				setfillstyle(1, LIGHTGRAY);
				bar(0, 150, 160, 200);
				puthz(20, 160, "矩形", 32, 32, BLUE);
				bar(0, 380, 160, 430);
				puthz(0, 390, "自定义图形", 32, 32, BLUE);
				bar(0, 310, 160, 360);
				puthz(10, 315, "多边形", 32, 32, BLUE);
				bar(0, 230, 160, 280);
				puthz(20, 235, "圆形", 32, 32, BLUE);
				setfillstyle(1, WHITE);
				bar(200, 150, 600, 460);
				setcolor(RED);
				rectangle(200, 150, 600, 460);
				flagcan = 1;
				flagcan1 = 1;
			}
		}

		if (mouse_press(0, 150, 160, 200) == 1) // 矩形
		{
			if (judgeS(S.string) == 1)
			{
				strcpy(abc->S, S.string);
				abc->shape = 'a';
				return 0;
			}
		}
		else if (mouse_press(0, 230, 160, 280) == 1) // 圆形
		{
			if (judgeS(S.string) == 1)
			{
				strcpy(abc->S, S.string);
				abc->shape = 'b';
				return 0;
			}
```

```
            }
            else if (mouse_press(0, 300, 160, 350) == 1) // 多边形
            {
                if (judgeS(S.string) == 1)
                {
                    strcpy(abc->S, S.string);
                    abc->shape = 'c';
                    select02(abc);
                    return 0;
                }
            }
            else if (mouse_press(0, 370, 160, 420) == 1) // 自定义图形
            {
                if (judgeS(S.string) == 1)
                {
                    strcpy(abc->S, S.string);
                    abc->shape = 'd';
                    select03(abc);
                    return 0;
                }
            }

            // quit
            if (mouse_press(0, 0, 40, 30) == 0 || mouse_press(0, 450, 40, 480) == 0)
            {
                MouseS = 0;
            }
            if (mouse_press(0, 0, 40, 30) == 2 || mouse_press(0, 450, 40, 480) == 2)
            {
                MouseS = 1;
            }
            if (mouse_press(0, 0, 40, 30) == 1)
            {
                exit(0);
            }

            // last
            if (mouse_press(0, 450, 40, 480) == 1)
            {
                return 1;
            }
        }
    }

void edit03_screen()
{
    int dindian0[10] = {100, 150, 100, 200, 250, 200, 250, 150, 100, 150};
    clrmous(MouseX, MouseY);
    cleardevice();
    setbkcolor(WHITE);

    quit();
    last();

    puthz(160, 30, "请选择收割机类型", 32, 32, BLUE);
    puthz(140, 70, "（自动计算所需数量）", 32, 32, BLUE);

    setfillstyle(1, LIGHTGRAY);
    bar(70, 120, 570, 420);

    setfillstyle(1, RED);
    bar(100, 150, 250, 350);
    setcolor(0); // 白色
    setlinestyle(0, 0, 3);
    drawpoly(5, dindian0);
    rectangle(100, 240, 250, 275);
    rectangle(100, 275, 250, 300);
    setlinestyle(0, 0, 5);
    rectangle(100, 300, 250, 325);
    rectangle(100, 325, 250, 340);
    setlinestyle(0, 0, 2);
    rectangle(150, 340, 200, 350);
    bar(150, 350, 200, 380);
    setfillstyle(1, YELLOW);
    bar(100, 340, 150, 350);
    bar(200, 340, 250, 350);
    setfillstyle(1, BLUE);
    bar(90, 170, 100, 220);
    bar(90 + 160, 170, 100 + 160, 220);
    bar(90, 250, 100, 300);
    bar(90 + 160, 250, 100 + 160, 300);

    setfillstyle(1, GREEN);
    bar(100 + 290, 150, 250 + 290, 350);
    setcolor(0); // 白色
    setlinestyle(0, 0, 3);
    rectangle(100 + 290, 150, 250 + 290, 200);
```

```
        rectangle(100 + 290, 240, 250 + 290, 275);
        rectangle(100 + 290, 275, 250 + 290, 300);
        setlinestyle(0, 0, 5);
        rectangle(100 + 290, 300, 250 + 290, 325);
        rectangle(100 + 290, 325, 250 + 290, 340);
        setlinestyle(0, 0, 2);
        rectangle(150 + 290, 340, 200 + 290, 350);
        setfillstyle(1, YELLOW);
        bar(100 + 290 - 25, 340, 150 + 290, 350);
        bar(200 + 290, 340, 250 + 290 + 25, 350);
        setfillstyle(1, BLUE);
        bar(90 + 290, 170, 100 + 290, 220);
        bar(90 + 160 + 290, 170, 100 + 160 + 290,
220);
        bar(90 + 290, 250, 100 + 290, 300);
        bar(90 + 160 + 290, 250, 100 + 160 + 290,
300);

        puthz(110, 130, "垂直式收割机", 16, 16,
BLUE);
        puthz(400, 130, "水平式收割机", 16, 16,
BLUE);
    }

    int edit03(struct Parameter *abc)
    {
        int flag = 0;

        edit03_screen();

        for (;;)
        {
            newmouse(&MouseX,        &MouseY,
&press);
            delay(15);

            // 收割机按钮 70, 120, 570, 420
            if (mouse_press(70, 120, 300, 420) ==
2)
            {
                if (flag == 0)
                {
                    MouseS = 1;
                    setcolor(RED);
                    setlinestyle(0, 0, 5);
                    rectangle(70, 120, 300, 420);
                    flag = 1;
                }
            }
            else if (mouse_press(70, 120, 300, 420)
== 1)
            {
                abc->type = 'a';
                return 0;
            }
            else  if  (mouse_press(340,  120,  570,
420) == 2)
            {
                if (flag == 0)
                {
                    MouseS = 1;
                    setcolor(RED);
                    setlinestyle(0, 0, 5);
                    rectangle(340,    120,    570,
420);

                    flag = 1;
                }
            }
            else  if  (mouse_press(340,  120,  570,
420) == 1)
            {
                abc->type = 'b';
                return 0;
            }
            else
            {
                if (flag == 1)
                {
                    MouseS = 0;
                    setlinestyle(0, 0, 5);
                    setcolor(LIGHTGRAY);
                    rectangle(340,    120,    570,
420);
                    rectangle(70, 120, 300, 420);
                    flag = 0;
                }
            }

            // quit
            if (mouse_press(0, 0, 40, 30) == 0)
            {
                MouseS = 0;
        }
        if (mouse_press(0, 0, 40, 30) == 2)
        {
            MouseS = 1;
        }
        if (mouse_press(0, 0, 40, 30) == 1)
        {
            exit(0);
        }

        // last
        if (mouse_press(0, 450, 40, 480) == 0)
        {
            MouseS = 0;
        }
        if (mouse_press(0, 450, 40, 480) == 2)
        {
            MouseS = 1;
        }
        if (mouse_press(0, 450, 40, 480) == 1)
        {
            return 1;
        }
    }
}
// 产区，面积，形状（坐标），收割机类型，名字
// 产区 place：a--新疆；b--黄河；c--长江；分别对应：粗绒棉，长绒棉，细绒棉
// 面积 S：123（数字）
// 形状 shape：a--矩形；b--圆形；c--多边形；d 自定义形状
// 坐标 xyz：c/d 才用：x1,y1,x2,y2,x3,y3....
// 收割机类型 type：a 垂直/b 水平
// 名字 name：abcdefg

// 土地形状按钮，按下去
void edit()
{
 parameter *abc = (parameter *)malloc(sizeof(parameter));

edit01:
 if (edit01(abc) == 1)
 {
     return;
 }
edit02:
 if (edit02(abc) == 1)
 {
     goto edit01;
 }

 if (edit03(abc) == 1)
 {
     goto edit02;
 }
 wr_parameter(abc);
 free(abc);
 // h->parameter[h->lenpar]=*abc;
}
```

3、 HELP.C

```
#include "COMMON.H"
#include "HELP.H"
#include "LOGFUN.H"

void draw_help()
{
    setfillstyle(1, 0);
    bar(0, 0, 640, 480);
    puthz(240, 30, "帮助及说明", 32, 32, BLUE);
    quit();
    last();
    puthz(30, 90, "本程序作为棉花自动化采摘之模拟程序，可就我国三大棉花产区进行收割模拟，并计算其收成及模拟仓储过程。", 32, 32, BLUE);

    setfillstyle(1, MAGENTA);
    bar(320 + 15 - 5, 300 - 10, 320 + 15 + 120 + 5, 380);
    puthz(320 + 15, 305, "编辑参数", 32, 32, BLUE);
    bar(320 - 15 + 5, 300 - 10, 320 - 15 - 120 - 5, 380);
    puthz(320 - 15 - 120, 305, "开始模拟", 32, 32, BLUE);
    bar(320 + 15 + 150 - 5, 300 - 10, 320 + 15 + 120 + 150 + 5, 380);
    puthz(320 + 15 + 150, 305, "参数列表", 32, 32, BLUE);
    bar(320 - 15 - 150 + 5, 300 - 10, 320 - 15 - 120 - 150 - 5, 380);
    puthz(320 - 15 - 120 - 150, 305, "仓库管理", 32,
32, BLUE);
	}
	void help01()
	{
		int i, flag = 0, flag1 = 1;
		cleardevice();
		setbkcolor(WHITE);
		draw_help();

		setlinestyle(0,0,5);
		for (;;)
		{
			newmouse(&MouseX, &MouseY, &press);
			if (mouse_press(320 + 15 - 5, 300 - 10, 320
+ 15 + 120 + 5, 380) == 2) //  编辑参数
			{
				if (flag1 == 1)
				{
					edit_help();
					flag = 0;
					flag1 = 0;
				}
			}
			else if (mouse_press(320 - 15 - 120 - 5, 300
- 10, 320 - 15 + 5, 380) == 2) //  开始模拟
			{
				if (flag1 == 1)
				{
					start_help();
					flag = 0;
					flag1 = 0;
				}
			}
			else if (mouse_press(320 + 15 + 150 - 5, 300
- 10, 320 + 15 + 120 + 150 + 5, 380) == 2) //  参数列表
			{
				if (flag1 == 1)
				{
					past_help();
					flag = 0;
					flag1 = 0;
				}
			}
			else if (mouse_press(320 - 15 - 120 - 150 - 5,
300 - 10, 320 - 15 - 150 + 5, 380) == 2) //  仓库管理
			{
				if (flag1 == 1)
				{
					ware_help();
					flag = 0;
					flag1 = 0;
				}
			}
			else
			{
				if (flag == 0)
				{
					clrmous(MouseX, MouseY);
					setfillstyle(1, 0);
					bar(15, 85, 640, 240);
					puthz(25, 90, "本程序作为棉花
自动化采摘之模拟程序，可就我国三大棉花产区进行收
割模拟，并计算其收成及模拟仓储过程。", 32, 32, BLUE);
					setfillstyle(1, MAGENTA);
					bar(320 + 15 - 5, 300 - 10, 320 +
15 + 120 + 5, 380);
					puthz(320 + 15, 305, "编辑参数",
32, 32, BLUE);
					bar(320 - 15 + 5, 300 - 10, 320 -
15 - 120 - 5, 380);
					puthz(320 - 15 - 120, 305, "开始
模拟", 32, 32, BLUE);
					bar(320 + 15 + 150 - 5, 300 - 10,
320 + 15 + 120 + 150 + 5, 380);
					puthz(320 + 15 + 150, 305, "参数
列表", 32, 32, BLUE);
					bar(320 - 15 - 150 + 5, 300 - 10,
320 - 15 - 120 - 150 - 5, 380);
					puthz(320 - 15 - 120 - 150, 305, "
仓库管理", 32, 32, BLUE);
					flag = 1;
					flag1 = 1;
				}
			}

			if (mouse_press(0, 0, 40, 30) == 2 ||
mouse_press(0, 450, 40, 480) == 2)
			{
				MouseS = 1;
			}
			if (mouse_press(0, 0, 40, 30) == 1)
			{
				exit(0);
			}
```

```
            // last
            else if (mouse_press(0, 450, 40, 480) == 1)
            {
                return;
            }
            if (mouse_press(0, 0, 40, 30) == 0 || mouse_press(0, 450, 40, 480) == 0)
            {
                MouseS = 0;
            }
            // quit
            delay(15);
        }
    }

    void edit_help(void)
    {
        clrmous(MouseX, MouseY);
        setfillstyle(1, 0);
        bar(15, 85, 640, 240);
        puthz(25, 90, "选择编辑参数按钮即可创建新的参数。参数的数据包括：棉花产区、收割机类型、土地的面积和形状等。", 32, 32, BLUE);
        setcolor(RED);
        rectangle(15, 85, 635, 240);
        setfillstyle(1, RED);
        bar(320 + 15 - 5, 300 - 10, 320 + 15 + 120 + 5, 380);
        puthz(320 + 15, 305, "编辑参数", 32, 32, BLUE);
    }
    void start_help(void)
    {
        clrmous(MouseX, MouseY);
        setfillstyle(1, 0);
        bar(15, 85, 640, 240);
        puthz(25, 90, "点击开始模拟按钮即可选择已经创建的参数进行模拟采摘，将会播放采摘动画并显示采摘所需时间。", 32, 32, BLUE);
        setcolor(LIGHTRED);
        rectangle(15, 85, 635, 240);
        setfillstyle(1, LIGHTRED);
        bar(320 - 15 + 5, 300 - 10, 320 - 15 - 120 - 5, 380);
        puthz(320 - 15 - 120, 305, "开始模拟", 32, 32, BLUE);
    }
    void past_help(void)
    {
        clrmous(MouseX, MouseY);
        setfillstyle(1, 0);
        bar(15, 85, 640, 240);
        puthz(25, 90, "点击参数列表按钮可以查看已经创建了的参数,并对他们进行修改和删除等操作。", 32, 32, BLUE);
        setcolor(CYAN);
        rectangle(15, 85, 635, 240);
        setfillstyle(1, CYAN);
        bar(320 + 15 + 150 - 5, 300 - 10, 320 + 15 + 120 + 150 + 5, 380);
        puthz(320 + 15 + 150, 305, "参数列表", 32, 32, BLUE);
    }
    void ware_help(void)
    {
        clrmous(MouseX, MouseY);
        setfillstyle(1, 0);
        bar(15, 85, 640, 240);
        puthz(25, 90, "点击仓库管理按钮即可对仓库进行管理，可以进行出库和入库等操作", 32, 32, BLUE);
        setcolor(LIGHTGRAY);
        rectangle(15, 85, 635, 240);
        setfillstyle(1, LIGHTGRAY);
        bar(320 - 15 - 150 + 5, 300 - 10, 320 - 15 - 120 - 150 - 5, 380);
        puthz(320 - 15 - 120 - 150, 305, "仓库管理", 32, 32, BLUE);
    }

    void help(void)
    {
        help01();
    }
```

4、 HOME.C

```
#include "COMMON.H"
#include "START.H"
#include "LOGFUN.H"
#include "PARAMETE.H"
#include "HOME.H"
#include "PAST.H"
char str[15];
int k;
```

```
	//	U_ware
here[5]={"ware01",0,{100,200,300},"ware02",1,{1
000,2000,3000},\
	//
"ware03",2,{123,456,789},"ware04",1,{1234,4545
,234},"ware05",0,{34535,3423,6465}};

	//the page of n-w warehouse
	void draw_home01()
	{
	 int i,type,location=0;//1 means the
norwestern,0 means others
	 long int temp, c_tal;
	 clrmous(MouseX,MouseY);
	 for(i=0;i<PAR;i++)
	 {

  if(strcmp(h->parameter[i].name,"\0")==0)
		{
			break;
		}
	 }
	 i--;
	 settextjustify(0,2);
	 switch (h->parameter[i].place)
	 {
		case 'a':
		{
			location=1;
			break;
		}
		case 'b':
		{
			location=0;
			break;
		}
		case 'c':
		{
			location=0;
			break;
		}

		default:
			break;
	 }
	 type=h->here[k].cotton_type,temp=0;
	 c_tal=h->here[k].total[type];
	 if(c_tal>ware_full||c_tal<0)
	 {
		c_tal=ware_full;
	 }
	   //int num;
	 settextstyle(3,0,4);
	   cleardevice();
	   setbkcolor(WHITE);
	 draw_warehouse();
	 draw_trunk();
	 last();
	 if(location==1)
	 {
		puthz(180,30,"新疆地区：室外仓库
",32,32,BLUE);
	 }
	 else
	 {
		puthz(120,30,"黄河、长江流域：室内
仓库",32,32,BLUE);
		setfillstyle(1,DARKGRAY);
		bar(320,90,800,120);
		setfillstyle(1,LIGHTBLUE);
		bar(350,120,360,500);
	 }
	 if(strcmp(str,"\0"))
	 {
		temp=atoi(str);
		if(temp>c_tal)
		{
			temp=c_tal;
		}
		h->here[k].total[type]-=temp;
		for(i=0;i<15;i++)
		{
			str[i]='\0';
		}
	 }
	 in_warehouse(h->here+k);
	 quit();
	 clrmous(MouseX,MouseY);
	 //wr_h();
	 // for(;;)
	 // {
	 //
  newmouse(&MouseX,&MouseY,&press);
	 //	press_home(&(here[k].cotton_type));
```

```
    //    delay(15);
    // }
  }

  //add the press moudule
  void press_home(int *c_t)
  {
    if(mouse_press(0,0,40,30)==0||mouse_pre
ss(53,90,280,190)==0||mouse_press(26,130,46,1
50)==0\
    ||mouse_press(287,130,307,150)==0||mo
use_press(100,300,200,360)==0||mouse_press(0,
450,40,480)==0\
    ||mouse_press(310,150,370,190)==0)
    {
        MouseS=0;
    }
    if(mouse_press(0,0,40,30)==2||mouse_pre
ss(53,90,280,190)==2||mouse_press(26,130,46,1
50)==2\
    ||mouse_press(287,130,307,150)==2||mo
use_press(100,300,200,360)==2||mouse_press(0,
450,40,480)==2\
    ||mouse_press(310,150,370,190)==2)
    {
        MouseS=1;
    }
    if(mouse_press(0,450,40,480)==1)
    {
        mode=0;
        mode1=0;
    }
    if(mouse_press(0,0,40,30)==1)
    {
        // draw_wel();
        wr_h();
        free(h);
        exit(0);
    }
    if(mouse_press(100,300,200,360)==1)
    {
        // warehouse_list(here,5);
        mode1=1;
    }
    if(mouse_press(26,130,46,150)==1)
    {
        (*c_t)--;
        if(*c_t<0)
        {
            *c_t=2;
        }
        // draw_home01();
        mode1=-1;
    }
    if(mouse_press(287,130,307,150)==1)
    {
        (*c_t)++;
        if(*c_t>2)
        {
            *c_t=0;
        }
        // draw_home01();
        mode1=-1;
    }
    if(mouse_press(310,150,370,190)==1)
    {
        changewarename(k+1);
        mode1=-1;
    }
    if(mouse_press(53,90,280,190)==1)
    {
        //
detailed_warehouse(here[k].total[*c_t]);
        mode1=2;
    }
  }

  /*void draw_home00()
  {
      int i;
      //int num;
      cleardevice();
      setbkcolor(WHITE);

    quit();
      mouseinit();
    for(i=0;i<1000;i++)
    {

 newmouse(&MouseX,&MouseY,&press);
        delay(4);
    }
  }*/
```

```
    //draw the board which show the cotton in warehouse
    void in_warehouse(U_ware* now)
    {
      char str1[8];
      int arr1[6]={32-5,140,47-5,132,47-5,148},arr2[6]={301+5,140,286+5,132,286+5,148};
      int type,count;
      type=now->cotton_type,count=now->total[type];
      clrmous(MouseX,MouseY);
      setfillstyle(1,LIGHTGRAY);
      setlinestyle(0,0,1);
      bar(53,70,280,190);
      setfillstyle(1,CYAN);
      bar(310,150,370,190);
      puthz(324,160,"修改",16,16,DARKGRAY);
      setcolor(RED);
      settextstyle(1,0,3);
      outtextxy(110,70,now->ware_name);
      puthz(60,110,"库存量：",16,16,WHITE);
      puthz(200,110,"吨",16,16,WHITE);
      puthz(60,145,"棉花种类：",16,16,WHITE);
      setfillstyle(1,BROWN);
      bar(100,300,200,360);
      puthz(115,320,"仓库列表",16,16,YELLOW);
      switch (type)
      {
      case 0:
          puthz(138,145,"长绒棉",16,16,RED);
          break;
      case 1:
          puthz(138,145,"细绒棉",16,16,RED);
          break;
      case 2:
          puthz(138,145,"粗绒棉",16,16,RED);
          break;
      default:
          break;
      }
      setcolor(RED);
      itoa(count,str1,10);
      settextstyle(1,0,2);
      outtextxy(130,105,str1);
      setfillstyle(1,LIGHTBLUE);
      setcolor(BLUE);
      fillellipse(41-5,140,10,10);
      fillellipse(292+5,140,10,10);
      setfillstyle(1,LIGHTGRAY);
      fillpoly(3,arr1);
      fillpoly(3,arr2);

    }

    void warehouse_list(U_ware *w)
    {
      int i;
      cleardevice();
      setbkcolor(WHITE);
      clrmous(MouseX,MouseY);
      setfillstyle(1,LIGHTGRAY);
      settextjustify(0,2);
      puthz(220,30,"当前仓库列表",32,32,BLUE);
      bar(100,100,540,400);

      settextstyle(1,0,3);
      setfillstyle(1,WHITE);
      for(i=0;i<num_ware;i++)
      {
          char str[15];
          int up=100+60*i,down=160+i*60,type=w[i].cotton_type;
          bar(100+2,up+2,540-2,down-2);
          setlinestyle(0,0,1);
          setcolor(LIGHTBLUE);
          rectangle(100+3,up+3,540-3,down-3);
          setcolor(RED);
          puthz(104,up+10,"仓库名：",16,16,DARKGRAY);
          outtextxy(168,up+5,w[i].ware_name);
          puthz(104,up+30,"棉花种类：",16,16,DARKGRAY);
          puthz(320,up+30,"库存量：",16,16,DARKGRAY);
          puthz(510,up+30,"吨",16,16,DARKGRAY);
          switch (type)
          {
              case 0:
              {
 if(w[i].total[type]<=ware_full)
```

```
  itoa(w[i].total[type],str,10);
                        else
                              itoa(ware_full,str,10);
                        puthz(184,up+30," 长 绒 棉
",16,16,RED);
                        outtextxy(384,up+25,str);
                        break;
                  }
                  case 1:
                  {

  if(w[i].total[type]<=ware_full)

  itoa(w[i].total[type],str,10);
                        else
                              itoa(ware_full,str,10);
                        puthz(184,up+30," 细 绒 棉
",16,16,RED);
                        outtextxy(384,up+25,str);
                        break;
                  }
                  case 2:
                  {

  if(w[i].total[type]<=ware_full)

  itoa(w[i].total[type],str,10);
                        else
                              itoa(ware_full,str,10);
                        puthz(184,up+30," 粗 绒 棉
",16,16,RED);
                        outtextxy(384,up+25,str);
                        break;
                  }

                  default:
                        break;
            }
      }
      quit();
      last();

      // while(1)
      // {
      //
  newmouse(&MouseX,&MouseY,&press);
```

```
      //    press_warelist(num_ware);
      //    delay(15);
      // }
    }

    void press_warelist(int *re)
    {
      int i;
      for(i=0;i<num_ware;i++)
      {
            int up=100+60*i,down=160+i*60;

  if(mouse_press(100+2,up+2,540-2,down-2)==0)
            {
                  MouseS=0;
                  continue;
            }
            else
if(mouse_press(100+2,up+2,540-2,down-2)==2)
            {
                  MouseS=1;
                  return;
            }
            else
if(mouse_press(100+2,up+2,540-2,down-2)==1)
            {
                  k=i;
                  // draw_home01();
                  *re=1;
                  mode1=0;
                  return;
            }
      }
      if(mouse_press(0,0,40,30)==0||mouse_pre
ss(0,450,40,480)==0)
      {
            MouseS=0;
      }
      if(mouse_press(0,0,40,30)==2||mouse_pre
ss(0,450,40,480)==2)
      {
            MouseS=1;
      }
      if(mouse_press(0,450,40,480)==1)
      {
            // draw_home01();
            mode1=0;
  }
  else if(mouse_press(0,0,40,30)==1)
  {
        // draw_wel();
        wr_h();
        free(h);
        exit(0);
  }
}


//detail message of warehouse
void detailed_warehouse(long int count)
{
  char str1[8];
  cleardevice();
  setbkcolor(WHITE);
  clrmous(MouseX,MouseY);
  setfillstyle(1,LIGHTGRAY);
  puthz(220,30,"当前仓储信息",32,32,BLUE);
  bar(100,100,540,300);
  puthz(120,130,"库存量：",32,32,WHITE);
  puthz(400,130,"吨",32,32,WHITE);
  puthz(120,220,"棉花种类：",32,32,WHITE);
  switch (h->here[k].cotton_type)
  {
  case 0:
        puthz(280,220,"长绒棉",32,32,RED);
        break;

  case 1:
        puthz(280,220,"细绒棉",32,32,RED);
        break;

  case 2:
        puthz(280,220,"粗绒棉",32,32,RED);
        break;

  default:
        break;
  }
  itoa(count,str1,10);
  setcolor(RED);
  settextstyle(1,0,4);
  outtextxy(250,125,str1);
  last();
        setfillstyle(1,LIGHTBLUE);
        bar(140,320,240,380);
        setfillstyle(1,RED);
        bar(380,320,480,380);
        puthz(155,333,"采摘",32,32,WHITE);
        puthz(395,333,"出库",32,32,WHITE);

        quit();
        // for(;;)
        // {
        //
  newmouse(&MouseX,&MouseY,&press);
        //     press_detwarehouse(count);
        //     delay(15);
        // }
      }


      void press_detwarehouse()
      {
        if(mouse_press(0,0,40,30)==0||mouse_pre
ss(140,320,240,380)==0||mouse_press(380,320,4
80,380)==0\
        ||mouse_press(0,450,40,480)==0)
        {
              MouseS=0;
        }
        if(mouse_press(0,0,40,30)==2||mouse_pre
ss(140,320,240,380)==2||mouse_press(380,320,4
80,380)==2\
        ||mouse_press(0,450,40,480)==2)
        {
              MouseS=1;
        }
        if(mouse_press(0,0,40,30)==1)
        {
              // draw_wel();
              wr_h();
              free(h);
              exit(0);
        }
        if(mouse_press(0,450,40,480)==1)
        {
              // draw_home01();
              mode1=0;
        }
        if(mouse_press(140,320,240,380)==1)
  {
        // draw_simu01(x_max,y_max,5);
        mode=3;
        mode1=0;
  }
  if(mouse_press(380,320,480,380)==1)
  {
        // out_warehouse(count);
        mode1=3;
  }
}

//page of the cotton out
void out_warehouse()
{
  //int out;
  clrmous(MouseX,MouseY);
  //int kick=0;
  cleardevice();
  setbkcolor(WHITE);
  setfillstyle(1,LIGHTGRAY);
  puthz(220,30,"请输入出库量",32,32,BLUE);

  setfillstyle(1,LIGHTGRAY);
  bar(100,100,540,300);
  setfillstyle(1,WHITE);
  bar(130,150,510,250);
  puthz(460,190,"吨",32,32,BLUE);
  setfillstyle(1,RED);
  bar(270,320,370,380);
  puthz(285,333,"完成",32,32,WHITE);
  last();

  quit();
  // for(;;)
  // {
  //
newmouse(&MouseX,&MouseY,&press);
  //    press_outware(count,str);
  //    delay(15);
  // }
  //return out;
}

void press_outware()
{
  if(mouse_press(0,0,40,30)==0||mouse_pre
ss(130,150,510,250)==0||mouse_press(270,320,3
70,380)==0\
        ||mouse_press(0,450,40,480)==0)
        {
              MouseS=0;
        }
        if(mouse_press(0,0,40,30)==2||mouse_pre
ss(130,150,510,250)==2||mouse_press(270,320,3
70,380)==2\
        ||mouse_press(0,450,40,480)==2)
        {
              MouseS=1;
        }
        if(mouse_press(0,0,40,30)==1)
        {
              // draw_wel();
              wr_h();
              free(h);
              exit(0);
        }
        if(mouse_press(0,450,40,480)==1)
        {
              // detailed_warehouse(count);
              mode1=2;
        }
        // if(mouse_press(0,0,40,30)==1)
        // {
        //    // detailed_warehouse(count);
        //    mode=2;
        // }
        if(mouse_press(130,150,510,250)==1)
        {

  input_text(str,140,190,15,DARKGRAY,1);
              return;
        }
        if(mouse_press(270,320,370,380)==1)
        {
              out_finish();
        }
        //return out;
      }

      void out_finish()
      {
        clrmous(MouseX,MouseY);
        setfillstyle(1,WHITE);
  bar(200,160,430,240);
  setcolor(BLUE);
  setlinestyle(0,0,3);
  rectangle(200,160,430,240);
  puthz(220,180,"出库完成",48,48,RED);
  delay(1000);
  // draw_home01();
  mode1=0;
}

//draw the trunk int the page
void draw_trunk()
{
  setfillstyle(1,RED);
  //setcolor(DARKGRAY);
  bar(50,210,160,260);
  setfillstyle(1,LIGHTGRAY);
  bar(160,225,190,260);
  setfillstyle(1,DARKGRAY);
  fillellipse(175,255,11,11);
  fillellipse(76,255,11,11);
  fillellipse(104,255,11,11);
  setfillstyle(1,LIGHTBLUE);
  bar(165,230,185,240);
}

//draw the picture of warehouse
void draw_warehouse()
{
  int i,j,y_d=160;
  setcolor(DARKGRAY);
  setlinestyle(0,0,3);
  line(0,230,640,230);
  for(i=0;i<4;i++)
  {
        int x_d=400;
        for(j=0;j<3;j++)
        {
              int m=rand()%10;
              cot_mount(x_d+m,y_d);
              x_d+=70;
        }
        y_d+=80;
  }
}

//draw the cotton in warehouse
```

```
    void cot_mount(int x,int y)
    {
      int i,d_y=y;
      setcolor(DARKGRAY);
      //setlinestyle(0,0,3);
      setfillstyle(0,WHITE);
      for(i=0;i<4;i++)
      {
            rectangle(x,d_y,x+50,d_y+30);
            bar(x,d_y,x+50,d_y+30);
            d_y+=30;
      }
    }

    /*int main()
    {
      int gd=VGA,gm=VGAHI;
      initgraph(&gd,&gm,"..\\borlandc\\bgi");
      draw_home01();
      closegraph();
      return 0;
}*/
```

5、 LOGFUN.C

```
    #include "COMMON.H"
    #include "LOGFUN.H"
    #include "PARAMETE.H"

    // 录入注册账号到文件
    void wr_user(char username1[], char
password1[], char phonenumber1[])
    {
      FILE *fp;
      int i,j;
      user *u = (user *)malloc(sizeof(user));
      u->lenpar = 0;
      for (i = 0; i < 5; i++)
      {
            u->here[i].cotton_type = 0;
      }
      for (i=0;i<5;i++)
      {
            u->here[i].ware_name[0]='\0';
            for (j=0;j<3;j++)
            {
                  u->here[i].total[j]=0;
            }
    }
    if ((fp = fopen("User.dat", "rb+")) == NULL)
    {
        puthz(120, 300, "打开错误", 32, 32, BLUE);
        delay(3000);
        return;
    }
    for (i = 0; i < 10; i++)
    {
        u->username[i] = username1[i];
        u->password[i] = password1[i];
    }
    for (i = 0; i < 12; i++)
    {
        u->phonenumber[i] = phonenumber1[i];
    }
    fseek(fp, 0, SEEK_END);
    fwrite(u, sizeof(user), 1, fp);
    free(u);
    u = NULL;
    if (fclose(fp) != 0) // 关闭文件
    {
        puthz(120, 300, "关闭错误", 32, 32, BLUE);
        delay(3000);
        return;
    }
}

// 判断注册的账号是否已经存在和电话号码是否正确
// return flag 为 1 则存在，0 则不存在
int username_same(char username0[], char phonenumber0[])
{
    FILE *fp;
    user *u = (user *)malloc(sizeof(user));
    int i, j, all, flag = 0;
    if ((fp = fopen("User.dat", "rb+")) == NULL)
    {
        puthz(120, 300, "打开错误", 32, 32, BLUE);
        delay(3000);
        return 0;
    }
    fseek(fp, 0, SEEK_END);
    all = ftell(fp) / sizeof(user);
    for (i = 0; i < all; i++)
    {
        flag = 0;
        fseek(fp, i * sizeof(user), SEEK_SET);
        fread(u, sizeof(user), 1, fp);

        for (j = 0; j < 10; j++) // 查找账号位置
        {
            if (u->username[j] != username0[j])
            {
                break;
            }
            if (username0[j] == '\0')
            {
                j = 10;
                break;
            }
        }
        if (j == 10)
        {
            setfillstyle(1, CYAN);
            bar(200, 30, 440, 72);
            puthz(210, 32, "注册账号已存在", 32, 32, BLUE);
            delay(1200);
            setfillstyle(1, 0);
            bar(200, 30, 440, 72);
            puthz(260, 30, "注册账号", 32, 32, BLUE);
            flag = 1;
        }
    }
    if (strlen(phonenumber0) != 11)
    {
        setfillstyle(1, CYAN);
        bar(180, 30, 460, 72);
        puthz(190, 32, "请输入十一位号码", 32, 32, BLUE);
        delay(1200);
        setfillstyle(1, 0);
        bar(180, 30, 460, 72);
        puthz(260, 30, "注册账号", 32, 32, BLUE);
        flag = 1;
    }
    if (flag == 0)
    {
        setfillstyle(1, CYAN);
        bar(200, 30, 440, 72);
        puthz(210, 32, "注册账号成功", 32, 32,
BLUE);
        delay(1200);
        setfillstyle(1, 0);
        bar(200, 30, 440, 72);
        puthz(260, 30, "注册账号", 32, 32,
BLUE);
    }
    if (fclose(fp) != 0) // 关闭文件
    {
        puthz(120, 300, "关闭错误", 32, 32,
BLUE);
        delay(3000);
        return 0;
    }
    free(u);
    u = NULL;
    return flag;
  }

  // 登录，判断账号是否存在 and 密码是否
正确
  int logg(char username0[], char password0[])
  {
    FILE *fp;
    user *u = (user *)malloc(sizeof(user));
    int i, j, k, l, o, flag = 0, all;
    // char a[2];
    // a[2] = '\0';

    if ((fp = fopen("User.dat", "rb+")) == NULL)
    {
        puthz(120, 300, "打开错误", 32, 32,
BLUE);
        delay(3000);
        return 0;
    }
    fseek(fp, 0, SEEK_END);
    all = ftell(fp) / sizeof(user);
    for (i = 0; i < all; i++)
    {
        fseek(fp, i * sizeof(user), SEEK_SET);
        fread(u, sizeof(user), 1, fp);

        for (j = 0; j < 10; j++) // 查找账号位置
        {
            if (u->username[j] !=
username0[j])
            {
                break;
            }
            if (username0[j] == '\0')
            {
                j = 10;
                break;
            }
        }
        if (j == 10) // 如果找到了账号就开始
比对密码
        {
            for (k = 0; k < 10; k++)
            {
                if (u->password[k] !=
password0[k])
                {
                    break;
                }
                if (password0[k] == '\0')
                {
                    k = 10;
                    break;
                }
            }
            if (k == 10)
            {
                strcpy(h->username,
u->username);
                strcpy(h->password,
u->password);
                strcpy(h->phonenumber,
u->phonenumber);
                h->lenpar = u->lenpar;
                for (l = 0; l < u->lenpar; l++)
                {

 parcpy(&(h->parameter[l]),&(u->parameter[l]));
                }
                for (l = 0; l < 5; l++)
```

```
					{
						// for (k = 0; k < 15; k++)
						// {
						//
  h->here[l].ware_name[k]                        =
u->here[l].ware_name[k];
						// }

  strcpy(h->here[l].ware_name,u->here[l].ware_na
me);
						for (k = 0; k < 3; k++)
						{
							h->here[l].total[k]
= u->here[l].total[k];
						}
						h->here[l].cotton_type
= u->here[l].cotton_type;
					}
					flag = 1;
					break;
				}
			}
		}
		if ((j != 10) && (flag == 0))
		{
			setfillstyle(1, CYAN);
			bar(220, 30, 420, 80);
			puthz(250, 35, "账号未注册", 32, 32,
BLUE);
			delay(1000);
			setfillstyle(1, 0);
			bar(220, 30, 420, 80);
			puthz(180, 30, "棉花模拟采集系统",
32, 32, BLUE);
		}
		else if ((k != 10) && (flag == 0))
		{
			setfillstyle(1, CYAN);
			bar(220, 30, 420, 80);
			puthz(250, 35, "密码不正确", 32, 32,
BLUE);
			delay(1000);
			setfillstyle(1, 0);
			bar(220, 30, 420, 80);
			puthz(180, 30, "棉花模拟采集系统",
32, 32, BLUE);
		}
		else if (flag == 0)
		{
			setfillstyle(1, CYAN);
			bar(240, 30, 400, 80);
			puthz(250, 35, "登录失败", 32, 32,
BLUE);
			delay(1000);
			setfillstyle(1, 0);
			bar(240, 30, 400, 80);
			puthz(180, 30, "棉花模拟采集系统",
32, 32, BLUE);
		}
		if (fclose(fp) != 0) // 关闭文件
		{
			puthz(120, 300, "关闭错误", 32, 32,
BLUE);
			delay(3000);
			return 0;
		}
		free(u);
		u = NULL;
		return flag;
	}

	int changepassword(char username0[], char
newpassword0[], char phonenumber0[])
	{
		FILE *fp;
		user *u = (user *)malloc(sizeof(user));
		int i, j, k, flag = 0, all;

		if ((fp = fopen("User.dat", "rb+")) == NULL)
		{
			puthz(120, 300, "打开错误", 32, 32,
BLUE);
			delay(3000);
			return 0;
		}
		fseek(fp, 0, SEEK_END);
		all = ftell(fp) / sizeof(user);
		for (i = 0; i < all; i++)
		{
			fseek(fp, i * sizeof(user), SEEK_SET);
			fread(u, sizeof(user), 1, fp);
			for (j = 0; j < 10; j++) // 查找账号位置
			{
```

```
            if      (u->username[j]       !=
username0[j])
            {
                break;
            }
            if (username0[j] == '\0')
            {
                j = 10;
                break;
            }
        }
        if (j == 10) // 找到了就判断电话号码
正不正确
        {

            for (j = 0; j < 12; j++)
            {

                if   (u->phonenumber[j]   !=
phonenumber0[j])
                {
                    //
outtextxy(200,200,u->phonenumber);
                    break;
                }
                if (phonenumber0[j] == '\0')
                {
                    j = 12;
                    break;
                }
            }
            if (j == 12) // 电话号码正确就改
密码
            {
                //
outtextxy(200,200,u->phonenumber);
                for (k = 0; k < 10; k++)
                {
                    u->password[k]        =
newpassword0[k];
                }
                fseek(fp,   i   *   sizeof(user),
SEEK_SET);
                fwrite(u, sizeof(user), 1, fp);
                flag = 1;
            }
        }
    }
    if (flag == 1)
    {
        setfillstyle(1, CYAN);
        bar(220, 100, 420, 150);
        puthz(220,  100,  "更改密码成功",  32,
32, BLUE);
        delay(1500);
    }
    else
    {
        setfillstyle(1, CYAN);
        bar(220, 80, 420, 130);
       puthz(220, 85, "更改密码失败", 32, 32,
BLUE);
        delay(1200);
        setfillstyle(1, LIGHTBLUE);
        bar(220, 80, 420, 130);
    }

    free(u);
    u = NULL;
    if (fclose(fp) != 0) // 关闭文件
    {
        puthz(120,  300,  "关闭错误",  32,  32,
BLUE);
        delay(3000);
        return 0;
    }
    return flag;
}
```

6、 LOGIN.C

```
    #include "COMMON.H"
    #include "PARAMETE.H"
    #include "LOGFUN.H"
    #include "LOGIN.H"
    #include "REGISTER.H"
    #include "RESET.H"


    void loginit_screen()
    {
        clrmous(MouseX,MouseY);
        cleardevice();
        setbkcolor(WHITE);
        puthz(180,30," 棉 花 模 拟 采 集 系 统
",32,32,BLUE);
            quit();


            setcolor(BLUE);
            setfillstyle(1,LIGHTGRAY);
            //rectangle(123, 103, 523, 153);
            bar(120, 100, 220, 160);
            //rectangle(123, 203, 523, 253);
            bar(120, 200, 220, 260);
            setfillstyle(1,LIGHTGREEN);
            rectangle(220, 100, 520, 160);
            rectangle(220, 200, 520, 260);
            //bar(220, 100, 520, 160);
            //bar(220, 200, 520, 260);

            bar(280, 300, 360, 340);

            setfillstyle(1,DARKGRAY);
            bar(120, 350, 280, 400);
            bar(360, 350, 520, 400);

            puthz(123,110,"账号：",32,32,BLUE);
            puthz(123,210,"密码：",32,32,BLUE);
            puthz(130,360,"    忘    记    密    码
",32,32,WHITE);
            puthz(370,360,"    注    册    账    号
",32,32,WHITE);
            puthz(285,305,"登录",32,32,BLUE);
      }

      int logenter()
      {
            if(mouse_press(0,0,40,30)==1)
        {
              exit(0);

        }
            if(mouse_press(280, 300, 360, 340)==1)
            {
                  return 1;
            }
            return 0;
      }

      void loginit()
      {
            INPUT username = {220, 100, 520,
160,"",10,0,0};
          INPUT password = {220, 200, 520,
260,"",10,0,0};
            loginit_screen();
            clrmous(MouseX, MouseY);
            for(;;)
          {

  newmouse(&MouseX,&MouseY,&press);
                  if(logenter()==1)
                  {

if(logg(username.string,password.string)==1)
                        {
                              return;
                        }
                  }
                  input_s(223, 100, &username, 16 ,
0);
                  input_s(223, 200, &password, 16 ,
1);
                  if(mouse_press(360, 350, 520,
400)==1)
                  {
                        log_register();
                        loginit_screen();
                  }
                  if(mouse_press(120, 350, 280,
400)==1)
                  {
                        reset();
                        loginit_screen();
                  }


          delay(15);
          }
}
```

7、 MAIN.C

```
#include "COMMON.H"
#include "LOGFUN.H"
#include "LOGIN.H"
#include "PARAMETE.H"
#include "START.H"
#include "HOME.H"
```

```
        #include "HELP.H"
        #include "EDIT.H"
        #include "WELCOME.H"
        #include "PAST.H"
        struct User *h;//登录的用户
        int mode=0;
        int mode1=0;

        void main()
        {
            int gd=VGA,gm=VGAHI,i=0;
          h=(user*)malloc(sizeof(user));// 登 录 的 用
户
          initgraph(&gd,&gm,"..\\borlandc\\bgi");
            loginit();
            for(i=0;i<5;i++)
            {

if(strcmp(h->here[i].ware_name,"\0")==0)
                {
                    char s[10]="ware0";
                    s[5]='0'+i+1;

strcpy(h->here[i].ware_name,s);
                }
            }
            while(1)
            {
                int
pre_mode=mode,pre_mode1=mode1;
                switch (mode)
                {
                    case 0:   //the welcome page
                    {
                        draw_wel();
                        break;
                    }

                    case 1: //the home page
                    {

                        switch(mode1)
                        {
                            case -1:
                            {
                                //
draw_home01();
                                mode1=0;
                                break;
                            }
                            case  0:  //the  home
main page
                            {

draw_home01();
                                break;
                            }
                            case      1:      //the
warehouse list page
                            {

warehouse_list(h->here);
                                break;
                            }
                            case      2:      //the
detailed warehouse page
                            {

detailed_warehouse(h->here[k].total[(h->here[k].
cotton_type)]);
                                break;
                            }
                            case  3:  //the  export
cotton page
                            {

out_warehouse();
                                break;
                            }
                        }
                        break;
                    }
                    case 2: //the edit page
                    {
                        edit();
                        pre_mode=2;
                        mode=2;
                        break;
                    }

                    case 3: // the start page
                    {
                        start();
                        pre_mode=3;
```

```
					// mode=3;
					break;
				}

				case 4: //the past page
				{
					past();
					pre_mode=4;
					mode=4;
					break;
				}

				case 5:	//the help page
				{
					help();
					break;
				}

				default:
					break;
			}
while(pre_mode==mode&&pre_mode1==mode1)
			{
newmouse(&MouseX,&MouseY,&press);
				switch (mode)
				{
					case  0:	//the  welcome
page
					{
						enter_next();
						break;
					}

					case 1: //the home page
					{
						switch(mode1)
						{
							case		-1:
//refresh the home main page
							{
clrmous(MouseX,MouseY);
								//
press_home(&(here[k].cotton_type));
								mode1=0;
								break;
							}
							case	0:	//the
home main page
							{
press_home(&(h->here[k].cotton_type));
								break;
							}
							case	1:	//the
warehouse list page
							{
								int re=0;
press_warelist(&re);
								break;
							}
							case	2:	//the
detailed warehouse page
							{
press_detwarehouse();
								break;
							}
							case	3:	//the
export cotton page
							{
press_outware();
								break;
							}
						}
						break;
					}
					case 2: //the edit page
					{
						cleardevice();
						mode=0;
						break;
					}

					case 3: // the start page
					{
						cleardevice();
						mode=0;
						break;
					}
```

```
                        case 4: //the past page
                        {
                            cleardevice();
                            mode=0;
                            break;
                        }

                        case 5:   //the help page
                        {
                            cleardevice();
                            mode=0;
                            break;
                        }
                        default:
                            break;
                    }
                    delay(20);
                }
            }
        }
```

8、 PARAMETE.C

```
    include "COMMON.H"
    #include "PARAMETE.H"
    #include "LOGFUN.H"
    #include "START.H"
    // 专门用来写有关 edit 和 past 的函数

    // 键盘输入    mode 为 0 输出文字，mode
为 1 输出*
    int input_s(int x, int y, INPUT *word, int size,
int mode)
    {
        static int p = 0; // 画一个框
        int k = 0;           // 判断是否输出文
字
        settextjustify(0, 2);
        if (p == 0)
        {
            setcolor(BLUE);
            rectangle(word->x1,    word->y1,
word->x2, word->y2);
            setfillstyle(1, 0);
            bar(word->x1 + 2, word->y1 + 2,
word->x2 - 2, word->y2 - 2);
            p = 1;
        }
        if (press == 1)
        {
            if       (mouse_press(word->x1,
word->y1, word->x2, word->y2) == 1)
            {
                word->flag = 1;

                clrmous(MouseX, MouseY);
                setcolor(RED);
                setlinestyle(0, 0, 1);
                rectangle(word->x1,  word->y1,
word->x2, word->y2);
                setcolor(DARKGRAY);
                k = 1;
            }
            else
            {
                word->flag = 0;

                clrmous(MouseX, MouseY);
                setcolor(BLUE);
                setlinestyle(0, 0, 1);
                rectangle(word->x1,  word->y1,
word->x2, word->y2);
                // 不可输入则把光标遮蔽掉
                if (mode == 0)
                {
                    setfillstyle(1, 0);
                    bar(word->x1   +   2,
word->y1 + 2, word->x2 - 2, word->y2 - 2);
                    setcolor(DARKGRAY);
                    outtextxy(x,          y,
word->string);
                }
                else
                {
                    int i;
                    setfillstyle(1, 0);
                    bar(word->x1   +   2,
word->y1 + 2, word->x2 - 2, word->y2 - 2);
                    for  (i  =  0;  i  <
word->cursor; i++)
                    {
                        outtextxy(x + i * (2 *
size - 2), y, "*");
```

```
                    }
                    setcolor(DARKGRAY);
                }
            }
        }

        // flag 为 1 时表示可以接收键盘输入
        if (word->flag == 1)
        {
            char t;

            if (kbhit())
            {
                t = getch();

                if (t == '\b')
                {
                    if (word->cursor > 0)
                    {
(word->string)[word->cursor - 1] = '\0';
                        (word->cursor)--;
                        k = 1;
                    }
                }
                else if (t >= '!' && t <= '~')
                {
                    if      (word->cursor       <
word->length)
                    {

(word->string)[word->cursor] = t;

(word->string)[word->cursor + 1] = '\0';
                        (word->cursor)++;
                        k = 1;
                    }
                    else
                    {
                        return 1;
                    }
                }
            }
            if (k == 1)
            {
                setcolor(DARKGRAY);
                setlinestyle(0, 0, 1);
                setfillstyle(SOLID_FILL,
WHITE);
                settextjustify(LEFT_TEXT,
TOP_TEXT);
                settextstyle(SMALL_FONT,
HORIZ_DIR, size);
                bar(word->x1 + 2, word->y1 +
2, word->x2 - 2, word->y2 - 2);
                if (mode == 0)
                {
                    setcolor(DARKGRAY);
                    outtextxy(x,              y,
word->string);
                    line(x + (word->cursor) *
(2 * size - 8) + 2, word->y1 + 3, x + (word->cursor)
* (2 * size - 8) + 2, word->y2 - 3);
                }
                else
                {
                    int i;
                    for   (i   =   0;   i   <
word->cursor; i++)
                    {
                        outtextxy(x + i * (2 *
size - 2), y, "*");
                    }
                    setcolor(DARKGRAY);
                    line(x + (word->cursor) *
(2 * size - 2) + 2, word->y1 + 3, x + (word->cursor)
* (2 * size - 2) + 2, word->y2 - 3);
                }
            }
        }
        return 0;
    }

    int judgename(char name[])
    {
        static int i, j;
        if (name[0] == '\0')
        {
            void *buffer;
            unsigned s;
            setfillstyle(1, CYAN);
            bar(240, 30, 470, 70);
            puthz(242, 33, "参数名不能为空",
32, 32, BLUE);
```

```
                delay(1000);
                setfillstyle(1, 0);
                bar(240, 30, 470, 70);
                puthz(240, 30, "请选择地区", 32,
32, BLUE);
                return 0;
            }
            for (i = 0; i < h->lenpar; i++)
            {
                for (j = 0; j < 10; j++)
                {
                    if (name[j] !=
(h->parameter[i]).name[j])
                    {
                        break;
                    }
                    if (name[j] == '\0')
                    {
                        j = 10;
                        break;
                    }
                }
                if (j == 10)
                {
                    setfillstyle(1, CYAN);
                    bar(240, 30, 470, 70);
                    puthz(242, 33, "参数名已经
存在", 32, 32, BLUE);
                    delay(1000);
                    setfillstyle(1, 0);
                    bar(240, 30, 470, 70);
                    puthz(240, 30, "请选择地区",
32, 32, BLUE);

                    return 0;
                }
            }
            return 1;
        }

        int judgeS(char S[])
        {
            int i = 0;
            if (S[0] == '\0')
            {
                setfillstyle(1, CYAN);
                bar(80, 100, 380, 140);
                puthz(83, 100, "土地面积不能为
空", 32, 32, BLUE);
                delay(1000);
                setfillstyle(1, 0);
                bar(80, 100, 380, 140);
                puthz(70, 100, "本地推荐种植的
棉花种类为：", 32, 32, BLUE);
                return 0;
            }
            for (i = 0; i < strlen(S); i++)
            {
                if (S[i] > '9' || S[i] < '0')
                {
                    setfillstyle(1, CYAN);
                    bar(80, 100, 340, 140);
                    puthz(83, 100, "请输入数字",
32, 32, BLUE);
                    delay(1000);
                    setfillstyle(1, 0);
                    bar(80, 100, 340, 140);
                    puthz(70, 100, "本地推荐种
植的棉花种类为：", 32, 32, BLUE);
               return 0;
                }
            }
            if (S[0] == '0')
            {
                setfillstyle(1, CYAN);
                bar(80, 100, 380, 140);
                puthz(83, 100, "土地面积不能为
零", 32, 32, BLUE);
                delay(1000);
                setfillstyle(1, 0);
                bar(80, 100, 380, 140);
                puthz(70, 100, "本地推荐种植的
棉花种类为：", 32, 32, BLUE);
                return 0;
            }
            return 1;
        }

        void wr_parameter(struct Parameter *abc)
        {
            FILE *fp;
            user *u = (user *)malloc(sizeof(user));
            int i, j, k, all;
            if ((fp = fopen("User.dat", "rb+")) ==
NULL)
        {
            puthz(120, 300, "打开错误", 32, 32,
BLUE);
            delay(3000);
            return;
        }
        fseek(fp, 0, SEEK_END);
        all = ftell(fp) / sizeof(user); // 文件里
user 的数量
        for (i = 0; i < all; i++)
        {
            fseek(fp, i * sizeof(user),
SEEK_SET);
            fread(u, sizeof(user), 1, fp);
            for (j = 0; j < 10; j++) // 查找账号
位置
            {

                if (u->username[j] !=
h->username[j])
                {
                    break;
                }
                if (h->username[j] == '\0')
                {
                    j = 10;
                    break;
                }
            }
            if (j == 10) // 找到了就修改参数
            {
                //
strcpy(h->parameter[h->lenpar].name,abc->name
);
                for (k = 0; k < 10; k++)
                {

h->parameter[h->lenpar].name[k] =
abc->name[k];
                }

h->parameter[h->lenpar].place = abc->place;

h->parameter[h->lenpar].shape = abc->shape;
                h->parameter[h->lenpar].type
= abc->type;

strcpy(h->parameter[h->lenpar].S, abc->S);
                for (k = 0; k <
dense_points_max; k++)
                {

h->parameter[h->lenpar].x[k] = abc->x[k];

h->parameter[h->lenpar].y[k] = abc->y[k];
                }

h->parameter[h->lenpar].lenxy = abc->lenxy;
                h->lenpar += 1;
                fseek(fp, i * sizeof(user),
SEEK_SET);
                fwrite(h, sizeof(user), 1, fp);
                puthz(220, 100, "增加参数成
功", 32, 32, BLUE);
                delay(2000);
            }
        }
        // char name[10];//名字
        // char place;//产区
        // char shape;//土地形状
        // char type;//收割机类型
        // int S;//面积
        // int xyz[200];//坐标

        free(u);
        u = NULL;
        if (fclose(fp) != 0) // 关闭文件
        {
            puthz(120, 300, "关闭错误", 32, 32,
BLUE);
            delay(3000);
            return;
        }
        return;
    }

    // 把当前登录的用户重新写进文件
    void wr_h(void)
    {
        FILE *fp;
        user *u = (user *)malloc(sizeof(user));
        int i, j, k, all;
        if ((fp = fopen("User.dat", "rb+")) ==
NULL)
        {
            puthz(120, 300, "打开错误", 32, 32,
BLUE);
            delay(3000);
            return;
        }
        fseek(fp, 0, SEEK_END);
        all = ftell(fp) / sizeof(user); // 文件里
user 的数量
        for (i = 0; i < all; i++)
        {
            fseek(fp, i * sizeof(user),
SEEK_SET);
            fread(u, sizeof(user), 1, fp);
            for (j = 0; j < 10; j++)
            {

                if (u->username[j] !=
h->username[j])
                {
                    break;
                }
                if (h->username[j] == '\0')
                {
                    j = 10;
                    break;
                }
            }
            if (j == 10) // 找到了就把 h 传进去
            {
                fseek(fp, i * sizeof(user),
SEEK_SET);
                fwrite(h, sizeof(user), 1, fp);
            }
        }

        free(u);
        u = NULL;
        if (fclose(fp) != 0) // 关闭文件
        {
            puthz(120, 300, "关闭错误", 32, 32,
BLUE);
            delay(3000);
            return;
        }
        return;
    }

    // 删除参数
    void deletepar(int par)
    {
        int i, j, k;
        par -= 1;
        for (i = par; i < h->lenpar; i++)
        {
            parcpy(&(h->parameter[par]),
&(h->parameter[par + 1]));
        }
        h->lenpar -= 1;
        wr_h();
    }

    // 下面四个返回 1 为修改成功，返回-1 为
无修改

    // 修改产区
    int changeplace(int par)
    {
        char choose;
        clrmous(MouseX, MouseY);
        setfillstyle(1, BROWN);
        bar(100, 130, 540, 370);
        puthz(220, 150, "请重新选择产区", 32,
32, BLUE);
        setfillstyle(1, CYAN);
        bar(150, 290, 240, 340);
        puthz(153, 293, "返回", 32, 32, BLUE);
        bar(400, 290, 490, 340);
        puthz(403, 293, "确认", 32, 32, BLUE);

        setfillstyle(1, LIGHTRED);
        bar(305, 240, 335, 270);
        puthz(300, 200, "长江", 16, 16, BLUE);
        bar(180, 240, 210, 270);
        puthz(175, 200, "新疆", 16, 16, BLUE);
        bar(430, 240, 460, 270);
        puthz(425, 200, "黄河", 16, 16, BLUE);

        quit();

        for (;;)
        {
            delay(15);
```

```
			newmouse(&MouseX,	&MouseY,
&press);

			if (mouse_press(305, 240, 335, 270)
== 1) // 长江
			{
				clrmous(MouseX, MouseY);
				choose = 'c';
				setfillstyle(1, BROWN);
				bar(180, 240, 460, 270);
				setfillstyle(1, BLUE);
				bar(305, 240, 335, 270);
				setfillstyle(1, LIGHTRED);
				bar(180, 240, 210, 270);
				bar(430, 240, 460, 270);
			}
			if (mouse_press(180, 240, 210, 270)
== 1) // 新疆
			{
				clrmous(MouseX, MouseY);
				choose = 'a';
				setfillstyle(1, BROWN);
				bar(180, 240, 460, 270);
				setfillstyle(1, BLUE);
				bar(180, 240, 210, 270);
				setfillstyle(1, LIGHTRED);
				bar(305, 240, 335, 270);
				bar(430, 240, 460, 270);
			}
			if (mouse_press(430, 240, 460, 270)
== 1) // 黄河
			{
				clrmous(MouseX, MouseY);
				choose = 'b';
				setfillstyle(1, BROWN);
				bar(180, 240, 460, 270);
				setfillstyle(1, BLUE);
				bar(430, 240, 460, 270);
				setfillstyle(1, LIGHTRED);
				bar(180, 240, 210, 270);
				bar(305, 240, 335, 270);
			}

			if (mouse_press(150, 290, 240, 340)
== 1) // 返回
			{
				return -1;
			}
			if (mouse_press(400, 290, 490, 340)
== 1) // 确认
			{
				h->parameter[par - 1].place =
choose;
				wr_h();
				return 1;
			}
		}
	}

	// 修改土地形状
	int changeshape(int par)
	{
		char choose;
		clrmous(MouseX, MouseY);
		setfillstyle(1, BROWN);
		bar(100, 130, 540, 370);
		puthz(200, 150, "请重新选择土地形状",
32, 32, BLUE);
		setfillstyle(1, CYAN);
		bar(150, 290, 240, 340);
		puthz(153, 293, "返回", 32, 32, BLUE);
		bar(400, 290, 490, 340);
		puthz(403, 293, "确认", 32, 32, BLUE);

		setfillstyle(1, YELLOW);
		pieslice(170, 240, 0, 360, 20);
		puthz(155, 195, "矩形", 16, 16, BLUE);
		pieslice(270, 240, 0, 360, 20);
		puthz(255, 195, "圆形", 16, 16, BLUE);
		pieslice(370, 240, 0, 360, 20);
		puthz(340, 195, "多边形", 16, 16,
BLUE);
		pieslice(470, 240, 0, 360, 20);
		puthz(425, 195, "自定义图形", 16, 16,
BLUE);

		for (;;)
		{
			delay(15);
			newmouse(&MouseX,	&MouseY,
&press);
			if (mouse_press(150, 220, 190, 360)
== 1) // 矩形
			{
```

```
                        clrmous(MouseX, MouseY);
                        choose = 'a';
                        setfillstyle(1, BROWN);
                        bar(150, 220, 490, 260);
                        setfillstyle(1, YELLOW);
                        pieslice(270, 240, 0, 360, 20);
                        pieslice(370, 240, 0, 360, 20);
                        pieslice(470, 240, 0, 360, 20);
                        setfillstyle(1, BLUE);
                        pieslice(170, 240, 0, 360, 20);
                    }
                    if (mouse_press(250, 220, 290, 360)
== 1) //  圆形
                    {
                        clrmous(MouseX, MouseY);
                        choose = 'b';
                        setfillstyle(1, BROWN);
                        bar(150, 220, 490, 260);
                        setfillstyle(1, YELLOW);
                        pieslice(170, 240, 0, 360, 20);
                        pieslice(370, 240, 0, 360, 20);
                        pieslice(470, 240, 0, 360, 20);
                        setfillstyle(1, BLUE);
                        pieslice(270, 240, 0, 360, 20);
                    }
                    if (mouse_press(350, 220, 390, 360)
== 1) //  多边形
                    {
                        clrmous(MouseX, MouseY);
                        choose = 'c';
                        setfillstyle(1, BROWN);
                        bar(150, 220, 490, 260);
                        setfillstyle(1, YELLOW);
                        pieslice(270, 240, 0, 360, 20);
                        pieslice(170, 240, 0, 360, 20);
                        pieslice(470, 240, 0, 360, 20);
                        setfillstyle(1, BLUE);
                        pieslice(370, 240, 0, 360, 20);
                    }
                    if (mouse_press(450, 220, 490, 360)
== 1) //  自定义图形
                    {
                        clrmous(MouseX, MouseY);
                        choose = 'd';
                        setfillstyle(1, BROWN);
                        bar(150, 220, 490, 260);
                        setfillstyle(1, YELLOW);
                        pieslice(270, 240, 0, 360, 20);
                        pieslice(370, 240, 0, 360, 20);
                        pieslice(170, 240, 0, 360, 20);
                        setfillstyle(1, BLUE);
                        pieslice(470, 240, 0, 360, 20);
                    }
                    if (mouse_press(150, 290, 240, 340)
== 1) //  返回
                    {
                        return -1;
                    }
                    if (mouse_press(400, 290, 490, 340)
== 1) //  确认
                    {
                        if (choose == 'c')
                        {

select02(&(h->parameter[par - 1]));
                        }
                        else if (choose == 'd')
                        {

select03(&(h->parameter[par - 1]));
                        }
                        h->parameter[par - 1].shape =
choose;
                        wr_h();
                        return 1;
                    }
                }
            }

            //  修改收割机类型
            int changetype(int par)
            {
                char choose;
                clrmous(MouseX, MouseY);
                setfillstyle(1, BROWN);
                bar(100, 130, 540, 370);
                puthz(220, 150, "请重新选择产区", 32,
32, BLUE);
                setfillstyle(1, CYAN);
                bar(150, 290, 240, 340);
                puthz(153, 293, "返回", 32, 32, BLUE);
                bar(400, 290, 490, 340);
                puthz(403, 293, "确认", 32, 32, BLUE);
```

```
        setfillstyle(1, YELLOW);
        pieslice(200, 240, 0, 360, 20);
        puthz(175, 195, "垂直式", 16, 16,
BLUE);
        pieslice(440, 240, 0, 360, 20);
        puthz(415, 195, "水平式", 16, 16,
BLUE);

        for (;;)
        {
            delay(15);
            newmouse(&MouseX, &MouseY,
&press);
            if (mouse_press(180, 220, 220, 260)
== 1) // 垂直式
            {
                clrmous(MouseX, MouseY);
                choose = 'a';
                setfillstyle(1, BROWN);
                bar(180, 220, 460, 260);
                setfillstyle(1, YELLOW);
                pieslice(440, 240, 0, 360, 20);
                setfillstyle(1, BLUE);
                pieslice(200, 240, 0, 360, 20);
            }
            if (mouse_press(420, 220, 460, 260)
== 1) // 水平式
            {
                clrmous(MouseX, MouseY);
                choose = 'b';
                setfillstyle(1, BROWN);
                bar(180, 220, 460, 260);
                setfillstyle(1, YELLOW);
                pieslice(200, 240, 0, 360, 20);
                setfillstyle(1, BLUE);
                pieslice(440, 240, 0, 360, 20);
            }
            if (mouse_press(150, 290, 240, 340)
== 1) // 返回
            {
                return -1;
            }
            if (mouse_press(400, 290, 490, 340)
== 1) // 确认
            {
                h->parameter[par - 1].type =
choose;
                wr_h();
                return 1;
            }
        }
    }

    // 修改面积
    int changeS(int par)
    {
        INPUT S = {245, 220, 445, 260, "", 6, 0,
0};
        clrmous(MouseX, MouseY);
        setfillstyle(1, BROWN);
        bar(100, 130, 540, 370);
        puthz(220, 150, "请重新输入面积", 32,
32, BLUE);
        setfillstyle(1, CYAN);
        bar(150, 290, 240, 340);
        puthz(153, 293, "返回", 32, 32, BLUE);
        bar(400, 290, 490, 340);
        puthz(403, 293, "确认", 32, 32, BLUE);
        puthz(170, 220, "面积：", 32, 32, BLUE);
        setfillstyle(1, 0);
        bar(245, 220, 445, 260);

        for (;;)
        {
            delay(15);
            newmouse(&MouseX, &MouseY,
&press);
            input_s(248, 220, &S, 16, 0);

            if (mouse_press(150, 290, 240, 340)
== 1) // 返回
            {
                return -1;
            }
            if (mouse_press(400, 290, 490, 340)
== 1) // 确认
            {
                if (judgeS(S.string) == 1)
                {
                    strcpy(h->parameter[par
- 1].S, S.string);
                    wr_h();
                }
                return 1;
            }
        }
    }

    void parcpy(struct Parameter *a, struct
Parameter *b)
    {
        int k;
        for (k = 0; k < 10; k++)
        {
            a->name[k] = b->name[k];
        }
        a->place = b->place;
        a->shape = b->shape;
        a->type = b->type;
        strcpy(a->S, b->S);
        for (k = 0; k < dense_points_max; k++)
        {
            a->x[k] = b->x[k];
            a->y[k] = b->y[k];
        }
        a->lenxy = b->lenxy;
    }

    int choosepar(void)
    {
        int i, j, flag = 1;
        char page[3] = {'1', '/', '1'};
        int barcolor[11] = {1, 2, 3, 4, 5, 9, 10, 11,
12, 13, 14};
        // delay(400);
        clrmous(MouseX, MouseY);
        cleardevice();
        setbkcolor(WHITE);
        puthz(240, 30, "请选择参数", 32, 32,
BLUE);
        quit();
        last();
        delay(15);

        setfillstyle(1, LIGHTGRAY);
        bar(50, 80, 590, 420);
        settextstyle(0, 0, 2);
        settextjustify(1, 1);

        page[3] = '\0';
        for (i = 0; i < 10; i++)
        {
            char a[1];
            a[1] = '\0';
            if ((h->lenpar / 4) == i)
            {
                itoa(i, a, 10);
                page[2] = a[0];
                if ((h->lenpar % 4) != 0)
                {
                    itoa(i + 1, a, 10);
                    page[2] = a[0];
                }
            }
        }
        for (;;)
        {
            int pagepar;
            char page2[1];
            page2[1] = '\0';
            page2[0] = page[0];
            pagepar = atoi(page2) - 1;
            newmouse(&MouseX, &MouseY,
&press);

            if (flag == 1)
            {
                if (h->lenpar == 0)
                {
                    setfillstyle(1,
LIGHTGRAY);
                    bar(50, 80, 590, 420);
                    settextjustify(1, 1);
                    settextstyle(0, 0, 4);
                    setcolor(RED);
                    outtextxy(320, 250, "No
Parameter");
                    flag = 0;
                }
                else
                {
                    setfillstyle(1,
LIGHTGRAY);
                    bar(50, 80, 590, 420);
                    settextstyle(0, 0, 2);
                    outtextxy(320, 405,
page);
                    outtextxy(280, 405,
"<<");
                        outtextxy(360,          405,
">>");
                        settextstyle(0, 0, 3);
                        setcolor(DARKGRAY);

                        for (i = 0, j = pagepar * 4;
i < (4 > ((h->lenpar) - (pagepar * 4)) ? ((h->lenpar)
- (pagepar * 4)) : 4); i++, j++)
                        {
                            int a, b;
                            a  =  barcolor[rand()  %
12];
                            b  =  barcolor[rand()  %
12];
                            if (a == b)
                            {
                                a += 1;
                            }
                            setcolor(a);
                            setfillstyle(1, b);
                            bar(70, 150 + 60 * i,
320, 150 + 40 + 60 * i);
                         if
(strlen(h->parameter[j].name) >= 7)
                         {
                             settextstyle(0,     0,
2);
                         }
                         else
                         {
                             settextstyle(0,     0,
3);
                         }
                            outtextxy(195, 170 +
60 * i, h->parameter[j].name);
                        }

                        for  (i  =  0;  i  <  (4  >
((h->lenpar)  -  (pagepar  *  4))  ?  ((h->lenpar)  -
(pagepar * 4)) : 4); i++)
                        {
                            setfillstyle(1, 14);
                            bar(460, 150 + 60 * i,
520, 150 + 40 + 60 * i);
                            puthz(470, 160 + 60
* i, "选择", 16, 16, BLUE);
```

```
                                }
                                flag = 0;
                            }
                        }
                        if (mouse_press(260, 395, 310, 415)
== 1)
                        {
                            if (page[0] > '1')
                            {
                                page[0] -= 1;
                                flag = 1;
                                delay(100);
                            }
                            else
                            {
                                setfillstyle(1, CYAN);
                                bar(230, 30, 450, 80);
                                puthz(240,  30,  "第一页
啦", 32, 32, BLUE);
                                delay(300);
                                setfillstyle(1, 0);
                                bar(230, 30, 450, 80);
                                puthz(240,  30,  "参数列
表", 32, 32, BLUE);
                            }
                        }
                        // page next
                        if (mouse_press(330, 395, 380, 415)
== 1)
                        {
                            if (page[0] < page[2])
                            {
                                page[0] += 1;
                                flag = 1;
                                delay(100);
                            }
                            else if (page[0] == page[2])
                            {
                                setfillstyle(1, CYAN);
                                bar(230, 30, 450, 80);
                                puthz(240,  30,  "最后一
页啦", 32, 32, BLUE);
                                delay(300);
                                setfillstyle(1, 0);
                                bar(230, 30, 450, 80);
                                puthz(240,  30,  "参数列
表", 32, 32, BLUE);
```

```
			}
		}

		// 四个选择按钮
		if ((mouse_press(460, 150 + 60 * 0, 520, 150 + 40 + 60 * 0) == 1) && ((h->lenpar - (pagepar * 4)) > 0))
		{
			int pagepar;
			char page2[1];
			page2[1] = '\0';
			page2[0] = page[0];
			pagepar = atoi(page2) - 1;

			clrmous(MouseX, MouseY);
			return (pagepar)*4 + 1;
		}
		if ((mouse_press(460, 150 + 60 * 1, 520, 150 + 40 + 60 * 1) == 1) && ((h->lenpar - (pagepar * 4)) > 1))
		{
			int pagepar;
			char page2[1];
			page2[1] = '\0';
			page2[0] = page[0];
			pagepar = atoi(page2) - 1;
			clrmous(MouseX, MouseY);

			return (pagepar)*4 + 2;
		}
		if ((mouse_press(460, 150 + 60 * 2, 520, 150 + 40 + 60 * 2) == 1) && ((h->lenpar - (pagepar * 4)) > 2))
		{
			int pagepar;
			char page2[1];
			page2[1] = '\0';
			page2[0] = page[0];
			pagepar = atoi(page2) - 1;
			clrmous(MouseX, MouseY);

			return (pagepar)*4 + 3;
		}
		if ((mouse_press(460, 150 + 60 * 3, 520, 150 + 40 + 60 * 3) == 1) && ((h->lenpar - (pagepar * 4)) > 3))
		{
			int pagepar;
			char page2[1];
			page2[1] = '\0';
			page2[0] = page[0];
			pagepar = atoi(page2) - 1;
			clrmous(MouseX, MouseY);

			return (pagepar)*4 + 4;
		}

		// enter
		if (mouse_press(0, 0, 40, 30) == 0 || mouse_press(0, 450, 40, 480) == 0 || mouse_press(260, 395, 310, 415) == 0 || mouse_press(330, 395, 380, 415) == 0 || mouse_press(510, 80, 580, 125) == 0)
		{
			MouseS = 0;
		}
		if (mouse_press(0, 0, 40, 30) == 2 || mouse_press(0, 450, 40, 480) == 2 || mouse_press(260, 395, 310, 415) == 2 || mouse_press(330, 395, 380, 415) == 2 || mouse_press(510, 80, 580, 125) == 2)
		{
			MouseS = 1;
		}
		// quit
		if (mouse_press(0, 0, 40, 30) == 1)
		{
			exit(0);
		}
		// last
		if (mouse_press(0, 450, 40, 480) == 1)

		{
			return -1;
		}
	}
}

// 搜索
int search(char name[])
{
	int i, j, flag = 0;
	clrmous(MouseX, MouseY);
	cleardevice();
```

```
    clrmous(MouseX, MouseY);
    setbkcolor(WHITE);
    quit();

    setfillstyle(1, LIGHTGRAY);
    bar(50, 80, 590, 420);
    settextstyle(0, 0, 3);
    settextjustify(1, 1);
    outtextxy(320, 60, name);

    for (i = 0; i < h->lenpar; i++)
    {
        for (j = 0; j < 10; j++) // 查找账号位置
        {
            if (name[j] != h->parameter[i].name[j])
            {
                break;
            }
            if (name[j] == '\0')
            {
                j = 10;
                break;
            }
        }
        if (j == 10)
        {
            flag = 1;
            break;
        }
    }
    if (flag == 1)
    {
        setfillstyle(1, BLUE);
        bar(90, 150, 300, 150 + 40);
        setfillstyle(1, YELLOW);
        bar(460, 150, 520, 150 + 40);
        puthz(470, 160, "查看", 16, 16, BLUE);
        settextstyle(0, 0, 3);
        settextjustify(1, 1);
        outtextxy(195, 170, h->parameter[i].name);
    }
    else
    {
        settextstyle(0, 0, 3);
        settextjustify(1, 1);
        outtextxy(320, 250, "the parameter");
        outtextxy(320, 200, "Can not find");
    }

    setfillstyle(1, CYAN);
    bar(80, 120 + 60 * 4, 150, 160 + 60 * 4);
    puthz(83, 125 + 60 * 4, "返回", 32, 32, BLUE);

    for (;;)
    {
        newmouse(&MouseX, &MouseY, &press);
        if (mouse_press(460, 150, 520, 150 + 40) == 1)
        {
            return i + 1;
        }
        // enter
        if (mouse_press(0, 0, 40, 30) == 0 || mouse_press(80, 120 + 60 * 4, 150, 160 + 60 * 4) == 0)
        {
            MouseS = 0;
        }
        if (mouse_press(0, 0, 40, 30) == 2 || mouse_press(80, 120 + 60 * 4, 150, 160 + 60 * 4) == 2)
        {
            MouseS = 1;
        }
        // quit
        if (mouse_press(0, 0, 40, 30) == 1)
        {
            exit(0);
        }

        if (mouse_press(80, 120 + 60 * 4, 150, 160 + 60 * 4) == 1)
        {
            return -1;
        }
    }
}
```

```
void changewarename(int wi)
{
    int i, j;
    INPUT name = {185, 220, 455, 270, "", 8, 0, 0};
    clrmous(MouseX, MouseY);
    setfillstyle(1, CYAN);
    bar(100, 130, 540, 370);
    puthz(180, 150, "请重新输入仓库名字", 32, 32, BLUE);
    setfillstyle(1, GREEN);
    bar(150, 290, 240, 340);
    puthz(153, 293, "返回", 32, 32, BLUE);
    bar(400, 290, 490, 340);
    puthz(403, 293, "确认", 32, 32, BLUE);
    setfillstyle(1, 0);
    bar(185, 220, 455, 270);

    for (;;)
    {
        delay(15);
        newmouse(&MouseX, &MouseY, &press);
        input_s(188, 220, &name, 16, 0);
        if (mouse_press(150, 290, 240, 340) == 1) // 返回
        {
            return;
        }
        if (mouse_press(400, 290, 490, 340) == 1) // 确认
        {
            strcpy(h->here[wi - 1].ware_name, name.string);
            wr_h();
            return;
        }
    }
}

void changeparname(int par)
{
    int i, j;
    INPUT name = {175, 220, 465, 270, "", 10, 0, 0};
    clrmous(MouseX, MouseY);
    setfillstyle(1, BROWN);
    bar(100, 130, 540, 370);
    puthz(180, 150, "请重新输入参数名字", 32, 32, BLUE);
    setfillstyle(1, CYAN);
    bar(150, 290, 240, 340);
    puthz(153, 293, "返回", 32, 32, BLUE);
    bar(400, 290, 490, 340);
    puthz(403, 293, "确认", 32, 32, BLUE);
    setfillstyle(1, 0);
    bar(175, 220, 465, 270);

    for (;;)
    {
        delay(15);
        newmouse(&MouseX, &MouseY, &press);
        input_s(178, 220, &name, 16, 0);

        if (mouse_press(150, 290, 240, 340) == 1) // 返回
        {
            return;
        }
        if (mouse_press(400, 290, 490, 340) == 1) // 确认
        {
            strcpy(h->parameter[par - 1].name, name.string);
            wr_h();
            return;
        }
    }
}
```

9、 PAST.C

```
#include "COMMON.H"
#include "PAST.H"
#include "PARAMETE.H"
#include "LOGFUN.H"
#include "time.h"
void past01_screen(void)
{
  int i;
  clrmous(MouseX, MouseY);
  cleardevice();
  clrmous(MouseX, MouseY);
```

```
    setbkcolor(WHITE);
    puthz(240, 30, "参数列表", 32, 32, BLUE);
    quit();
    last();

    setfillstyle(1, LIGHTGRAY);
    bar(50, 80, 590, 420);
    settextstyle(0, 0, 2);
    settextjustify(1, 1);

    setfillstyle(1, YELLOW);
    bar(505, 77, 580, 125);
  }

  int past01()
  {
    int i, j, flag = 1;
              //  判断是否换页
    INPUT searchname = {220, 80, 500, 125, "",
10, 0, 0}; //  搜索
    char page[3] = {'1', '/', '1'};
    int barcolor[11] = {1, 2, 3, 4, 5, 9, 10, 11, 12,
13, 14};
    page[3] = '\0';
    for (i = 0; i < PAR; i++)
    {
        char a[1];
        a[1] = '\0';
        if ((h->lenpar / 4) == i)
        {
            itoa(i, a, 10);
            page[2] = a[0];
            if ((h->lenpar % 4) != 0)
            {
                itoa(i + 1, a, 10);
                page[2] = a[0];
            }
        }
    }
    past01_screen();
    delay(100);
    clrmous(MouseX, MouseY);

    for (;;)
    {
        int pagepar;
        char page2[1];
        page2[1] = '\0';
        page2[0] = page[0];
        pagepar = atoi(page2) - 1;
        newmouse(&MouseX,          &MouseY,
&press);
        if (h->lenpar != 0)
        {
            input_s(223,   80,   &searchname,
16, 0);
        }
        // flag 为 1 则换页
        if (flag == 1)
        {
            if (h->lenpar == 0)
            {
                setfillstyle(1, LIGHTGRAY);
                bar(50, 80, 590, 420);
                settextjustify(1, 1);
                settextstyle(0, 0, 4);
                setcolor(RED);
                outtextxy(320,     250,     "No
Parameter");
                flag = 0;
            }
            else
            {
                setfillstyle(1, LIGHTGRAY);
                bar(50, 80, 590, 420);
                setfillstyle(1, YELLOW);
                bar(505, 77, 580, 125);
               puthz(130, 83, "搜索", 32, 32,
BLUE);
               puthz(505, 83, "确认", 32, 32,
RED);
                settextjustify(1, 1);
                setfillstyle(1, 0);
                bar(220, 80, 500, 125);
                rectangle(220, 80, 500, 125);
                settextstyle(0, 0, 2);
                outtextxy(320, 405, page);
                outtextxy(280, 405, "<<");
                outtextxy(360, 405, ">>");
                // settextstyle(0, 0, 3);
                setcolor(BLACK);

 srand((unsigned)time(NULL));
                for (i = 0, j = pagepar * 4; i <
(4 > ((h->lenpar) - (pagepar * 4)) ? ((h->lenpar) -
(pagepar * 4)) : 4); i++, j++)
                                {
                                        int a, b;
                                        a = barcolor[rand() %
12];
                                        b = barcolor[rand() %
12];
                                        if (a == b)
                                        {
                                                a += 1;
                                        }
                                        setcolor(a);
                                        setfillstyle(1, b);
                                       bar(70, 150 + 60 * i, 320,
150 + 40 + 60 * i);
                                        if
(strlen(h->parameter[j].name) >= 7)
                                        {
                                                settextstyle(0,     0,
2);
                                        }
                                        else
                                        {
                                                settextstyle(0,     0,
3);
                                        }
                                        outtextxy(195, 170 + 60
* i, h->parameter[j].name);
                                }
                                for (i = 0; i < (4 > ((h->lenpar)
- (pagepar * 4)) ? ((h->lenpar) - (pagepar * 4)) : 4);
i++)
                                {
                                        setfillstyle(1, 14);
                                        bar(460, 150 + 60 * i,
520, 150 + 40 + 60 * i);
                                        puthz(474, 160 + 60 * i,
"查看", 16, 16, BLUE);
                                }
                                flag = 0;
                        }
                }
                // search   返回-2
                if (mouse_press(505, 77, 580, 125) ==
1)
                {
                        for (;;)
                        {
                                int act;
                                act                              =
search(searchname.string);
                                if (act != -1)
                                {
                                past02there:
                                        if (past02(act) == -1)
                                        {
                                                goto past02there;
                                        }
                                }
                                else
                                {
                                        break;
                                }
                        }
                        return -2;
                }
                // page last
                if (mouse_press(260, 395, 310, 415) ==
1)
                {
                        if (page[0] > '1')
                        {
                                page[0] -= 1;
                                flag = 1;
                                delay(100);
                        }
                        else
                        {
                                setfillstyle(1, CYAN);
                                bar(230, 30, 450, 80);
                                puthz(240, 30, "第一页啦",
32, 32, BLUE);
                                delay(300);
                                setfillstyle(1, 0);
                                bar(230, 30, 450, 80);
                                puthz(240, 30, "参数列表",
32, 32, BLUE);
                                rectangle(220, 80, 500, 125);
                        }
                }

                // page next
                if (mouse_press(330, 395, 380, 415) ==
1)
		{
			if (page[0] < page[2])
			{
				page[0] += 1;
				flag = 1;
				delay(100);
			}
			else if (page[0] == page[2])
			{
				setfillstyle(1, CYAN);
				bar(230, 30, 450, 80);
				puthz(240, 30, "最后一页啦",
32, 32, BLUE);
				delay(300);
				setfillstyle(1, 0);
				bar(230, 30, 450, 80);
				puthz(240, 30, "参数列表",
32, 32, BLUE);
				rectangle(220, 80, 500, 125);
			}
		}

		// 四个查看按钮
		if ((mouse_press(460, 150 + 60 * 0,
520, 150 + 40 + 60 * 0) == 1) && ((h->lenpar -
(pagepar * 4)) > 0))
		{
			int pagepar;
			char page2[1];
			page2[1] = '\0';
			page2[0] = page[0];
			pagepar = atoi(page2) - 1;

			return (pagepar)*4 + 1;
		}
		if ((mouse_press(460, 150 + 60 * 1,
520, 150 + 40 + 60 * 1) == 1) && ((h->lenpar -
(pagepar * 4)) > 1))
		{
			int pagepar;
			char page2[1];
			page2[1] = '\0';
			page2[0] = page[0];
			pagepar = atoi(page2) - 1;

			return (pagepar)*4 + 2;
		}
		if ((mouse_press(460, 150 + 60 * 2,
520, 150 + 40 + 60 * 2) == 1) && ((h->lenpar -
(pagepar * 4)) > 2))
		{
			int pagepar;
			char page2[1];
			page2[1] = '\0';
			page2[0] = page[0];
			pagepar = atoi(page2) - 1;

			return (pagepar)*4 + 3;
		}
		if ((mouse_press(460, 150 + 60 * 3,
520, 150 + 40 + 60 * 3) == 1) && ((h->lenpar -
(pagepar * 4)) > 3))
		{
			int pagepar;
			char page2[1];
			page2[1] = '\0';
			page2[0] = page[0];
			pagepar = atoi(page2) - 1;

			return (pagepar)*4 + 4;
		}

		// enter
		if (mouse_press(0, 0, 40, 30) == 0 ||
mouse_press(0, 450, 40, 480) == 0 ||
mouse_press(260, 395, 310, 415) == 0 ||
mouse_press(330, 395, 380, 415) == 0 ||
mouse_press(510, 80, 580, 125) == 0)
		{
			MouseS = 0;
		}
		if (mouse_press(0, 0, 40, 30) == 2 ||
mouse_press(0, 450, 40, 480) == 2 ||
mouse_press(260, 395, 310, 415) == 2 ||
mouse_press(330, 395, 380, 415) == 2 ||
mouse_press(510, 80, 580, 125) == 2)
		{
			MouseS = 1;
		}
		// quit
		if (mouse_press(0, 0, 40, 30) == 1)
		{
			exit(0);
			}
			// last
			if (mouse_press(0, 450, 40, 480) == 1)
			{
				return -1;
			}
		}
	}

	void past02_screen(int par)
	{
		int i;
		clrmous(MouseX, MouseY);
		cleardevice();
		setbkcolor(WHITE);
		settextstyle(0, 0, 3);
		settextjustify(1, 1);
		setcolor(BLUE);
		outtextxy(320, 45, h->parameter[par - 1].name);
		quit();

		setfillstyle(1, LIGHTGRAY);
		bar(50, 100, 590, 420);
		puthz(80, 120, "产区：", 32, 32, BLUE);
		if (h->parameter[par - 1].place == 'a')
		{
			puthz(180, 120, "新疆", 32, 32, BLUE);
		}
		else if (h->parameter[par - 1].place == 'b')
		{
			puthz(180, 120, "黄河", 32, 32, BLUE);
		}
		else if (h->parameter[par - 1].place == 'c')
		{
			puthz(180, 120, "长江", 32, 32, BLUE);
		}

		puthz(80, 120 + 60, "土地形状：", 32, 32, BLUE);
		if (h->parameter[par - 1].shape == 'a')
		{
			puthz(240, 120 + 60, "矩形", 32, 32, BLUE);
		}
		else if (h->parameter[par - 1].shape == 'b')
		{
			puthz(240, 120 + 60, "圆形", 32, 32, BLUE);
		}
		else if (h->parameter[par - 1].shape == 'c')
		{
			puthz(240, 120 + 60, "多边形", 32, 32, BLUE);
		}
		else if (h->parameter[par - 1].shape == 'd')
		{
			puthz(240, 120 + 60, "自定义图形", 32, 32, BLUE);
		}

		puthz(80, 120 + 60 * 2, "收割机类型：", 32, 32, BLUE);
		if (h->parameter[par - 1].type == 'a')
		{
			puthz(260, 120 + 60 * 2, "垂直式收割机", 32, 32, BLUE);
		}
		else if (h->parameter[par - 1].type == 'b')
		{
			puthz(260, 120 + 60 * 2, "水平式收割机", 32, 32, BLUE);
		}

		puthz(80, 120 + 60 * 3, "面积：", 32, 32, BLUE);
		settextjustify(0, 2);
		outtextxy(180, 130 + 60 * 3, h->parameter[par - 1].S);
		puthz(320, 120 + 60 * 3, "公顷", 32, 32, BLUE);

		setfillstyle(1, YELLOW);
		for (i = 0; i < 4; i++)
		{
			bar(500, 118 + 60 * i, 550, 118 + 40 + 60 * i);
			puthz(510, 130 + 60 * i, "修改", 16, 16, BLUE);
		}
		setfillstyle(1, CYAN);
		bar(80, 120 + 60 * 4, 150, 160 + 60 * 4);
		puthz(83, 125 + 60 * 4, "返回", 32, 32, BLUE);
```

```
    bar(400, 120 + 60 * 4, 550, 160 + 60 * 4);
    puthz(408, 125 + 60 * 4, "修改名字", 32, 32, BLUE);

    setfillstyle(1, RED);
    bar(400 + 40, 30, 480 + 40, 80);
    setcolor(BLACK);
    rectangle(399 + 40, 29, 481 + 40, 81);
    puthz(409 + 40, 38, "删除", 32, 32, BLUE);
}

int past02(int par)
{
    int i;
    past02_screen(par);

    for (;;)
    {
        newmouse(&MouseX, &MouseY, &press);

        // 删除按钮
        if (mouse_press(400 + 40, 30, 480 + 40, 80) == 1)
        {
            deletepar(par);
            return 1;
        }
        // 改名字
        if (mouse_press(400, 120 + 60 * 4, 550, 160 + 60 * 4) == 1)
        {
            changeparname(par);
            return -1;
        }
        // 四个修改按钮
        if (mouse_press(500, 118 + 60 * 0, 550, 118 + 40 + 60 * 0) == 1) // place
        {
            changeplace(par);
            return -1;
        }
        if (mouse_press(500, 118 + 60 * 1, 550, 118 + 40 + 60 * 1) == 1) // shape
        {
            changeshape(par);
            return -1;
        }
        if (mouse_press(500, 118 + 60 * 2, 550, 118 + 40 + 60 * 2) == 1) // type
        {
            changetype(par);
            return -1;
        }
        if (mouse_press(500, 118 + 60 * 3, 550, 118 + 40 + 60 * 3) == 1) // S
        {
            changeS(par);
            return -1;
        }

        // quit,last
        if (mouse_press(0, 0, 40, 30) == 0 || mouse_press(80, 120 + 60 * 4, 150, 160 + 60 * 4) == 0)
        {
            MouseS = 0;
        }
        if (mouse_press(0, 0, 40, 30) == 2 || mouse_press(80, 120 + 60 * 4, 150, 160 + 60 * 4) == 2)
        {
            MouseS = 1;
        }
        if (mouse_press(0, 0, 40, 30) == 1)
        {
            exit(0);
        }

        // last
        if (mouse_press(80, 120 + 60 * 4, 150, 160 + 60 * 4) == 1)
        {
            return 1;
        }
        delay(15);
    }
}
void past()
{
    int act, i;

    for (;;)
```

```
        {
            act = past01();
            if (act == -1) // 返回主界面
            {
                return;
            }
            else if (act == -2) // 搜索函数
            {
                continue;
            }
        there:
            if (past02(act) == -1)
            {
                goto there;
            }
        }
    }
```

10、 REGISTER.C

```
#include "COMMON.H"
#include "REGISTER.H"
#include "PARAMETE.H"
#include "LOGFUN.H"

void register_screen()
{
    cleardevice();
    setbkcolor(WHITE);
    setfillstyle(1, LIGHTBLUE);
    bar(80, 70, 560, 500);

    setfillstyle(1, LIGHTGRAY);
    puthz(260, 30, "注册账号", 32, 32, BLUE);
    bar(120, 105, 530, 150); // 账号
    puthz(123, 110, "账号", 32, 32, BLUE);
    bar(120, 180, 530, 230); // 密码
    puthz(123, 190, "密码", 32, 32, BLUE);
    bar(120, 260, 530, 310); // 确认密码
    puthz(123, 270, "确认密码", 32, 32, BLUE);
    bar(120, 340, 530, 390); // 电话号码
    puthz(123, 350, "电话号码", 32, 32, BLUE);

    setfillstyle(1, DARKGRAY);
    bar(120, 420, 280, 460); // 返回
    puthz(160, 423, "返回", 32, 32, RED);
    bar(360, 420, 520, 460); // 确认
    puthz(400, 423, "确认", 32, 32, RED);

    setfillstyle(1, WHITE);
    bar(260, 340, 530, 390);
    bar(260, 260, 530, 310);
    bar(220, 180, 530, 230);
    bar(220, 100, 530, 150);
}

void log_register()
{
    // 界面
    INPUT username = {220, 100, 530, 150, "", 10, 0, 1};
    INPUT password = {220, 180, 530, 230, "", 10, 0, 1};
    INPUT realpassword = {260, 260, 530, 310, "", 10, 0, 1};
    INPUT phonenumber = {260, 340, 530, 390, "", 11, 0, 1};

    register_screen();

    for (;;)
    {
        newmouse(&MouseX, &MouseY, &press);
        input_s(260, 260, &realpassword, 16, 1);
        if (input_s(223, 100, &username, 16, 0) == 1)
        {
            setfillstyle(1, CYAN);
            bar(170, 30, 470, 72);
            puthz(180, 32, "账号名最多十位字符", 32, 32, BLUE);
            delay(800);
            setfillstyle(1, 0);
            bar(170, 30, 470, 72);
            puthz(260, 30, "注册账号", 32, 32, BLUE);
        }
        if (input_s(220, 180, &password, 16, 1) == 1)
        {
                setfillstyle(1, CYAN);
                bar(170, 30, 470, 72);
                puthz(180, 32, "密码最多十
位字符", 32, 32, BLUE);
                delay(800);
                setfillstyle(1, 0);
                bar(170, 30, 470, 72);
                puthz(260, 30, "注册账号", 32,
32, BLUE);
            }
            if ((input_s(263, 340,
&phonenumber, 16, 0))==1)
            {
                setfillstyle(1, CYAN);
                bar(170, 30, 470, 72);
                puthz(180, 32, "请输入十一
位号码", 32, 32, BLUE);
                delay(800);
                setfillstyle(1, 0);
                bar(170, 30, 470, 72);
                puthz(260, 30, "注册账号", 32,
32, BLUE);
            }
            if (mouse_press(120, 420, 280, 460)
== 1)
            {
                return;
            }
            if ((mouse_press(360, 420, 520,
460) == 1) &&
                (username.string[0] != '\0')
&& (phonenumber.string[0] != '\0') &&
                (password.string[0] != '\0') &&
(strcmp(password.string, realpassword.string) ==
0))
            {
                if
(username_same(username.string,
phonenumber.string) == 0)
                {
                    wr_user(username.string,
password.string, phonenumber.string);
                    return;
                }
            }
            delay(15);
        }
}
```

11、 RESET.C

```
    #include "COMMON.H"
    #include "LOGIN.H"
    #include "PARAMETE.H"
    #include "RESET.H"
    #include "LOGFUN.H"


    void reset_screen()
    {
        cleardevice();
        setbkcolor(WHITE);
        setfillstyle(1,LIGHTBLUE);
        bar(80,80,560,500);

        setfillstyle(1,LIGHTGRAY);
        puthz(260,30,"忘记密码",32,32,BLUE);
        bar(120, 140, 520, 200);
//账号
        puthz(123,150,"账号",32,32,BLUE);
        bar(120, 220, 520, 280);
//电话号码
        puthz(123,230,"电话号码",32,32,BLUE);
        bar(120, 300, 520, 360);
//新密码
        puthz(123,310,"新密码",32,32,BLUE);

        setfillstyle(1,DARKGRAY);
        bar(120, 420, 280, 460);
//返回
        puthz(160,423,"返回",32,32,BLUE);
        bar(360, 420, 520, 460);
//确认
        puthz(400,423,"确认",32,32,BLUE);

        setfillstyle(1,WHITE);
        bar(220, 140, 520, 200);
        bar(260, 220, 520, 280);
        bar(220, 300, 520, 360);
    }


    void reset()
    {
        INPUT username = {220, 140, 520,
200,"",10,0,1};
        INPUT phonenumber = {260, 220, 520,
280,"",11,0,1};
            INPUT newpassword = {220, 300, 520,
360,"",10,0,1};

            reset_screen();

            for(;;)
            {

newmouse(&MouseX,&MouseY,&press);
                  input_s(223, 140, &username, 12 ,
0);
                  input_s(263, 220, &phonenumber,
12 , 1);
                  input_s(223, 300, &newpassword,
12 , 1);
                  if(mouse_press(120,   420,   280,
460)==1)
                  {
                        return;
                  }
                  if(mouse_press(360,   420,   520,
460)==1)
                  {

if(changepassword(username.string,
newpassword.string, phonenumber.string)==1)
                        {
                              return;
                        }
                  }
                  delay(15);
            }
}
```

12、 START.C

```
      #include "START.H"
      #include "COMMON.H"
      #include "PARAMETE.H"
      #include "PAST.H"
      #include "DSTART.H"
      #include "LOGFUN.H"
      #include "HOME.H"
      int delaytime=20;
      // int main()
      // {
      //      int gd=VGA,gm=VGAHI;
      //
  initgraph(&gd,&gm,"..\\borlandc\\bgi");
      //      setbkcolor(WHITE);
      //      h=(user*)malloc(sizeof(user));// 登 录
的用户
      //      //select03(&(h->parameter[1]));
      //      strcpy(h->parameter[0].name,"abc");
      //      strcpy(h->parameter[1].name,"acc");
      //      //strcpy(&h->parameter[0].type,"a");
      //      h->parameter[1].type='a';
      //      h->parameter[1].place='a';
      //      //strcpy(&h->parameter[0].shape,"a");
      //      h->parameter[1].shape='a';
      //      strcpy(h->parameter[1].S,"500");
      //      start();
      //      delay(5000);
      //      return 0;
      // }
      // int main()
      // {
      //      int gd=VGA,gm=VGAHI;
      //      int
gd=VGA,gm=VGAHI,start_x=400,start_y=100,des_
x=100,des_y=300;
      //
  initgraph(&gd,&gm,"..\\borlandc\\bgi");
      //      setbkcolor(WHITE);
      //      h=(user*)malloc(sizeof(user));// 登 录
的用户
      //      strcpy(h->parameter[0].name,"abc");
      //      strcpy(h->parameter[1].name,"acc");
      //      //strcpy(&h->parameter[0].type,"a");
      //      h->parameter[1].type='a';
      //      h->parameter[1].place='a';
      //      //strcpy(&h->parameter[0].shape,"a");
      //      h->parameter[1].shape='b';
      //      strcpy(h->parameter[1].S,"300");
      //      start();
      //      delay(5000);
      //      return 0;
      // }
      // int main()
      // {
      //      int gd=VGA,gm=VGAHI,x=300,y=200;
      //
  initgraph(&gd,&gm,"..\\borlandc\\bgi");
      //      setbkcolor(BROWN);
      //      mouseinit();
      //      picker_anime(50,50,&x,&y,10,1);
      //      delay(5000);
      //      return 0;
      // }

      void start()
      {
        int i=0,time=0,co_type=0;
        double space=0,harvest=0;
        delaytime=20;
        // for(i=0;i<10;i++)
        // {
        //
  if(strcmp(h->parameter[i].name,"\0")==0)
        //    {
        //          break;
        //    }
        // }
        // i--;
        i=choosepar();
        i--;
        delay(500);
        space=atoi(h->parameter[i].S);
        if(space>32767||space<=0)
               space=32767;
        if(space<500)
               space=500;
        time=space/pick_ph;
        if(time<tra_time)
               time=tra_time;
        if(time>tracktor_num_max*tra_time)
               time=tracktor_num_max*tra_time;
        switch(h->parameter[i].place)
        {
               case 'a':
               {

  harvest=space*Xinjinag_har*(1+(rand()%8)/100)
;
                      co_type=0;
                      break;
               }
               case 'b':
               {

  harvest=space*Huanghe_har*(1+(rand()%8)/100
);
                      co_type=1;
                      break;
               }
               case 'c':
               {

  harvest=space*Chnagjiang_har*(1+(rand()%8)/1
00);
                      co_type=2;
                      break;
               }
        }
        if(harvest==0)
               harvest=3;

        switch(h->parameter[i].type)
        {
               case 'a':
               {
                      //draw_simu01(time);
                      switch(h->parameter[i].shape)
                      {
                             case 'a':
                             {

  start_ainime01(0,space,time);
                                    break;
                             }
                             case 'b':
                             {

  cal_tracktor_circle(0,space,time);
                                    break;
                             }
                             case 'c':
                             {

  init_field02(h->parameter[i].x,h->parameter[i].y,
&(h->parameter[i].lenxy),0,time);
                                    break;
                             }
                             case 'd':
                             {
```

```
  init_field03(h->parameter[i].x,h->parameter[i].y,
&(h->parameter[i].lenxy),0,time);
                            break;
                        }
                    }
                    if(mode!=3)
                        return;
                    pick_finish(harvest,co_type);
                    break;
                }

                case 'b':
                {
                    switch(h->parameter[i].shape)
                    {
                        case 'a':
                        {

  start_ainime01(1,space,time*0.8);
                            break;
                        }
                        case 'b':
                        {

  cal_tracktor_circle(1,space,time*0.8);
                            break;
                        }
                        case 'c':
                        {

  init_field02(h->parameter[i].x,h->parameter[i].y,
&h->parameter[i].lenxy,1,time*0.8);
                            break;
                        }
                        case 'd':
                        {

  init_field03(h->parameter[i].x,h->parameter[i].y,
&h->parameter[i].lenxy,1,time*0.8);
                            break;
                        }
                    }
                    if(mode!=3)
                        return;
                    pick_finish(harvest*0.8,co_type);
                    break;
                }
```

```
        default:
            break;
    }


    // switch (h->parameter[i].place)
    // {
    //     case 'a':
    //     {
    //         h->here[k].total[0]+=harvest;
    //         break;
    //     }
    //     case 'b':
    //     {
    //         h->here[k].total[1]+=harvest;
    //         break;
    //     }
    //     case 'c':
    //     {
    //         h->here[k].total[2]+=harvest;
    //         break;
    //     }

    //     default:
    //         break;
    // }
    return;
}


//draw the process of harvest in animition
void draw_simu01(int time)
{
  char str[8];
  itoa(time,str,10);
  clrmous(MouseX,MouseY);
      cleardevice();
      setbkcolor(WHITE);
  puthz(150,30,"采摘完成共需",32,32,BLUE);
  settextstyle(3,0,4);
  setcolor(RED);
  settextjustify(0,2);
  outtextxy(350,28,str);
      puthz(450,30,"小时",32,32,BLUE);
  init_based_field();
      //outtextxy()
    quit();
    skip();
        // mouseinit();
    // start_ainime01(0,x,y,num);
    // pick_finish(temp);
    // tal[c_t]+=temp;
    // if(tal[c_t]>ware_full||tal[c_t]<0)
    // {
    //     tal[c_t]=ware_full;
    // }
    // for(;;)
    // {
    //
 newmouse(&MouseX,&MouseY,&press);
    //     press_start();
    //     delay(20);
    // }

    //bmp_convert(".\\photo\\map.bmp",".\\p
hoto\\map.dbm");
    //show_dbm(5,100,".\\photo\\map.dbm");
    //getchar();
    //closegraph();
    return;
  }

  void init_based_field()
  {
    setfillstyle(1,BROWN);
    bar(0,80,625,480);
    return;
  }

  void draw_copak(int x,int y)
  {
    if(x<x_start||y<y_start)
        return;
    setfillstyle(1,WHITE);
    bar(x,y,x+tracktor_w,y+co_pak_w);
  }

  //start playing the picking video
  void    start_ainime01(int    t_trac,double
space,int time)
  {
    int i,x_p,y_p,flag,out,x,y,num,xy[2],\
    cal_time=0,des_x[2*tracktor_num_max],de
s_y[2*tracktor_num_max];
    double temp_x,temp_y;
    temp_y=sqrt(space/55*32)*10;
    temp_x=temp_y*55/32;
    x=temp_x,y=temp_y,num=time/tra_time;
    if(x>x_max)
        x=x_max;
    if(y>y_max)
        y=y_max;
    if(num==0)
        num=1;
    for(i=0;i<num;i++)
    {
        des_x[i]=600;
        des_y[i]=0;
    }
    i=0,x_p=x_start,y_p=y_start+y-40,flag=0,ou
t=0;
    select_setoff01(xy,x_start+x,y_start+y);
    draw_simu01(time);
    clrmous(MouseX,MouseY);
    draw_setoff(xy);
    init_field(x,y);
    x/=num;
    if(t_trac==0)
    {
        draw_setoff(xy);

 tracktor_set_off(xy[0],xy[1],x_start,y_start+y,x,n
um);
        earth_fill01(x_p, y_p);
        init_tracktor01_f(x_p, y_p);
        while (1)
        {

 if(cal_time>0&&cal_time>=pick_bar)
            {
                cal_time=-1;
                for(i=0;i<num;i++)
                {
                    if(flag==0)
                    {

 draw_copak(x_p+i*x,y_p+tracktor_l+co_pak_w);
                        des_x[i]=x_p+i*x;

 des_y[i]=y_p+tracktor_l+co_pak_w;
```

```
                        }
                        else
                        {
  draw_copak(x_p+i*x,y_p-2*co_pak_w);
                              des_x[i]=x_p+i*x;
  des_y[i]=y_p-2*co_pak_w;
                        }
                  }
            }
            if(cal_time>=0)
                  cal_time++;
            if (x_p >= 50 + x)
            {
                  break;
            }
            for (i = 0; i < num; i++)
            {
                  newmouse(&MouseX,
&MouseY, &press);
                  if (flag == 0)
                  {
                        if (y_p == 120 + y - 45)
                        {
                              earth_fill03(x_p + i
* x, y_p - 4);
                        }
                        if (y_p >= 120 + 7)
                        {
                              earth_fill01(x_p + i
* x, y_p);
  init_tracktor01_f(x_p + i * x, y_p);
                        }
                        if (y_p <= 120 + 7)
                        {
                              earth_fill03(x_p + i
* x, 120);
                        }
                  }
                  else
                  {
                        if (y_p == 120 + 7)
                        {
                              earth_fill01(x_p + i
* x, y_p - 7);
                        }
                        if (y_p <= 120 + y - 45)
                        {
                              earth_fill02(x_p + i
* x, y_p);
  init_tracktor01_b(x_p + i * x, y_p);
                        }
                        if (y_p >= 120 + y - 45)
                        {
                              earth_fill03(x_p + i
* x, 120 + y - 49);
                        }
                  }
                  delay(delaytime / num);
            }
            if (flag == 0)
            {
                  y_p--;
                  if (y_p <= 120 + 7)
                  {
                        for (i = 0; i < num; i++)
                              earth_fill03(x_p + i
* x, 120);
                        flag = 1;
                        x_p += 25;
                  }
            }
            else
            {
                  y_p++;
                  if (y_p >= 120 + y - 45)
                  {
                        for (i = 0; i < num; i++)
                              earth_fill03(x_p + i
* x, 120 + y - 49);
                        flag = 0;
                        x_p += 25;
                  }
            }
            out                                    =
pressed_anime(x_start,y_start,x * num, y);
            if (out != 0)
            {
                  // for (i = 1; i <= num; i++)
                  // {
                  //
  init_tracktor01_f(x_start + i * x - x % 25, y_start + 
7);
                        // }
                        break;
                    }
                    for(i=0;i<num;i++)
                    {

  draw_copak(des_x[i],des_y[i]);
                    }
                    // delay(15*num);
                }

  tracktor_return(xy[0],xy[1],x_start+x,y_start,x,nu
m);

  picker_anime(xy[0],xy[1],des_x,des_y,x,num);
                return;
                // for(i=1;i<=num;i++)
                // {
                //    if (flag == 1)
                //            init_tracktor01_f(x_start + i * 
x - x % 25, y_start + 7);
                //    else
                //            init_tracktor01_b(x_start + i 
* x - x % 25, y_start + y - 45);
                // }
            }
            else
            {
                clrmous(MouseX,MouseY);
                draw_setoff(xy);

  tracktor_set_off0(xy[0],xy[1],x_start,y_start+y,x,
num);
                earth_fill01(x_p, y_p);
                init_tracktor02_f(x_p, y_p);
                while (1)
                {

  if(cal_time>0&&cal_time>=pick_bar)
                    {
                        cal_time=-1;
                        for(i=0;i<num;i++)
                        {
                            if(flag==0)
                            {

  draw_copak(x_p+i*x,y+tracktor_l+45);
                                des_x[i]=x_p+i*x;

  des_y[i]=y+tracktor_l+45;
                            }
                            else
                            {

  draw_copak(x_p+i*x,y-45-co_pak_w);
                                des_x[i]=x_p+i*x;

  des_y[i]=y-45-co_pak_w;
                            }
                        }
                    }
                    if(cal_time>=0)
                        cal_time++;
                    if (x_p >= 50 + x)
                    {
                        break;
                    }
                    for (i = 0; i < num; i++)
                    {
                        newmouse(&MouseX, 
&MouseY, &press);
                        if (flag == 0)
                        {
                            if (y_p == 120 + y - 45)
                            {
                                earth_fill03(x_p + i 
* x, y_p - 4);
                            }
                            if (y_p >= 120 + 7)
                            {
                                earth_fill01(x_p + i 
* x, y_p);

  init_tracktor02_f(x_p + i * x, y_p);
                            }
                            if (y_p <= 120 + 7)
                            {
                                earth_fill03(x_p + i 
* x, 120);
                            }
                        }
                        else
                        {
                                if (y_p == 120 + 7)
                                {
                                        earth_fill01(x_p + i
* x, y_p - 7);
                                }
                                if (y_p <= 120 + y - 45)
                                {
                                        earth_fill02(x_p + i
* x, y_p);

  init_tracktor02_b(x_p + i * x, y_p);
                                }
                                if (y_p >= 120 + y - 45)
                                {
                                        earth_fill03(x_p + i
* x, 120 + y - 49);
                                }
                        }
                        delay(delaytime/num);
                }
                if (flag == 0)
                {
                        y_p--;
                        if (y_p <= 120 + 7)
                        {
                                for (i = 0; i < num; i++)
                                        earth_fill03(x_p + i
* x, 120);
                                flag = 1;
                                x_p += 25;
                        }
                }
                else
                {
                        y_p++;
                        if (y_p >= 120 + y - 45)
                        {
                                for (i = 0; i < num; i++)
                                        earth_fill03(x_p + i
* x, 120 + y - 49);
                                flag = 0;
                                x_p += 25;
                        }
                }
                out                                    =
pressed_anime(x_start,y_start,x * num, y);
                        if (out != 0)
                        {
                                // for (i = 1; i <= num; i++)
                                // {
                                //
  init_tracktor02_f(x_start + i * x - x % 25, y_start +
7);
                                // }
                                break;
                        }
                        for(i=0;i<num;i++)
                        {

  draw_copak(des_x[i],des_y[i]);
                        }
                        // delay(15*num);
                }

  tracktor_return0(xy[0],xy[1],x_start+x,y_start,x,n
um);

  picker_anime(xy[0],xy[1],des_x,des_y,x,num);
                return;
                // for(i=1;i<=num;i++)
                // {
                //      if (flag == 1)
                //              init_tracktor02_f(x_start + i *
x - x % 25, y_start + 7);
                //      else
                //              init_tracktor02_b(x_start  +  i
* x - x % 25, y_start + y - 45);
                // }
        }
}

//add the press moudules in start page
void press_start(int *bk)
{
  if   (mouse_press(0,   0,   40,   30)   ==   0   ||
mouse_press(265, 350, 365, 410) == 0)
  {
        MouseS = 0;
  }
  if   (mouse_press(0,   0,   40,   30)   ==   2   ||
mouse_press(265, 350, 365, 410) == 2)
  {
        MouseS = 1;
        }
        if (mouse_press(0, 0, 40, 30) == 1)
        {
            wr_h();
            free(h);
            exit(0);
        }
        else if (mouse_press(265, 350, 365, 410) ==
1)
        {
            //draw_home01();
            mode=1;
            mode1=0;
            *bk=1;
            return;
        }
    }

    // add press moudules in video page
    int  pressed_anime(int  x_sta,int  y_sta,  int
x_des,int y_des)
    {
        int re;
        re=0;
        if(MouseX>=x_sta&&MouseY>=y_sta&&Mo
useX<=x_sta+x_des&&MouseY<=y_sta+y_des)
        {
            clrmous(MouseX, MouseY);
        }
        if  (mouse_press(0,  0,  40,  30)  ==  0  ||
mouse_press(585, 450, 625, 480) == 0)
        {
            MouseS = 0;
        }
        if  (mouse_press(0,  0,  40,  30)  ==  2  ||
mouse_press(585, 450, 625, 480) == 2)
        {
            MouseS = 1;
        }
        //                          if(*times>0.5&&
bioskey(0)==p_Up_arrow)
        // {
        //    *times-=0.1;
        // }
        //
if(*times<2&&bioskey(0)==p_Donw_arrow)
        // {
        //    *times+=0.1;
        // }
        if (mouse_press(0, 0, 40, 30) == 1)
        {
            re=1;
            wr_h();
            free(h);
            exit(0);
        }
        if (mouse_press(585, 450, 625, 480) == 1)
        {
            anime_skip_result(x_sta,
y_sta,x_des,y_des);
            re = 1;
        }
        return re;
    }

    // show after picking
    void pick_finish(int count,int co_type)
    {
        char str[10];
        int bk=0,re=0;
        itoa(count,str,10);
        setfillstyle(1,WHITE);
        bar(160,170,470,340);
        setcolor(RED);
        setlinestyle(0, 0, 3);
        rectangle(165, 175, 465, 335);

        puthz(210,190,"采摘完成",32,32,BLUE);
        puthz(210,230,"共计：",32,32,BLUE);
        settextstyle(3,0,4);
        puthz(380,230,"吨",32,32,BLUE);
        settextjustify(0,2);
        outtextxy(300,225,str);
        switch(co_type)
        {
        case 0:
        {
            puthz(210, 270, "种类：长绒棉", 32, 32,
BLUE);
            break;
        }
        case 1:
        {
            puthz(210, 270, "种类：细绒棉", 32, 32,
BLUE);
            break;
        }
        case 2:
        {
            puthz(210, 270, "种类：粗绒棉", 32, 32,
BLUE);
            break;
        }
        default:
            break;
        }

        setfillstyle(1,YELLOW);
        bar(265,350,365,410);
        puthz(278,360,"入库",32,32,RED);
        while(1)
        {
            if(bk!=0)
                break;

newmouse(&MouseX,&MouseY,&press);
            press_start(&bk);
        }
        clrmous(MouseX,MouseY);
        warehouse_list(h->here);
        delay(500);
        while(1)
        {
            if(re!=0)
            {
                break;
            }

newmouse(&MouseX,&MouseY,&press);
            press_warelist(&re);
        }
        h->here[k].total[co_type]+=count;
        if(h->here[k].total[co_type]<0)
        {
            h->here[k].total[co_type]=ware_full;
        }
        mode=1;
        return;
    }

    // moudules which skip the video and show
the result
    void anime_skip_result(int x_sta,int y_sta, int
x_des,int y_des)
    {
      //int i = 0;
      setfillstyle(1, BROWN);
      setcolor(WHITE);
      bar(x_sta, y_sta, x_des, y_des);
      // for (i = 0; i < x*y*0.05; i++)
      // {
      //    int x_t = rand() % x, y_t = rand() % y;
      //    line(x_start + x_t, y_start + y_t, x_start
+ x_t, y_start + y_t);
      // }
    }

    // initialize the cotton field
    void init_field(int x, int y)
    {
      int i,j;
      setfillstyle(1,WHITE);
      bar(x_start,y_start,x_start+x,y_start+y);
      setcolor(BROWN);
      setlinestyle(0, 0, 1);
      line(50, 120, 50 + x, 120);
      line(50, 120 + y, 50 + x, 120 + y);
      for (i = 50; i <= 50 + x; i += 25)
      {
          int temp = 0;
          for (j = 120; j <= 120 + y; j += 1)
          {
              temp = rand() % 2;
              line(i + temp, j, i + temp, j);
          }
      }
      for (i = 0; i < 5000; i++)
      {
          int x_temp = 0, y_temp = 0;
          x_temp = rand() % x;
          y_temp = rand() % y;
          line(50 + x_temp, 120 + y_temp, 50 +
x_temp, 120 + y_temp);
      }
    }


    void select02(struct Parameter *abc)
    {
     int x[point_max], y[point_max], flag = 0,
back = 0,i;
     cleardevice();
     setbkcolor(WHITE);
     setcolor(GREEN);
     setlinestyle(0, 0, 3);
     rectangle(x_start, y_start, x_start + x_max,
y_start + y_max);
    puthz(170, 30, "请依次在框内选点", 32, 32,
BLUE);
     setfillstyle(1, GREEN);
     bar(200, 70, 280, 115);
     puthz(207, 76, "开始", 32, 32, WHITE);
     setfillstyle(1, RED);
     bar(320, 70, 400, 115);
     puthz(327, 76, "完成", 32, 32, WHITE);
     settextstyle(3, 0, 4);
     quit();

     while (1)
     {
         if (back != 0)
         {
             break;
         }
         newmouse(&MouseX,        &MouseY,
&press);
         press_select02(x, y, &flag, &back);
         delay(20);
     }
     abc->lenxy=flag;
     for (i=0;i<flag;i++)
     {
         abc->x[i]=x[i];
         abc->y[i]=y[i];
     }
    }

    void press_select02(int *x, int *y, int *flag,
int *back)
    {
     if (mouse_press(0, 0, 40, 30) == 0 ||
mouse_press(200, 70, 280, 115) == 0 ||
mouse_press(320, 70, 400, 115) == 0)
     {
         MouseS = 0;
     }
     if (mouse_press(0, 0, 40, 30) == 2 ||
mouse_press(200, 70, 280, 115) == 2 ||
mouse_press(320, 70, 400, 115) == 2)
     {
         MouseS = 1;
     }
     if (mouse_press(0, 0, 40, 30) == 1)
     {
         mode=0;
     }
     if (mouse_press(200, 70, 280, 115) == 1)
     {
         pick_points(x, y, flag);
         return;
     }
     //
if(mouse_press(200,70,280,115)==1&&(*flag)!=0)
     // {
     //     pick_points(x,y,flag);
     //     return;
     // }
     if (mouse_press(320, 70, 400, 115) == 1)
     {
         //init_field02(x, y, flag);
         *back = 1;
     }
    }

    void pick_wait()
    {
     int i;
     for (i = 0; i < 30; i++)
     {
         newmouse(&MouseX,        &MouseY,
&press);
         delay(10);
     }
    }

    // void pick_start(int *x,int *y)
    // {
    //     while(1)
    //     {
    //
  newmouse(&MouseX,&MouseY,&press);
    //
  if(mouse_press(x_start,y_start,x_start+x_max,y_
start+y_max)==1)
       //              {
       //                     *x=MouseX;
       //                     *y=MouseY;
       //                     break;
       //              }
       //              delay(20);
       //       }
       //       clrmous(MouseX,MouseY);
       //       setfillstyle(1,RED);
       //       bar((*x)-5,(*y)-5,(*x)+5,(*y)+5);
       //       setfillstyle(1,GREEN);
       //       bar(200,70,280,115);
       //       puthz(207,76,"继续",32,32,WHITE);
       //       return;
       // }

       void pick_points(int *x, int *y, int *flag)
       {
         clrmous(MouseX, MouseY);
         setfillstyle(1, WHITE);
         bar(200, 70, 280, 115);
         // mouseinit();
         while (*flag == 0)
         {
                newmouse(&MouseX,        &MouseY,
&press);
                if    (mouse_press(x_start,     y_start,
x_start + x_max, y_start + y_max) == 1)
                {
                       *x = MouseX;
                       *y = MouseY;
                       clrmous(MouseX, MouseY);
                       setfillstyle(1, GREEN);
                       bar(x[*flag]  -  5,  y[*flag]  -  5,
x[*flag] + 5, y[*flag] + 5);
                       (*flag)++;
                }
                delay(20);
         }
         while ((*flag) < point_max)
         {
                newmouse(&MouseX,        &MouseY,
&press);
                if  (MouseX  !=  x[(*flag)  -  1]  &&
mouse_press(x_start,  y_start,  x_start  +  x_max,
y_start + y_max) == 1)
                {
                       x[*flag] = MouseX;
                       y[*flag] = MouseY;
                       clrmous(MouseX, MouseY);
                       setfillstyle(1, RED);
                       bar(x[*flag]  -  5,  y[*flag]  -  5,
x[*flag] + 5, y[*flag] + 5);
                       setcolor(BLUE);
                       line(x[(*flag)  -  1],  y[(*flag)  -  1],
x[*flag], y[*flag]);
                       (*flag)++;
                }
                if  (mouse_press(0,  0,  40,  30)  ==  0  ||
mouse_press(320, 70, 400, 115) == 0)
                {
                       MouseS = 0;
                }
                if  (mouse_press(0,  0,  40,  30)  ==  2  ||
mouse_press(320, 70, 400, 115) == 2)
                {
                       MouseS = 1;
                }
                if (mouse_press(0, 0, 40, 30) == 1)
                {
                       mode=0;
                       return;
                }
                //
if(mouse_press(200,70,280,115)==1&&(*flag)==0)
                // {
                //     pick_points(x,y,flag);
                //     return;
                // }
                //
if(mouse_press(200,70,280,115)==1&&(*flag)!=0)
                // {
                //     pick_points(x,y,flag);
                //     return;
                // }
                if  (mouse_press(320,  70,  400,  115)  ==
1)
                {
                       break;
                       // init_field02(x,y,flag);
                }
                delay(20);
```

```
    }
    return;
  }

  // void pick_points(int *x,int *y,int *flag)
  // {
  //     while((*flag)<point_max)
  //     {
  //
newmouse(&MouseX,&MouseY,&press);
  //
if(mouse_press(x_start,y_start,x_start+x_max,y_
start+y_max)==1)
  //         {
  //             x[*flag]=MouseX;
  //             y[*flag]=MouseY;
  //             break;
  //         }
  //         delay(20);
  //     }
  //     clrmous(MouseX,MouseY);
  //     setfillstyle(1,RED);
  //
bar(x[*flag]-5,y[*flag]-5,x[*flag]+5,y[*flag]+5);
  //     (*flag)++;
  //     return;
  // }

  void draw_points(int *arr,int *flag,long int
*xy_m)
  {
    int   i=0;
    setfillstyle(1, WHITE);
    fillpoly(*flag, arr);
    setcolor(BROWN);
    for (i = xy_m[0]; i <= xy_m[2]; i += 25)
    {
        int temp = 0, j;
        for (j = xy_m[1]; j <= xy_m[3]; j ++)
        {
            temp = rand() % 2;
            line(i + temp, j, i + temp, j);
        }
    }
    for (i = 0; i < ((xy_m[2] - xy_m[0]) * (xy_m[3]
- xy_m[1])) * 0.05; i++)
    {
        int x_r = rand() % (xy_m[2] - xy_m[0]),
y_r = rand() % (xy_m[3] - xy_m[1]);
        line(xy_m[0] + x_r, xy_m[1] + y_r,
xy_m[0] + x_r, xy_m[1] + y_r);
    }
    return;
  }

  /*void pick_points(int *x,int *y)
  {
    int xn,yn,i=1;
    setfillstyle(1,RED);
    setlinestyle(0,0,1);
    setcolor(BLUE);
    while(xn<x[0]-5&&xn>x[0]+5&&yn<y[0]-5&
&yn>x[0]+5)
    {
        pick_wait();
        bar(xn-5,yn-5,xn+5,yn+5);
        x[i]=xn;
        y[i]=yn;
        i++;
        while(1)
        {

newmouse(&MouseX,&MouseY,&press);

if(mouse_press(x_start,y_start,x_start+x_max,y_
start+y_max)==1)
            {
                xn=MouseX;
                yn=MouseY;
                break;
            }
            delay(15);
        }
        line(x[i],y[i],xn,yn);
    }
    return;
  }*/

  void select03(struct Parameter *abc)
  {
    int                      x[dense_points_max],
y[dense_points_max], flag = 0, back=0,i;
    cleardevice();
    setbkcolor(WHITE);
    setcolor(GREEN);
    setlinestyle(0, 0, 3);
    rectangle(x_start, y_start, x_start + x_max,
y_start + y_max);
    puthz(170, 30, "请缓慢移动鼠标勾勒图形",
32, 32, BLUE);
    setfillstyle(1, GREEN);
    bar(200, 70, 280, 115);
    puthz(207, 76, "开始", 32, 32, WHITE);
    setfillstyle(1, RED);
    bar(320, 70, 400, 115);
    puthz(327, 76, "完成", 32, 32, WHITE);
    settextstyle(3, 0, 4);
    quit();
    setfillstyle(1, RED);

    while (1)
    {
        if (back != 0)
        {
            break;
        }
        newmouse(&MouseX,        &MouseY,
&press);
        press_select03(x, y, &flag,&back);
        delay(20);
    }
    abc->lenxy=flag;
    for (i=0;i<flag;i++)
    {
        abc->x[i]=x[i];
        abc->y[i]=y[i];
    }
   }

   void press_select03(int *x,int *y,int *flag,int
*back)
   {
    if(mouse_press(0,0,40,30)==0||(mouse_pr
ess(200,70,280,115)==0&&(*flag==0))||mouse_p
ress(320,70,400,115)==0)
    {
        MouseS=0;
    }
    if(mouse_press(0,0,40,30)==2||(mouse_pr
ess(200,70,280,115)==2&&(*flag==0))||mouse_p
ress(320,70,400,115)==2)
    {
        MouseS=1;
    }
    if(mouse_press(0,0,40,30)==1)
    {
        mode=0;
        return;
    }
    if(mouse_press(200,70,280,115)==1&&(*fla
g)==0)
    {
        dense_pick(x,y,flag);
        return;
    }
    if(mouse_press(320,70,400,115)==1)
    {
        //init_field03(x,y,flag,0);
        *back=1;
        return;
    }
   }

   void dense_pick(int *x, int *y, int *flag)
   {
    clrmous(MouseX, MouseY);
    bar(200, 70, 280, 115);
    setlinestyle(0, 0, 1);
    setcolor(BLUE);
    while ((*flag) == 0)
    {
        newmouse(&MouseX,        &MouseY,
&press);
        if    (mouse_press(x_start,    y_start,
x_start + x_max, y_start + y_max) == 1)
        {
            *x = MouseX;
            *y = MouseY;
            clrmous(MouseX, MouseY);
            setfillstyle(1, GREEN);
            bar(x[*flag]  -  5,  y[*flag]  -  5,
x[*flag] + 5, y[*flag] + 5);
            (*flag)++;
            break;
        }
        delay(20);
    }
    pick_wait();
    setfillstyle(1, RED);
    while ((*flag) < dense_points_max)
    {
        if ((((*flag) >= 5 && abs(MouseX - x[0])
<= 3 && abs(MouseY - y[0]) <= 3) || MouseX <=
x_start || MouseX >= x_start + x_max || MouseY
<= y_start || MouseY >= y_start + y_max)
        {
            line(x[0],   y[0],   x[(*flag)  -  1],
y[(*flag) - 1]);
            return;
        }
        if ((MouseX - x[(*flag) - 1]) * (MouseX -
x[(*flag) - 1]) >= 4 && (MouseY - y[(*flag) - 1]) *
(MouseY - y[(*flag) - 1]) >= 4)
        {
            x[*flag] = MouseX;
            y[*flag] = MouseY;
            clrmous(MouseX, MouseY);
            setfillstyle(1, RED);
            bar(x[*flag]  -  5,  y[*flag]  -  5,
x[*flag] + 5, y[*flag] + 5);
            line(x[(*flag)  -  1],  y[(*flag)  -  1],
x[(*flag)], y[(*flag)]);
            (*flag)++;
        }
        pick_wait();
    }
    return;
  }

  void  dense_draw_points(int  *arr,  int  *flag,
long int *xy_m)
  {
    int    i;
    setfillstyle(1, WHITE);
    setcolor(WHITE);
    fillpoly(*(flag), arr);
    setcolor(BROWN);
    for (i = xy_m[0]; i <= xy_m[2]; i += 25)
    {
        int temp = 0, j;
        for (j = xy_m[1]; j <= xy_m[3]; j += 2)
        {
            temp = rand() % 2;
            line(i + temp, j, i + temp, j);
        }
    }
    for (i = 0; i < ((xy_m[2] - xy_m[0]) * (xy_m[3]
- xy_m[1])) * 0.05; i++)
    {
        int x_r = rand() % (xy_m[2] - xy_m[0]),
y_r = rand() % (xy_m[3] - xy_m[1]);
        line(xy_m[0]  +  x_r,  xy_m[1]  +  y_r,
xy_m[0] + x_r, xy_m[1] + y_r);
    }
    for (i = xy_m[0]; i <= xy_m[2]; i += 25)
    {
        int j = 0;
        for (j = xy_m[1]; j <= xy_m[3]; j++)
        {
            int temp = rand() % 2;
            line(i + temp, j, i + temp, j);
        }
    }
    return;
  }

  void  init_field02(int  *x,int  *y,int  *flag,int
type,int time)
  {
    // The meaning of elements in xy_m:
    //[0]:minest  of  x,[1]:minest  of  y,[2]:largest
of x,[3]:largest of y
    long                                   int
xy_m[4]={x_start+x_max,y_start+y_max,x_start,y
_start};
    int              i,arr[point_max              *
2],num=0,xy[2],des_x[tracktor_num_max],des_y[t
racktor_num_max];
    for (i = 0; i < (*flag); i++)
    {
        if (x[i] < xy_m[0])
        {
            xy_m[0] = x[i];
        }
        if (y[i] < xy_m[1]&&y[i]!=0)
        {
            xy_m[1] = y[i];
        }
        if (x[i] > xy_m[2])
        {
            xy_m[2] = x[i];
        }
```

```
            if (y[i] > xy_m[3])
            {
                  xy_m[3] = y[i];
            }
            arr[2 * i] = x[i];
            arr[2 * i + 1] = y[i];
      }
      for(i=0;i<num;i++)
      {
            des_x[i]=600;
            des_y[i]=0;
      }
      select_setoff02(xy,xy_m[0],xy_m[1],xy_m[2
],xy_m[3]);
      draw_simu01(time);
      // setfillstyle(1,BROWN);
      // setlinestyle(0,0,3);
      //
bar(x_start-5,y_start-5,x_start+x_max+5,y_start+y
_max+5);
      draw_points(arr,flag,xy_m);
      clrmous(MouseX,MouseY);
      draw_setoff(xy);
      //setcolor(GREEN);
      //rectangle(xy_m[0],xy_m[1],xy_m[2],xy_m
[3]);
      // setcolor(BROWN);
      //
for(i=0;i<((xy_m[2]-xy_m[0])*(xy_m[3]-xy_m[1]))/
20;i++)
      // {
      //    int
x_r=rand()%(xy_m[2]-xy_m[0]),y_r=rand()%(xy_m
[3]-xy_m[1]);
      //
  line(xy_m[0]+x_r,xy_m[1]+y_r,xy_m[0]+x_r,xy_m
[1]+y_r);
      // }
      num=time/tra_time;
      if(num==0)
            num=1;
      //delay(1000);
      if(type==0)
      {

  tracktor_set_off(xy[0],xy[1],xy_m[0],xy_m[3],(xy
_m[2]-xy_m[0])/num,num);
            clrmous(MouseX,MouseY);
            draw_setoff(xy);

  dense_init_tracktor01(x,y,flag,xy_m,num,des_x,
des_y);

  tracktor_return(xy[0],xy[1],xy_m[0]+(xy_m[2]-xy
_m[0])/num,xy_m[1],(xy_m[2]-xy_m[0])/num,nu
m);

  picker_anime(xy[0],xy[1],des_x,des_y,(xy_m[2]-x
y_m[0])/num,num);
      }
      else
      {

  tracktor_set_off0(xy[0],xy[1],xy_m[0],xy_m[3],(x
y_m[2]-xy_m[0])/num,num);
            clrmous(MouseX,MouseY);
            draw_setoff(xy);

  dense_init_tracktor02(x,y,flag,xy_m,num,des_x,
des_y);

  tracktor_return0(xy[0],xy[1],xy_m[0]+(xy_m[2]-x
y_m[0])/num,xy_m[1],(xy_m[2]-xy_m[0])/num,nu
m);

  picker_anime(xy[0],xy[1],des_x,des_y,(xy_m[2]-x
y_m[0])/num,num);
      }
    }



    void init_field03(int *x,int *y,int *flag,int
type,int time)
    {
      // The meaning of elements in xy_m:
      //[0]:minest of x,[1]:minest of y,[2]:largest
of x,[3]:largest of y
      long                                     int
xy_m[4]={x_start+x_max,y_start+y_max,x_start,y
_start};
      int     arr[dense_points_max    *    2],
i,num=0,xy[2],des_x[tracktor_num_max],des_y[tr
acktor_num_max];
      num=time/tra_time;
```

```
	if(num==0)
		num=1;
	for (i = 0; i < (*flag); i++)
	{
		if (x[i] < xy_m[0])
		{
			xy_m[0] = x[i];
		}
		if (y[i] < xy_m[1])
		{
			xy_m[1] = y[i];
		}
		if (x[i] > xy_m[2])
		{
			xy_m[2] = x[i];
		}
		if (y[i] > xy_m[3])
		{
			xy_m[3] = y[i];
		}
		arr[2 * i] = x[i];
		arr[2 * i + 1] = y[i];
	}
	for(i=0;i<num;i++)
	{
		des_x[i]=600;
		des_y[i]=0;
	}
	// setfillstyle(1,BROWN);
	// setlinestyle(0,0,3);
	//
bar(x_start-5,y_start-5,x_start+x_max+5,y_start+y
_max+5);
	select_setoff02(xy,xy_m[0],xy_m[1],xy_m[2
],xy_m[3]);
	draw_simu01(time);
	dense_draw_points(arr,flag,xy_m);
	clrmous(MouseX,MouseY);
	draw_setoff(xy);
	//setcolor(GREEN);
	//rectangle(xy_m[0],xy_m[1],xy_m[2],xy_m
[3]);
	// setcolor(BROWN);
	//
for(i=0;i<((xy_m[2]-xy_m[0])*(xy_m[3]-xy_m[1]))/
20;i++)
	// {
	//	int
x_r=rand()%(xy_m[2]-xy_m[0]),y_r=rand()%(xy_m
[3]-xy_m[1]);
	//
  line(xy_m[0]+x_r,xy_m[1]+y_r,xy_m[0]+x_r,xy_m
[1]+y_r);
	// }
	//delay(1000);
	if(type==0)
	{

  tracktor_set_off(xy[0],xy[1],xy_m[0],xy_m[3],(xy
_m[2]-xy_m[0])/num,num);
		clrmous(MouseX,MouseY);
		draw_setoff(xy);

  dense_init_tracktor01(x,y,flag,xy_m,num,des_x,
des_y);

  tracktor_return(xy[0],xy[1],xy_m[0]+(xy_m[2]-xy
_m[0])/num,xy_m[1],(xy_m[2]-xy_m[0])/num,nu
m);

  picker_anime(xy[0],xy[1],des_x,des_y,(xy_m[2]-x
y_m[0])/num,num);
	}
	else
	{

  tracktor_set_off0(xy[0],xy[1],xy_m[0],xy_m[3],(x
y_m[2]-xy_m[0])/num,num);
		clrmous(MouseX,MouseY);
		draw_setoff(xy);

  dense_init_tracktor02(x,y,flag,xy_m,num,des_x,
des_y);

  tracktor_return0(xy[0],xy[1],xy_m[0]+(xy_m[2]-x
y_m[0])/num,xy_m[1],(xy_m[2]-xy_m[0])/num,nu
m);

  picker_anime(xy[0],xy[1],des_x,des_y,(xy_m[2]-x
y_m[0])/num,num);
	}
}

void dense_init_tracktor01(int *x, int *y, int
*flag,long   int   *xy_m,   int   num,int   *des_x,int
*des_y)
    {
        long int total=0;
        int
x_d,i,tra_d[tracktor_num_max][4],tra_mark[trackt
or_num_max][4];
        x_d=xy_m[2]-xy_m[0];
        x_d/=num;
        for(i=0;i<num;i++)
        {
            int k=0;
            tra_d[i][0]=xy_m[0]+i*x_d;
            tra_d[i][2]=xy_m[0]+(i+1)*x_d;
            tra_d[i][1]=xy_m[3];
            tra_d[i][3]=xy_m[1];
            while(k<(*flag))
            {
                int target=0;

  if(x[k]>=tra_d[i][0]&&x[k]<=tra_d[i][2])
                {
                    if(target==0)
                    {
                        target=1;
                    }
                    if(y[k]<tra_d[i][1])
                    {
                        tra_d[i][1]=y[k];
                        tra_mark[i][1]=k;
                    }
                    if(y[k]>tra_d[i][3])
                    {
                        tra_d[i][3]=y[k];
                        tra_mark[i][3]=k;
                    }
                }

  if(target!=0&&(x[k]<tra_d[i][0]||x[k]>tra_d[i][2])
)
                {
                    if(y[k]<tra_d[i][1])
                    {
                        tra_d[i][1]=y[k];
                        tra_mark[i][1]=k;
                    }
                    if(y[k]>tra_d[i][3])
                    {
                        tra_d[i][3]=y[k];
                        tra_mark[i][3]=k;
                    }
                    break;
                }
                else
                {
                    k++;
                }
            }
        }
        for(i=0;i<num;i++)
        {

  if(tra_d[i][1]>y[tra_mark[i][1]-1]&&y[tra_mark[i]
[1]-1]!=0)
            {

  tra_d[i][1]=(y[tra_mark[i][1]-1]+y[tra_mark[i][1]]
)/2-30;
            }

  if(tra_d[i][1]>y[tra_mark[i][1]+1]&&y[tra_mark[i
][1]+1]!=0)
            {

  tra_d[i][1]=(y[tra_mark[i][1]+1]+y[tra_mark[i][1]
])/2-30;
            }
            if(tra_d[i][3]<y[tra_mark[i][3]-1])
            {

  tra_d[i][3]=(y[tra_mark[i][3]-1]+y[tra_mark[i][3]]
)/2+30;
            }
            if(tra_d[i][3]<y[tra_mark[i][3]+1])
            {

  tra_d[i][3]=(y[tra_mark[i][3]+1]+y[tra_mark[i][3]
])/2+30;
            }
        }
        for(i=0;i<num;i++)
        {
            total+=tra_d[i][1];
            total+=tra_d[i][3];
        }
        // for(i=0;i<num;i++)
        // {
        //    int
temp=(total-tra_d[i][1]-tra_d[i][3])/((num-1)*2);
        //    if(tra_d[i][3]<temp)
        //    {
        //
  tra_d[i][3]=(xy_m[3]+tra_d[i][3])/2+30;
        //    }
        //    if(tra_d[i][1]>temp)
        //    {
        //
  tra_d[i][1]=(xy_m[1]+tra_d[i][1])/2-30;
        //    }
        // }
        start_ainime03_01(tra_d,num,xy_m,des_x,
des_y);
      }

      void   dense_init_tracktor02(int  *x,int  *y,int
*flag,long   int*   xy_m,int   num,int   *des_x,int
*des_y)
      {
        long int total=0;
        int
x_d,i,tra_d[tracktor_num_max][4],tra_mark[trackt
or_num_max][4];
        x_d=xy_m[2]-xy_m[0];
        x_d/=num;
        for(i=0;i<num;i++)
        {
            int k=0;
            tra_d[i][0]=xy_m[0]+i*x_d;
            tra_d[i][2]=xy_m[0]+(i+1)*x_d;
            tra_d[i][1]=xy_m[3];
            tra_d[i][3]=xy_m[1];
            while(k<(*flag))
            {
                int target=0;

  if(x[k]>=tra_d[i][0]&&x[k]<=tra_d[i][2])
                {
                    if(target==0)
                    {
                        target=1;
                    }
                    if(y[k]<tra_d[i][1])
                    {
                        tra_d[i][1]=y[k];
                        tra_mark[i][1]=k;
                    }
                    if(y[k]>tra_d[i][3])
                    {
                        tra_d[i][3]=y[k];
                        tra_mark[i][3]=k;
                    }
                }

  if(target!=0&&(x[k]<tra_d[i][0]||x[k]>tra_d[i][2])
)
                {
                    if(y[k]<tra_d[i][1])
                    {
                        tra_d[i][1]=y[k];
                        tra_mark[i][1]=k;
                    }
                    if(y[k]>tra_d[i][3])
                    {
                        tra_d[i][3]=y[k];
                        tra_mark[i][3]=k;
                    }
                    break;
                }
                else
                {
                    k++;
                }
            }
        }
        for(i=0;i<num;i++)
        {

  if(tra_d[i][1]>y[tra_mark[i][1]-1]&&y[tra_mark[i]
[1]-1]!=0)
            {

  tra_d[i][1]=(y[tra_mark[i][1]-1]+y[tra_mark[i][1]]
)/2-30;
            }

  if(tra_d[i][1]>y[tra_mark[i][1]+1]&&y[tra_mark[i
][1]+1]!=0)
            {
```

```
  tra_d[i][1]=(y[tra_mark[i][1]+1]+y[tra_mark[i][1]
])/2-30;
                }
                if(tra_d[i][3]<y[tra_mark[i][3]-1])
                {

  tra_d[i][3]=(y[tra_mark[i][3]-1]+y[tra_mark[i][3]]
)/2+30;
                }
                if(tra_d[i][3]<y[tra_mark[i][3]+1])
                {

  tra_d[i][3]=(y[tra_mark[i][3]+1]+y[tra_mark[i][3]
])/2+30;
                }
            }
            for(i=0;i<num;i++)
            {
                total+=tra_d[i][1];
                total+=tra_d[i][3];
            }
            // for(i=0;i<num;i++)
            // {
            //     int
temp=(total-tra_d[i][1]-tra_d[i][3])/((num-1)*2);
            //     if(tra_d[i][3]<temp)
            //     {
            //
  tra_d[i][3]=(xy_m[3]+tra_d[i][3])/2+30;
            //     }
            //     if(tra_d[i][1]>temp)
            //     {
            //
  tra_d[i][1]=(xy_m[1]+tra_d[i][1])/2-30;
            //     }
            // }
            start_ainime03_02(tra_d,num,xy_m,des_x,
des_y);
        }

        void   start_ainime03_01(int   (*tra_d)[4],   int
num,long int *xy_m,int *des_x,int *des_y)
        {
            int
type[tracktor_num_max],x_p[tracktor_num_max]
,\
            y_p[tracktor_num_max],cal_time[tracktor_
num_max],i;
            for(i=0;i<num;i++)
            {
                type[i] = 0;
                cal_time[i]=0;
                x_p[i] = tra_d[i][0];
                y_p[i] = tra_d[i][3]-40;
            }
            while (1)
            {
                int count=0,re=0;

  newmouse(&MouseX,&MouseY,&press);

  re=pressed_anime(xy_m[0],xy_m[1],xy_m[2],xy_
m[3]);
                if(re!=0)
                {
                    for(i=0;i<num;i++)
                    {
                        if(type[i]==0)
                        {
                            earth_cover01(x_p[i],
y_p[i]);
                            earth_cover01(x_p[i],
y_p[i] + 5);
                        }
                        if(type[i]==1)
                        {

  earth_cover02(x_p[i],y_p[i]);

  earth_cover02(x_p[i]+7,y_p[i]);
                        }
                    }
                    if(mode==0)
                    {
                        return;
                    }
                    break;
                }
                for(i=0;i<num;i++)
                {
                    if (type[i] == 4)
                    {
                        count++;
                    }
                }
                for (i = 0; i < num; i++)
                {
                    if(cal_time[i]>=0)
                    {
                        cal_time[i]++;
                    }

    if(cal_time[i]>0&&cal_time[i]>=pick_bar)
                    {
                        cal_time[i]=-1;
                        if(type[i]==0)
                        {

    draw_copak(x_p[i],y_p[i]+tracktor_l+co_pak_w);
                            des_x[i]=x_p[i];

    des_y[i]=y_p[i]+tracktor_l+co_pak_w;
                        }
                        else
                        {

    draw_copak(x_p[i],y_p[i]-2*co_pak_w);
                            des_x[i]=x_p[i];

    des_y[i]=y_p[i]-2*co_pak_w;
                        }
                    }
                    if ((type[i] != 0 && y_p[i] - 40 >=
y_start + y_max) || x_p[i] >= tra_d[i][2])
                    {
                        if (type[i] != 4)
                        {
                            init_tracktor01_f(x_p[i]
- 25, y_p[i]);
                            type[i] = 4;
                        }
                        else
                        {
                            continue;
                        }
                    }
                    else if (type[i] == 0)
                    {
                        earth_fill01(x_p[i], y_p[i]);
                        init_tracktor01_f(x_p[i],
y_p[i]);
                        y_p[i]--;
                        if (y_p[i] <= tra_d[i][1])
                        {
                            earth_fill03(x_p[i], y_p[i]
- 7);
                            type[i] = 1;
                            x_p[i] += 25;
                        }
                        delay(delaytime / (num -
count));
                        continue;
                    }
                    else if (type[i] == 1)
                    {
                        earth_fill02(x_p[i], y_p[i]);
                        init_tracktor01_b(x_p[i],
y_p[i]);
                        y_p[i]++;
                        if (y_p[i] + 40 >= tra_d[i][3])
                        {
                            if (x_p[i] + 25 >
tra_d[i][2])
                            {
                                type[i] = 3;
                            }
                            else
                            {
                                earth_fill03(x_p[i],
y_p[i] - 1);
                                earth_fill03(x_p[i],
y_p[i] + 2);
                                type[i] = 2;
                                x_p[i] += 25;
                            }
                        }
                        delay(delaytime / (num -
count));
                        continue;
                    }
                    else
                    {
                        type[i]=2;
                        earth_fill01(x_p[i], y_p[i]);
                        init_tracktor01_f(x_p[i],
y_p[i]);
                        y_p[i]--;
                        if (y_p[i] <= tra_d[i][1])
                        {
                            earth_fill03(x_p[i], y_p[i]
- 7);
                             type[i] = 1;
                             x_p[i] += 25;
                        }
                        delay(delaytime / (num -
count));
                        continue;
                    }
                    // else if (type[i] = 3)
                    // {
                    //    earth_fill02(x_p[i], y_p[i]);
                    //    init_tracktor01_b(x_p[i],
y_p[i]);
                    //    y_p[i]++;
                    // }
                }
                for(i=0;i<num;i++)
                {
                    draw_copak(des_x[i],des_y[i]);
                }
                if (count >= num)
                {
                    break;
                }
            }
            for(i=0;i<num;i++)
            {
                earth_fill03(x_p[i]-25,y_p[i]);
                earth_fill03(x_p[i]-25,y_p[i]-6);
                if(cal_time[i]>=0)
                {
                    des_x[i]=0;
                }
                // else
                // {
                //    draw_copak(des_x[i],des_y[i]);
                // }
            }
        }

        void start_ainime03_02(int (*tra_d)[4], int
num,long int *xy_m,int *des_x,int *des_y)
        {
            int
type[tracktor_num_max],x_p[tracktor_num_max]
,\
            y_p[tracktor_num_max],i,cal_time[tracktor
_num_max];
            for(i=0;i<num;i++)
            {
                type[i] = 0;
                cal_time[i]=0;
                x_p[i] = tra_d[i][0];
                y_p[i] = tra_d[i][3]-40;
            }
            while (1)
            {
                int count=0,re=0;

  newmouse(&MouseX,&MouseY,&press);

  re=pressed_anime(xy_m[0],xy_m[1],xy_m[2],xy_
m[3]);
                if(re!=0)
                {
                    for(i=0;i<num;i++)
                    {
                        if(type[i]==0)
                        {
                            earth_cover01(x_p[i],
y_p[i]);
                            earth_cover01(x_p[i],
y_p[i] + 5);
                        }
                        if(type[i]==1)
                        {

  earth_cover02(x_p[i],y_p[i]);

  earth_cover02(x_p[i]+7,y_p[i]);
                        }
                    }
                    if(mode==0)
                    {
                        return;
                    }
                    break;
                }
                for(i=0;i<num;i++)
                {
                    if (type[i] == 4)
```

```
			{
				count++;
			}
		}
		for (i = 0; i < num; i++)
		{
			if(cal_time[i]>=0)
			{
				cal_time[i]++;
			}

  if(cal_time[i]>0&&cal_time[i]>=pick_bar)
			{
				cal_time[i]=-1;
				if(type[i]==0)
				{

  draw_copak(x_p[i],y_p[i]+tracktor_l+co_pak_w);
					des_x[i]=x_p[i];

  des_y[i]=y_p[i]+tracktor_l+co_pak_w;
				}
				else
				{

  draw_copak(x_p[i],y_p[i]-2*co_pak_w);
					des_x[i]=x_p[i];

  des_y[i]=y_p[i]-2*co_pak_w;
				}
			}
			if ((type[i] != 0 && y_p[i] - 40 >=
y_start + y_max) || x_p[i] >= tra_d[i][2])
			{
				if (type[i] != 4)
				{
					init_tracktor02_f(x_p[i]
- 25, y_p[i]);
					type[i] = 4;
				}
				else
				{
					continue;
				}
			}
			else if (type[i] == 0)
			{
				earth_fill01(x_p[i], y_p[i]);
				init_tracktor02_f(x_p[i],
y_p[i]);

				y_p[i]--;
				if (y_p[i] <= tra_d[i][1])
				{
				    earth_fill03(x_p[i], y_p[i]
- 7);

					type[i] = 1;
					x_p[i] += 25;
				}
				delay(delaytime / (num -
count));

				continue;
			}
			else if (type[i] == 1)
			{
				earth_fill02(x_p[i], y_p[i]);
				init_tracktor02_b(x_p[i],
y_p[i]);

				y_p[i]++;
				if (y_p[i] + 40 >= tra_d[i][3])
				{
					if (x_p[i] + 25 >
tra_d[i][2])

					{
						type[i] = 3;
					}
					else
					{
						earth_fill03(x_p[i],
y_p[i] - 1);

						earth_fill03(x_p[i],
y_p[i] + 2);

						type[i] = 2;
						x_p[i] += 25;
					}
				}
				delay(delaytime / (num -
count));

				continue;
			}
			else
			{
				type[i] = 2;
				earth_fill01(x_p[i], y_p[i]);
				init_tracktor02_f(x_p[i],
y_p[i]);
                    y_p[i]--;
                    if (y_p[i] <= tra_d[i][1])
                    {
                        earth_fill03(x_p[i], y_p[i]
- 7);
                        type[i] = 1;
                        x_p[i] += 25;
                    }
                    delay(delaytime  /  (num  -
count));
                    continue;
                }
                // else if (type[i] = 3)
                // {
                //     earth_fill02(x_p[i], y_p[i]);
                //     init_tracktor02_b(x_p[i],
y_p[i]);
                //     y_p[i]++;
                // }
            }
            for(i=0;i<num;i++)
            {
                draw_copak(des_x[i],des_y[i]);
            }
            if (count >= num)
            {
                break;
            }
        }
        for(i=0;i<num;i++)
        {
            earth_fill03(x_p[i]-25,y_p[i]);
            earth_fill03(x_p[i]-25,y_p[i]-6);
            if(cal_time[i]>=0)
            {
                des_x[i]=0;
            }
            // else
            // {
            //     draw_copak(des_x[i],des_y[i]);
            // }
        }
        // if(cal_time[0]>=0)
        // {
        //     des_x[0]=0;
        // }
```

```
    }
    void circle_field(long int r)
    {
        long int x0 = (2 * x_start + x_max) / 2, y0 =
(2 * y_start + y_max) / 2;
        int i;
        setfillstyle(1,WHITE);
        fillellipse(x0,y0,r,r);
        setfillstyle(1,BROWN);
        bar(x_start,y_start,x_start+x_max,y_start+y
_max);
        if(r>=y_max/2-3)
        {
            r = y_max / 2 - 3;
        }
        setfillstyle(1, WHITE);
        fillellipse(x0, y0, r, r);
        setcolor(BROWN);
        for (i = 0; i < 4 * r * r / 20; i++)
        {
            int x_r = rand() % (2 * r), y_r = rand() %
(2 * r);
            line(x0 - r + x_r, y0 - r + y_r, x0 - r + x_r,
y0 - r + y_r);
        }
        for (i = x0 - r; i <= x0 + r; i += 25)
        {
            int j = 0;
            for (j = y0 - r; j <= y0 + r; j++)
            {
                int temp = rand() % 2;
                line(i + temp, j, i + temp, j);
            }
        }
    }

    void  cal_tracktor_circle(int  type,double
space,int time)
    {
        long                                int
x0=(2*x_start+x_max)/2,y0=(2*y_start+y_max)/2,
r;
        int
tra_d[tracktor_num_max][4],i,d,num,xy[2],des_x[
tracktor_num_max],des_y[tracktor_num_max];
        r=sqrt(space/3.1415926)*10;
        num=time/tra_time;
```

```
	if(num==0)
		num=1;
	if(r>=y_max/2-3)
	{
		r = y_max / 2 - 3;
	}
	for(i=0;i<num;i++)
	{
		des_x[i]=600;
		des_y[i]=0;
	}
	d = 2 * r / num;
	for (i = 0; i < num; i++)
	{
		tra_d[i][0] = x0 - r + i * d;
		tra_d[i][2] = tra_d[i][0] + d;
		if (tra_d[i][0] <= x0)
		{
			tra_d[i][1] = y0 - sqrt(r * r - (r - d *
(i + 1)) * (r - d * (i + 1)));
		}
		else
		{
			tra_d[i][1] = y0 - sqrt(r * r - (r - d *
i) * (r - d * i));
		}
		tra_d[i][3] = 2 * y0 - tra_d[i][1];
	}
	tra_d[num / 2 - 1][1] = y0 - r;
	tra_d[num / 2 - 1][3] = y0 + r;
	tra_d[num / 2][1] = y0 - r;
	tra_d[num / 2][3] = y0 + r;
	// if(num%2==0)
	// {
	//	for(i=0;i<num/2;i++)
	//	{
	//		tra_d[i][0]=x0-r+i*d;
	//		tra_d[i][2]=tra_d[i][0]+d;
	//
  tra_d[i][1]=y0-sqrt(r*r-(r-d*(i+1))*(r-d*(i+1)));
	//		tra_d[i][3]=2*y0-tra_d[i][1];
	//	}
	//	tra_d[i-1][1]=y0-r;
	//	tra_d[i-1][3]=y0+r;
	//	for(i=num/2;i<num;i++)
	//	{
	//
  tra_d[i][0]=2*x0-tra_d[num-i-1][2];
	//
  tra_d[i][2]=2*x0-tra_d[num-i-1][0];
	//		tra_d[i][1]=tra_d[num-i-1][1];
	//		tra_d[i][3]=tra_d[num-i-1][3];
	//	}
	// }
	// else
	// {
	//	for(i=0;i<num/2;i++)
	//	{
	//		tra_d[i][0]=x0-r+i*d;
	//		tra_d[i][2]=tra_d[i][0]+d;
	//
  tra_d[i][1]=y0-sqrt(r*r-(r-d*(i+1))*(r-d*(i+1)));
	//		tra_d[i][3]=2*y0-tra_d[i][1];
	//	}
	//	tra_d[i][1]=y0-r;
	//	tra_d[i][3]=y0+r;
	//	for(i=num/2+1;i<num;i++)
	//	{
	//
  tra_d[i][0]=2*x0-tra_d[num-i-1][2];
	//
  tra_d[i][2]=2*x0-tra_d[num-i-1][0];
	//		tra_d[i][1]=tra_d[num-i-1][1];
	//		tra_d[i][3]=tra_d[num-i-1][3];
	//	}
	// }
	select_setoff02(xy,x0-r,y0-r,x0+r,y0+r);
	draw_simu01(time);
	circle_field(r);
	clrmous(MouseX,MouseY);
	draw_setoff(xy);
	if(type==0)
	{

  tracktor_set_off(xy[0],xy[1],x0-r,y0+r,2*r/num,n
um);
		clrmous(MouseX,MouseY);
		draw_setoff(xy);

  start_ainime04_01(tra_d,num,des_x,des_y);
		for(i=0;i<num;i++)
		{

  earth_fill03(x0-r+2*r/num+i*2*r/num,y0-r);
  earth_fill03(x0-r+2*r/num+i*2*r/num,y0-r-6);
                }

  tracktor_return(xy[0],xy[1],x0-r+2*r/num,y0-r,2*
r/num,num);

  picker_anime(xy[0],xy[1],des_x,des_y,2*r/num,n
um);
        }
        else
        {

  tracktor_set_off0(xy[0],xy[1],x0-r,y0+r,2*r/num,
num);
                clrmous(MouseX,MouseY);
                draw_setoff(xy);

  start_ainime04_02(tra_d,num,des_x,des_y);
                for(i=0;i<num;i++)
                {

  earth_fill03(x0-r+2*r/num+i*2*r/num,y0-r);

  earth_fill03(x0-r+2*r/num+i*2*r/num,y0-r-6);
                }

  tracktor_return0(xy[0],xy[1],x0-r+2*r/num,y0-r,2
*r/num,num);

  picker_anime(xy[0],xy[1],des_x,des_y,2*r/num,n
um);
        }
        return;
      }

      void    start_ainime04_01(int    (*tra_d)[4],int
num,int *des_x,int *des_y)
      {
        int
type[tracktor_num_max],i,x_p[tracktor_num_max
]\
        ,y_p[tracktor_num_max],cal_time[tracktor_
num_max];
        for(i=0;i<num;i++)
        {
                type[i]=0;
                cal_time[i]=0;
                x_p[i]=tra_d[i][0];
                y_p[i]=y_start+y_max-40;
        }
        while(1)
        {
                int count=0,re=0;

newmouse(&MouseX,&MouseY,&press);

  re=pressed_anime(tra_d[0][0],tra_d[num/2][1],t
ra_d[num-1][2],tra_d[num/2][3]);
                if(re!=0)
                {
                        for(i=0;i<num;i++)
                        {
                                if(type[i]==0)
                                {
                                        earth_cover01(x_p[i],
y_p[i]);
                                        earth_cover01(x_p[i],
y_p[i] + 5);
                                }
                                if(type[i]==1)
                                {

  earth_cover02(x_p[i],y_p[i]);

  earth_cover02(x_p[i]+7,y_p[i]);
                                }
                        }
                        if(mode==0)
                        {
                                return;
                        }
                        break;
                }
                for(i=0;i<num;i++)
                {
                        if(type[i]==4)
                        {
                                count++;
                        }
                }
                for(i=0;i<num;i++)
                {
                        if(cal_time[i]>=0)
                        {
                                cal_time[i]++;
                        }

  if(cal_time[i]>0&&cal_time[i]>=pick_bar)
                        {
                                cal_time[i]=-1;
                                if(type[i]==0)
                                {

  draw_copak(x_p[i],y_p[i]+tracktor_l+co_pak_w);
                                        des_x[i]=x_p[i];

  des_y[i]=y_p[i]+tracktor_l+co_pak_w;
                                }
                                else
                                {

  draw_copak(x_p[i],y_p[i]-2*co_pak_w);
                                        des_x[i]=x_p[i];

  des_y[i]=y_p[i]-2*co_pak_w;
                                }
                        }

  if((type[i]!=0&&y_p[i]-40>=y_start+y_max)||x_p
[i]>=tra_d[i][2])
                        {
                                if(type[i]!=4)
                                {

  init_tracktor01_f(x_p[i]-25,y_p[i]);
                                        type[i]=4;
                                }
                                else
                                {
                                        continue;
                                }
                        }
                        else if(type[i]==0)
                        {
                                earth_fill01(x_p[i],y_p[i]);

  init_tracktor01_f(x_p[i],y_p[i]);
                                y_p[i]--;
                                if(y_p[i]<=tra_d[i][1])
                                {

earth_fill03(x_p[i],y_p[i]-7);
                                        type[i]=1;
                                        x_p[i]+=25;
                                }

delay(delaytime/(num-count));
                                continue;
                        }
                        else if(type[i]==1)
                        {
                                earth_fill02(x_p[i],y_p[i]);

init_tracktor01_b(x_p[i],y_p[i]);
                                y_p[i]++;
                                if(y_p[i]+40>=tra_d[i][3])
                                {
                                        if(x_p[i]+25>tra_d[i][2])
                                        {
                                                type[i]=3;
                                        }
                                        else
                                        {

earth_fill03(x_p[i],y_p[i]-1);

earth_fill03(x_p[i],y_p[i]+2);
                                                type[i]=2;
                                                x_p[i]+=25;
                                        }
                                }

delay(delaytime/(num-count));
                                continue;
                        }
                        else
                        {
                                type[i]=2;
                                earth_fill01(x_p[i],y_p[i]);

init_tracktor01_f(x_p[i],y_p[i]);
                                y_p[i]--;
                                if(y_p[i]<=tra_d[i][1])
                                {

earth_fill03(x_p[i],y_p[i]-7);
                                        type[i]=1;
                                x_p[i]+=25;
                            }
  delay(delaytime/(num-count));
                            continue;
                        }
                        // else if(type[i]=3)
                        // {
                        //     earth_fill02(x_p[i],y_p[i]);
                        //
  init_tracktor01_b(x_p[i],y_p[i]);
                        //     y_p[i]++;
                        // }
                    }
                    for(i=0;i<num;i++)
                    {
                        draw_copak(des_x[i],des_y[i]);
                    }
                    if(count>=num)
                    {
                        break;
                    }
                }
                for(i=0;i<num;i++)
                {
                    earth_fill03(x_p[i]-25,y_p[i]);
                    earth_fill03(x_p[i]-25,y_p[i]-6);
                    if(cal_time[i]>=0)
                    {
                        des_x[i]=0;
                    }
                    // else
                    // {
                    //     draw_copak(des_x[i],des_y[i]);
                    // }
                }
                // if(cal_time[0]>=0)
                // {
                //     des_x[0]=0;
                // }
            }

        void    start_ainime04_02(int    (*tra_d)[4],int
num,int *des_x,int *des_y)
        {
            int
type[tracktor_num_max],i,x_p[tracktor_num_max
],\
            y_p[tracktor_num_max],cal_time[tracktor_
num_max];
            for(i=0;i<num;i++)
            {
                cal_time[i]=0;
                type[i]=0;
                x_p[i]=tra_d[i][0];
                y_p[i]=y_start+y_max-40;
            }
            while(1)
            {
                int count=0,re=0;

newmouse(&MouseX,&MouseY,&press);

  re=pressed_anime(tra_d[0][0],tra_d[num/2-1][1
],tra_d[num-1][2],tra_d[num/2-1][3]);
                if(re!=0)
                {
                    for(i=0;i<num;i++)
                    {
                        if(type[i]==0)
                        {
                            earth_cover01(x_p[i],
y_p[i]);
                            earth_cover01(x_p[i],
y_p[i] + 5);
                        }
                        if(type[i]==1)
                        {

  earth_cover02(x_p[i],y_p[i]);

  earth_cover02(x_p[i]+7,y_p[i]);
                        }
                    }
                    if(mode==0)
                    {
                        return;
                    }
                    break;
                }
                for(i=0;i<num;i++)
                {
                    if(type[i]==4)
                    {
                        count++;
                    }
                }
                for(i=0;i<num;i++)
                {
                    if(cal_time[i]>=0)
                    {
                        cal_time[i]++;
                    }

    if(cal_time[i]>0&&cal_time[i]>=pick_bar)
                    {
                        cal_time[i]=-1;
                        if(type[i]==0)
                        {

    draw_copak(x_p[i],y_p[i]+tracktor_l+co_pak_w);
                            des_x[i]=x_p[i];

    des_y[i]=y_p[i]+tracktor_l+co_pak_w;
                        }
                        else
                        {

    draw_copak(x_p[i],y_p[i]-2*co_pak_w);
                            des_x[i]=x_p[i];

    des_y[i]=y_p[i]-2*co_pak_w;
                        }
                    }

    if((type[i]!=0&&y_p[i]-40>=y_start+y_max)||x_p
[i]>=tra_d[i][2])
                    {
                        if(type[i]!=4)
                        {

    init_tracktor02_f(x_p[i]-25,y_p[i]);
                            type[i]=4;
                        }
                        else
                        {
                            continue;
                        }
                    }
                    else if(type[i]==0)
                    {
                        earth_fill01(x_p[i],y_p[i]);

init_tracktor02_f(x_p[i],y_p[i]);
                        y_p[i]--;
                        if(y_p[i]<=tra_d[i][1])
                        {

earth_fill03(x_p[i],y_p[i]-7);
                            type[i]=1;
                            x_p[i]+=25;
                        }

delay(delaytime/(num-count));
                        continue;
                    }
                    else if(type[i]==1)
                    {
                        earth_fill02(x_p[i],y_p[i]);

init_tracktor02_b(x_p[i],y_p[i]);
                        y_p[i]++;
                        if(y_p[i]+40>=tra_d[i][3])
                        {
                            if(x_p[i]+25>tra_d[i][2])
                            {
                                type[i]=3;
                            }
                            else
                            {

earth_fill03(x_p[i],y_p[i]-1);

earth_fill03(x_p[i],y_p[i]+2);
                                type[i]=2;
                                x_p[i]+=25;
                            }
                        }

delay(delaytime/(num-count));
                        continue;
                    }
                    else
                    {
                        type[i]=2;
                        earth_fill01(x_p[i],y_p[i]);

init_tracktor02_f(x_p[i],y_p[i]);
```

```
                        y_p[i]--;
                        if(y_p[i]<=tra_d[i][1])
                        {
earth_fill03(x_p[i],y_p[i]-7);
                              type[i]=1;
                              x_p[i]+=25;
                        }
delay(delaytime/(num-count));
                        continue;
                  }
                  // else if(type[i]=3)
                  // {
                  //    earth_fill02(x_p[i],y_p[i]);
                  //
init_tracktor02_b(x_p[i],y_p[i]);
                  //    y_p[i]++;
                  // }
            }
            for(i=0;i<num;i++)
            {
                  draw_copak(des_x[i],des_y[i]);
            }
            if(count>=num)
            {
                  break;
            }
      }
      for(i=0;i<num;i++)
      {
            earth_fill03(x_p[i]-25,y_p[i]);
            earth_fill03(x_p[i]-25,y_p[i]-6);
            if(cal_time[i]>=0)
            {
                  des_x[i]=0;
            }
            // else
            // {
            //    draw_copak(des_x[i],des_y[i]);
            // }
      }
      // if(cal_time[0]>=0)
      // {
      //    des_x[0]=0;
      // }
    }
```

```
      long int hellen(int x1, int y1, int x2, int y2, int
x3, int y3)
      {
        long int a = sqrt((x1 - x2) * (x1 - x2) + (y1 -
y2) * (y1 - y2)), b = sqrt((x1 - x3) * (x1 - x3) + (y1 -
y3) * (y1 - y3)),
                    c = sqrt((x3 - x2) * (x3 - x2) + (y3
- y2) * (y3 - y2)), p = (a + b + c) / 2;

        return sqrt(p * (p - a) * (p - b) * (p - c));
      }

      long int cal_poly_field(int *x, int *y, int *flag)
      {
        int x0 = (x[0] + x[(*flag) / 2]) / 2, y0 = (y[0] +
y[(*flag) / 2]) / 2, i;
        long int s_field = 0;
        for (i = 0; i < (*flag) - 2; i++)
        {
              s_field += hellen(x0, y0, x[i], y[i], x[i +
1], y[i + 1]);
        }
        s_field += hellen(x0, x0, x[i], y[i], x[0], x[0]);
        return s_field;
      }

      double cal_circle_field(int r)
      {
        double pi = 3.1415926;
        return pi * r * r;
      }


      void select_setoff01(int *xy,int x_end,int
y_end)
      {
        //int i=0;
        cleardevice();
        setbkcolor(WHITE);
        puthz(150,30,"请在框外选择农机出发点
",32,32,BLUE);
        init_based_field();
        setlinestyle(0,0,3);
        setcolor(GREEN);
        rectangle(x_start,y_start,x_end,y_end);
        // for(i=0;i<100;i++)
```

```
	// {
	//
  newmouse(&MouseX,&MouseY,&press);
	//	delay(10);
	// }
	while(1)
	{

  newmouse(&MouseX,&MouseY,&press);

  if(mouse_press(0,y_start,x_start,480)==1||mous
e_press(x_start,y_end,x_end,480)==1||mouse_pr
ess(x_end,y_start,615,480)==1)
		{
			xy[0]=MouseX;
			xy[1]=MouseY;
			break;
		}
		delay(20);
	}
	if(xy[0]<=x_start&&xy[0]+tra_start_l>=x_sta
rt)
	{
		xy[0]=x_start-tra_start_l;
	}
	if(xy[0]>=x_start&&xy[0]<=x_end&&xy[1]<=
y_end)
	{
		xy[1]=y_end;
	}

	clrmous(MouseX,MouseY);
	draw_setoff(xy);
	choose_speed();
	return;
       }

       void select_setoff02(int *xy,int x_s,int y_s,int
x_e,int y_e)
       {
	cleardevice();
	setbkcolor(WHITE);
	puthz(100,30,"请在框外选择农机出发点
",32,32,BLUE);
	//puthz(400,40,"速度",32,32,GREEN);
	init_based_field();
	setlinestyle(0,0,3);
	setcolor(GREEN);
	rectangle(x_s,y_s,x_e,y_e);
	// for(i=0;i<100;i++)
	// {
	//
  newmouse(&MouseX,&MouseY,&press);
	//	delay(10);
	// }
	while(1)
	{

  newmouse(&MouseX,&MouseY,&press);

  if(mouse_press(0,y_start,x_s,480)==1||mouse_
press(x_s,y_e,x_e,480)==1\

  ||mouse_press(x_e,y_start,615,480)==1||mous
e_press(x_s,y_start,x_e,y_s)==1)
		{
			xy[0]=MouseX;
			xy[1]=MouseY;
			break;
		}
		delay(20);
	}
	// while(1)
	// {
	//	if(bioskey(0)==p_Enter)
	//	{
	//		break;
	//	}
	//	if(flag==0)
	//	{
	//		itoa(100*times,str,5);
	//		outtextxy(450,40,str);
	//		flag=1;
	//	}
	//	if(times>0.5&&
bioskey(0)==p_Up_arrow)
	//	{
	//		times-=0.05;
	//		flag=0;
	//	}
	//
  if(times<2&&bioskey(0)==p_Donw_arrow)
	//	{
	//		times+=0.05;
```

```
    //          flag=0;
    //    }
    //    delay(delaytime);
    // }
    if(xy[0]<=x_s&&xy[0]+tra_start_l>=x_s)
    {
        xy[0]=x_s-tra_start_l;
    }
    if(xy[0]+tra_start_l>=615)
    {
        xy[0]=615-tra_start_l;
    }
    if(xy[0]>=x_s&&xy[0]<=x_e&&xy[1]<=y_e)
    {
        xy[1]=y_e;
    }
    if(xy[1]<y_s+tra_start_d)
    {
        xy[1]=y_s+tra_start_d;
    }
    clrmous(MouseX,MouseY);
    draw_setoff(xy);
    choose_speed();
    return;
   }

   void choose_speed()
   {
    float x_p=0;
    setcolor(RED);
    setfillstyle(1,WHITE);
    bar(0,0,480,70);
    puthz(100,30,"请在横轴上选择农机速度
值",32,32,BLUE);
    setfillstyle(1,RED);
    bar(320-100,85,320,95);
    setfillstyle(1,GREEN);
    bar(320,85,320+100,95);
    while(1)
    {

  newmouse(&MouseX,&MouseY,&press);

  if(mouse_press(320-100,85,320+100,95)==1)
        {
            x_p=420-MouseX;
            break;
        }
        if
(mouse_press(320-100,85,320+100,95) == 2 )
        {
            MouseS = 1;
        }
        delay(delaytime);
    }
    if(x_p<=20)
    {
        x_p=20;
    }
    x_p/=200;
    delaytime*=x_p;
    clrmous(MouseX,MouseY);
   }

   void draw_setoff(int *xy)
   {
    setfillstyle(1,LIGHTGRAY);
    bar(xy[0],xy[1],xy[0]+tra_start_l,xy[1]+tra_s
tart_d);
   }



   void    tracktor_set_off(int    start_x,int
start_y,int des_x,int des_y,int distance,int num)
   {
    if (start_x < des_x && start_y < des_y)
    {
        tracktor_set_off01(start_x,    start_y,
des_x, des_y, distance, num);
    }
    else if (start_x < des_x && start_y > des_y)
    {
        tracktor_set_off02(start_x,    start_y,
des_x, des_y, distance, num);
    }
    else if (start_x > des_x && start_y < des_y)
    {
        tracktor_set_off03(start_x,    start_y,
des_x, des_y, distance, num);
    }
    else
    {
        tracktor_set_off04(start_x,    start_y,
des_x, des_y, distance, num);
         }
       }

       // start_x<des_x&&start_y<des_y
       void tracktor_set_off01(int start_x, int start_y,
int des_x, int des_y, int distance, int num)
       {
         int                         x_p[tracktor_num_max],
y_p[tracktor_num_max],
type[tracktor_num_max],
time[tracktor_num_max], i;
         for (i = 0; i < num; i++)
         {
                type[i] = 0;
                time[i] = i * distance;
                x_p[i] = (2 * start_x + tra_start_l) / 2 -
tracktor_w / 2;
                y_p[i] = start_y + tra_start_d + 1;
         }
         //clrmous(MouseX,MouseY);
         while (1)
         {
                int count = 0,re=0;

  newmouse(&MouseX,&MouseY,&press);

  re=pressed_anime(x_start,y_start,x_start,y_start
);
                if(re!=0)
                {
                       for(i=0;i<num;i++)
                       {
                              if(type[i]==0)
                              {
                                     earth_cover01(x_p[i],
y_p[i]);
                                     earth_cover01(x_p[i],
y_p[i] + 5);
                              }
                              if(type[i]==1)
                              {

  earth_cover02(x_p[i],y_p[i]);

  earth_cover02(x_p[i]+7,y_p[i]);
                              }
                       }
                       if(mode==0)
                       {
                              return;
                       }
                       break;
                }
                for (i = 0; i < num; i++)
                {
                       if (time[i] > 0)
                       {
                              time[i]--;
                              continue;
                       }
                       if (type[i] == 2)
                       {
                              count++;
                              continue;
                       }
                       if (type[i] == 0)
                       {
                              earth_cover01(x_p[i], y_p[i]);
                              y_p[i]++;
                              init_tracktor01_b(x_p[i],
y_p[i]);
                              if (y_p[i] >= des_y)
                              {
                                     earth_cover01(x_p[i],
y_p[i]);
                                     earth_cover01(x_p[i],
y_p[i] + 5);
                                     type[i] = 1;
                                     x_p[i] += tracktor_l;
                                     init_tracktor01_r(x_p[i],
y_p[i]);
                                     continue;
                              }
                       }
                       if (type[i] == 1)
                       {
                              earth_cover02(x_p[i], y_p[i]);
                              x_p[i]++;
                              init_tracktor01_r(x_p[i],
y_p[i]);
                              if (x_p[i] >= des_x + (num - i -
1) * distance-tracktor_l)
                              {
```

```
  earth_cover02(x_p[i],y_p[i]);

  earth_cover02(x_p[i]+7,y_p[i]);
                              type[i]=2;

  //init_tracktor01_f(x_p[i],y_p[i]-35);
                              continue;
                          }
                      }
                  }
                  if (count >= num)
                  {
                      break;
                  }
                  delay(delaytime);
              }
          }

          // start_x<des_x&&start_y>des_y
          void tracktor_set_off02(int start_x, int start_y,
int des_x, int des_y, int distance, int num)
          {
              int                     x_p[tracktor_num_max],
y_p[tracktor_num_max],
type[tracktor_num_max],
time[tracktor_num_max], i;
              for (i = 0; i < num; i++)
              {
                  type[i] = 0;
                  time[i] = i * distance;
                  x_p[i] = (2 * start_x + tra_start_l) / 2 -
tracktor_w / 2;
                  y_p[i]  =  start_y  -  tra_start_d  -  1  -
tracktor_l;
              }
              //clrmous(MouseX,MouseY);
              while (1)
              {
                  int count = 0,re=0;

  newmouse(&MouseX,&MouseY,&press);

  re=pressed_anime(x_start,y_start,x_start,y_start
);
                  if(re!=0)
                  {
                      for(i=0;i<num;i++)
                      {
                          if(type[i]==0)
                          {
                              earth_cover01(x_p[i],
y_p[i]);
                              earth_cover01(x_p[i],
y_p[i] -7);
                          }
                          if(type[i]==1)
                          {

  earth_cover02(x_p[i],y_p[i]);

  earth_cover02(x_p[i]+7,y_p[i]);
                          }
                      }
                      if(mode==0)
                      {
                          return;
                      }
                      break;
                  }
                  for (i = 0; i < num; i++)
                  {
                      if (time[i] > 0)
                      {
                          time[i]--;
                          continue;
                      }
                      if (type[i] == 2)
                      {
                          count++;
                          continue;
                      }
                      if (type[i] == 0)
                      {
                          earth_cover01(x_p[i], y_p[i]);
                          y_p[i]--;
                          init_tracktor01_f(x_p[i],
y_p[i]);
                          if (y_p[i] <= des_y)
                          {
                              earth_cover01(x_p[i],
y_p[i]);
                              earth_cover01(x_p[i],
y_p[i] - 7);
```

```
                                type[i] = 1;
                                x_p[i] += tracktor_l;
                                init_tracktor01_r(x_p[i],
y_p[i]);
                                continue;
                        }
                }
                if (type[i] == 1)
                {
                        earth_cover02(x_p[i], y_p[i]);
                        x_p[i]++;
                        init_tracktor01_r(x_p[i],
y_p[i]);
                        if (x_p[i] >= des_x + (num - i -
1) * distance-tracktor_l)
                        {

  earth_cover02(x_p[i],y_p[i]);

  earth_cover02(x_p[i]+7,y_p[i]);
                                type[i]=2;

  //init_tracktor01_f(x_p[i],y_p[i]-35);
                                continue;
                        }
                }
        }
        if (count >= num)
        {
                break;
        }
      delay(delaytime);
      }
    }

    // start_x>des_x&&start_y<des_y
    void tracktor_set_off03(int start_x, int start_y,
int des_x, int des_y, int distance, int num)
    {
      int                     x_p[tracktor_num_max],
y_p[tracktor_num_max],
type[tracktor_num_max],
time[tracktor_num_max], i;
      for (i = 0; i < num; i++)
      {
            type[i] = 0;
            time[i] = i * distance;
            x_p[i] = (2 * start_x + tra_start_l) / 2 -
tracktor_w / 2;
            y_p[i] = start_y + tra_start_d + 1;
      }
      //clrmous(MouseX,MouseY);
      while (1)
      {
            int count = 0,re=0;

  newmouse(&MouseX,&MouseY,&press);

  re=pressed_anime(x_start,y_start,x_start,y_start
);
            if(re!=0)
            {
                  for(i=0;i<num;i++)
                  {
                        if(type[i]==0)
                        {
                              earth_cover01(x_p[i],
y_p[i]);
                              earth_cover01(x_p[i],
y_p[i] + 5);
                        }
                        if(type[i]==1)
                        {

  earth_cover02(x_p[i]+2,y_p[i]);

  earth_cover02(x_p[i]-5,y_p[i]);
                        }
                  }
                  if(mode==0)
                  {
                        return;
                  }
                  break;
            }
            for (i = 0; i < num; i++)
            {
                  if (time[i] > 0)
                  {
                        time[i]--;
                        continue;
                  }
                  if (type[i] == 2)
                  {
```

```
				count++;
				continue;
			}
			if (type[i] == 0)
			{
				earth_cover01(x_p[i], y_p[i]);
				y_p[i]++;
				init_tracktor01_b(x_p[i],
y_p[i]);
				if (y_p[i] >= des_y)
				{
					earth_cover01(x_p[i],
y_p[i]);
					earth_cover01(x_p[i],
y_p[i] + 5);
					type[i] = 1;
					x_p[i] += tracktor_l;
					init_tracktor01_l(x_p[i],
y_p[i]);
					continue;
				}
			}
			if (type[i] == 1)
			{
				earth_cover02(x_p[i]    +    5,
y_p[i]);
				x_p[i]--;
				init_tracktor01_l(x_p[i],
y_p[i]);
				if   (x_p[i]   <=   des_x   +   i   *
distance+tracktor_l)
				{

  earth_cover02(x_p[i]+2,y_p[i]);

  earth_cover02(x_p[i]-5,y_p[i]);
					type[i]=2;

  //init_tracktor01_f(x_p[i]-50,y_p[i]-35);
					continue;
				}
			}
		}
		if (count >= num)
		{
			break;
		}
			delay(delaytime);
		}
	}

	// start_x>=des_x&&start_y>=des_y
	void tracktor_set_off04(int start_x, int start_y,
int des_x, int des_y, int distance, int num)
	{
		int                          x_p[tracktor_num_max],
y_p[tracktor_num_max],
type[tracktor_num_max],
time[tracktor_num_max], i;
		for (i = 0; i < num; i++)
		{
			type[i] = 0;
			time[i] = i * distance;
			x_p[i] = (2 * start_x + tra_start_l) / 2 -
tracktor_w / 2;
			y_p[i]  =  start_y  -  tra_start_d  -  1  -
tracktor_l;
		}
		//clrmous(MouseX,MouseY);
		while (1)
		{
			int count = 0,re=0;

  newmouse(&MouseX,&MouseY,&press);

  re=pressed_anime(x_start,y_start,x_start,y_start
);
			if(re!=0)
			{
				for(i=0;i<num;i++)
				{
					if(type[i]==0)
					{
						earth_cover01(x_p[i],
y_p[i]);
						earth_cover01(x_p[i],
y_p[i] -7);
					}
					if(type[i]==1)
					{

  earth_cover02(x_p[i]+2,y_p[i]);

  earth_cover02(x_p[i]-5,y_p[i]);
```

```
                }
            }
            if(mode==0)
            {
                return;
            }
            break;
        }
        for (i = 0; i < num; i++)
        {
            if (time[i] > 0)
            {
                time[i]--;
                continue;
            }
            if (type[i] == 2)
            {
                count++;
                continue;
            }
            if (type[i] == 0)
            {
                earth_cover01(x_p[i], y_p[i]);
                y_p[i]--;
                init_tracktor01_f(x_p[i],
y_p[i]);
                if (y_p[i] <= des_y)
                {
                    earth_cover01(x_p[i],
y_p[i]);
                    earth_cover01(x_p[i],
y_p[i] - 7);
                    type[i] = 1;
                    x_p[i] += tracktor_l;
                    init_tracktor01_r(x_p[i],
y_p[i]);
                    continue;
                }
            }
            if (type[i] == 1)
            {
                earth_cover02(x_p[i]    +    7,
y_p[i]);
                x_p[i]--;
                init_tracktor01_l(x_p[i],
y_p[i]);
                if   (x_p[i]   <=   des_x   +   i   *
distance+tracktor_l)
                {

  earth_cover02(x_p[i]+2,y_p[i]);

  earth_cover02(x_p[i]-5,y_p[i]);
                    type[i]=2;

  //init_tracktor01_f(x_p[i]-50,y_p[i]-35);
                    continue;
                }
            }
        }
        if (count >= num)
        {
            break;
        }
        delay(delaytime);
    }
}

    void  tracktor_return(int  start_x,  int  start_y,
int des_x, int des_y, int distance, int num)
    {
      if(start_x<des_x&&start_y>des_y)
      {
          tracktor_return01(start_x,         start_y,
des_x, des_y, distance, num);
      }
      else if(start_x>des_x&&start_y>des_y)
      {

  tracktor_return02(start_x,start_y,des_x,des_y,di
stance,num);
      }
      else if(start_x<des_x&&start_y<des_y)
      {

  tracktor_return03(start_x,start_y,des_x,des_y,di
stance,num);
      }
      else
      {

  tracktor_return04(start_x,start_y,des_x,des_y,di
stance,num);
      }
```

```
	}

	void	tracktor_return01(int	start_x,int
start_y,int des_x,int des_y,int distance,int num)
	{
	  int
x_p[tracktor_num_max],y_p[tracktor_num_max],
type[tracktor_num_max],i;
	  if(distance<tracktor_l)
	  {
		distance=tracktor_l;
	  }
	  for(i=0;i<num;i++)
	  {
		type[i]=0;
		x_p[i]=des_x+i*distance;
		y_p[i]=des_y;
		earth_fill03(x_p[i],y_p[i]);
		x_p[i]-=tracktor_l;
		y_p[i]-=tracktor_w;
	  }
	  while(1)
	  {
		int count = 0,re=0;

  newmouse(&MouseX,&MouseY,&press);

  re=pressed_anime(x_start,y_start,x_start,y_start
);
		if(re!=0)
		{
			for(i=0;i<num;i++)
			{
				if(type[i]==0)
				{

  earth_cover02(x_p[i]+2,y_p[i]);

  earth_cover02(x_p[i]-6,y_p[i]);
				}
				if(type[i]==1)
				{

  earth_cover01(x_p[i],y_p[i]);

  earth_cover01(x_p[i],y_p[i]+5);
				}
			}
			if(mode==0)
			{
				return;
			}
			break;
		}
		for(i=0;i<num;i++)
		{
			if(type[i]==2)
			{
				count++;
				continue;
			}
			if(type[i]==0)
			{

  earth_cover02(x_p[i]+2,y_p[i]);
				x_p[i]--;

  init_tracktor01_l(x_p[i],y_p[i]);

  if(x_p[i]<=((2*start_x+tra_start_l)/2-tracktor_w/
2)+50)
				{

  earth_cover02(x_p[i]+2,y_p[i]);

  earth_cover02(x_p[i]-6,y_p[i]);

  x_p[i]=(2*start_x+tra_start_l)/2-tracktor_w/2;
					type[i]=1;
					continue;
				}
			}
			if(type[i]==1)
			{
				earth_cover01(x_p[i],y_p[i]);
				y_p[i]++;

  init_tracktor01_b(x_p[i],y_p[i]);
				if(y_p[i]>=start_y-48)
				{

  earth_cover01(x_p[i],y_p[i]);

  earth_cover01(x_p[i],y_p[i]+5);
```

```
                        type[i]=2;
                        continue;
                    }
                }
            }
            if(count>=num)
            {
                break;
            }
            delay(delaytime);
        }
    }

    void    tracktor_return02(int    start_x,int
start_y,int des_x,int des_y,int distance,int num)
    {
        int
x_p[tracktor_num_max],y_p[tracktor_num_max],
type[tracktor_num_max],i;
        if(distance<tracktor_l)
        {
            distance=tracktor_l;
        }
        for(i=0;i<num;i++)
        {
            type[num-1-i]=0;
            x_p[num-1-i]=des_x+i*distance;
            y_p[num-1-i]=des_y;
            earth_fill03(x_p[i],y_p[i]);
            x_p[num-1-i]+=tracktor_l;
            y_p[num-1-i]-=tracktor_w;
        }
        while(1)
        {
            int count = 0,re=0;

  newmouse(&MouseX,&MouseY,&press);

  re=pressed_anime(x_start,y_start,x_start,y_start
);
            if(re!=0)
            {
                for(i=0;i<num;i++)
                {
                    if(type[i]==0)
                    {

  earth_cover02(x_p[i]+6,y_p[i]);

  earth_cover02(x_p[i]-2,y_p[i]);
                    }
                    if(type[i]==1)
                    {

  earth_cover01(x_p[i],y_p[i]);

  earth_cover01(x_p[i],y_p[i]+5);
                    }
                }
                if(mode==0)
                {
                    return;
                }
                break;
            }
            for(i=0;i<num;i++)
            {
                if(type[i]==2)
                {
                    count++;
                    continue;
                }
                if(type[i]==0)
                {

  earth_cover02(x_p[i]+2,y_p[i]);
                    x_p[i]++;

  init_tracktor01_r(x_p[i],y_p[i]);

  if(x_p[i]>=((2*start_x+tra_start_l)/2-tracktor_w/
2))
                    {

  earth_cover02(x_p[i]+6,y_p[i]);

  earth_cover02(x_p[i]-2,y_p[i]);

  x_p[i]=(2*start_x+tra_start_l)/2-tracktor_w/2;
                        type[i]=1;
                        continue;
                    }
                }
                if(type[i]==1)
```

```
                    {
                            earth_cover01(x_p[i],y_p[i]);
                            y_p[i]++;

  init_tracktor01_b(x_p[i],y_p[i]);
                            if(y_p[i]>=start_y-48)
                            {

  earth_cover01(x_p[i],y_p[i]);

  earth_cover01(x_p[i],y_p[i]+5);
                                    type[i]=2;
                                    continue;
                            }
                    }
            }
            if(count>=num)
            {
                    break;
            }
            delay(delaytime);
      }
    }

    void      tracktor_return03(int      start_x,int
start_y,int des_x,int des_y,int distance,int num)
    {
      int
x_p[tracktor_num_max],y_p[tracktor_num_max],
type[tracktor_num_max],i;
      if(distance<tracktor_l)
      {
            distance=tracktor_l;
      }
      for(i=0;i<num;i++)
      {
            type[i]=0;
            x_p[i]=des_x+i*distance;
            y_p[i]=des_y;
            earth_fill03(x_p[i],y_p[i]);
            x_p[i]-=tracktor_l;
            y_p[i]-=tracktor_w;
      }
      while(1)
      {
            int count = 0,re=0;

  newmouse(&MouseX,&MouseY,&press);

  re=pressed_anime(x_start,y_start,x_start,y_start
);
            if(re!=0)
            {
                    for(i=0;i<num;i++)
                    {
                            if(type[i]==0)
                            {

  earth_cover02(x_p[i]+2,y_p[i]);

  earth_cover02(x_p[i]-6,y_p[i]);
                            }
                            if(type[i]==1)
                            {

  earth_cover01(x_p[i],y_p[i]-7);

  earth_cover01(x_p[i],y_p[i]+3);
                            }
                    }
                    if(mode==0)
                    {
                            return;
                    }
                    break;
            }
            for(i=0;i<num;i++)
            {
                    if(type[i]==2)
                    {
                            count++;
                            continue;
                    }
                    if(type[i]==0)
                    {

  earth_cover02(x_p[i]+2,y_p[i]);
                            x_p[i]--;

  init_tracktor01_l(x_p[i],y_p[i]);

  if(x_p[i]<=((2*start_x+tra_start_l)/2-tracktor_w/
2)+50)
                            {
```

```
  earth_cover02(x_p[i]+2,y_p[i]);

  earth_cover02(x_p[i]-6,y_p[i]);

  x_p[i]=(2*start_x+tra_start_l)/2-tracktor_w/2;
                            y_p[i]-=tracktor_l;
                            type[i]=1;
                            continue;
                        }
                    }
                    if(type[i]==1)
                    {
                        earth_cover01(x_p[i],y_p[i]);
                        y_p[i]--;

  init_tracktor01_f(x_p[i],y_p[i]);

  if(y_p[i]<=start_y+tra_start_l-6)
                        {

  earth_cover01(x_p[i],y_p[i]-7);

  earth_cover01(x_p[i],y_p[i]+3);
                            type[i]=2;
                            continue;
                        }
                    }
                }
                if(count>=num)
                {
                    break;
                }
                delay(delaytime);
            }
        }

    void    tracktor_return04(int    start_x,int
start_y,int des_x,int des_y,int distance,int num)
    {
        int
x_p[tracktor_num_max],y_p[tracktor_num_max],
type[tracktor_num_max],i;
        if(distance<tracktor_l)
        {
            distance=tracktor_l;
        }
        for(i=0;i<num;i++)
        {
            type[num-1-i]=0;

  x_p[num-1-i]=des_x+i*distance+tracktor_l;
            y_p[num-1-i]=des_y-tracktor_w;
            earth_fill03(x_p[i],y_p[i]);
        }
        while(1)
        {
            int count = 0,re=0;

  newmouse(&MouseX,&MouseY,&press);

  re=pressed_anime(x_start,y_start,x_start,y_start
);
            if(re!=0)
            {
                for(i=0;i<num;i++)
                {
                    if(type[i]==0)
                    {

  earth_cover02(x_p[i]+6,y_p[i]);

  earth_cover02(x_p[i]-2,y_p[i]);
                    }
                    if(type[i]==1)
                    {

  earth_cover01(x_p[i],y_p[i]-7);

  earth_cover01(x_p[i],y_p[i]+3);
                    }
                }
                if(mode==0)
                {
                    return;
                }
                break;
            }
            for(i=0;i<num;i++)
            {
                if(type[i]==2)
                {
                    count++;
                    continue;
```

```
			}
			if(type[i]==0)
			{
  earth_cover02(x_p[i]+2,y_p[i]);
				x_p[i]++;
  init_tracktor01_r(x_p[i],y_p[i]);
  if(x_p[i]>=((2*start_x+tra_start_l)/2-tracktor_w/
2))
				{
  earth_cover02(x_p[i]+6,y_p[i]);
  earth_cover02(x_p[i]-2,y_p[i]);
  x_p[i]=(2*start_x+tra_start_l)/2-tracktor_w/2;
					y_p[i]-=tracktor_w;
					type[i]=1;
					continue;
				}
			}
			if(type[i]==1)
			{
				earth_cover01(x_p[i],y_p[i]);
				y_p[i]--;
  init_tracktor01_f(x_p[i],y_p[i]);
  if(y_p[i]<=start_y+tra_start_l-6)
				{
  earth_cover01(x_p[i],y_p[i]-7);
  earth_cover01(x_p[i],y_p[i]+3);
					type[i]=2;
					continue;
				}
			}
		}
		if(count>=num)
		{
			break;
		}
		delay(delaytime);
	}
}

void	tracktor_set_off0(int	start_x,int
start_y,int des_x,int des_y,int distance,int num)
{
  if(start_x<des_x&&start_y<des_y)
  {
  tracktor_set_off001(start_x,start_y,des_x,des_y,
distance,num);
  }
  else if(start_x<des_x&&start_y>des_y)
  {
  tracktor_set_off002(start_x,start_y,des_x,des_y,
distance,num);
  }
  else if(start_x>des_x&&start_y<des_y)
  {
  tracktor_set_off003(start_x,start_y,des_x,des_y,
distance,num);
  }
  else
  {
  tracktor_set_off004(start_x,start_y,des_x,des_y,
distance,num);
  }
}

//start_x<des_x&&start_y<des_y
void	tracktor_set_off001(int	start_x,int
start_y,int des_x,int des_y,int distance,int num)
{
  int			x_p[tracktor_num_max],
y_p[tracktor_num_max],
type[tracktor_num_max],
time[tracktor_num_max], i;
  for (i = 0; i < num; i++)
  {
	type[i] = 0;
	time[i] = i * distance;
	x_p[i] = (2 * start_x + tra_start_l) / 2 -
tracktor_w / 2;
	y_p[i] = start_y + tra_start_d + 1;
  }
```

```
        while (1)
        {
                int count = 0,re=0;

    newmouse(&MouseX,&MouseY,&press);

    re=pressed_anime(x_start,y_start,x_start,y_start
);
                if(re!=0)
                {
                        for(i=0;i<num;i++)
                        {
                                if(type[i]==0)
                                {

    earth_cover01(x_p[i],y_p[i]);

    earth_cover01(x_p[i],y_p[i]+5);
                                }
                                if(type[i]==1)
                                {

    earth_cover02(x_p[i],y_p[i]);

    earth_cover02(x_p[i]+7,y_p[i]);
                                }
                        }
                        if(mode==0)
                        {
                                return;
                        }
                        break;
                }
                for (i = 0; i < num; i++)
                {
                        if (time[i] > 0)
                        {
                                time[i]--;
                                continue;
                        }
                        if (type[i] == 2)
                        {
                                count++;
                                continue;
                        }
                        if (type[i] == 0)
                        {
                                earth_cover01(x_p[i], y_p[i]);
                                y_p[i]++;

init_tracktor02_b(x_p[i],y_p[i]);
                                if(y_p[i]>=des_y)
                                {

earth_cover01(x_p[i],y_p[i]);

earth_cover01(x_p[i],y_p[i]+5);
                                        type[i]=1;
                                        x_p[i]+=tracktor_l;

init_tracktor02_r(x_p[i],y_p[i]);
                                        continue;
                                }
                        }
                        if (type[i] == 1)
                        {
                                earth_cover02(x_p[i], y_p[i]);
                                x_p[i]++;

init_tracktor02_r(x_p[i],y_p[i]);

if(x_p[i]>=des_x+(num-i-1)*distance-tracktor_l)
                                {

earth_cover02(x_p[i],y_p[i]);

earth_cover02(x_p[i]+7,y_p[i]);
                                        type[i]=2;

//init_tracktor02_f(x_p[i],y_p[i]-35);
                                        continue;
                                }
                        }
                }
                if(count>=num)
                {
                        break;
                }
                delay(delaytime);
        }
    }

    //start_x<des_x&&start_y>des_y
    void    tracktor_set_off002(int    start_x,int
start_y,int des_x,int des_y,int distance,int num)
	{
		int
x_p[tracktor_num_max],y_p[tracktor_num_max],
type[tracktor_num_max],time[tracktor_num_max
],i;
		for(i=0;i<num;i++)
		{
			type[i]=0;
			time[i]=i*distance;

  x_p[i]=(2*start_x+tra_start_l)/2-tracktor_w/2;
			y_p[i]=start_y-tra_start_d-1-tracktor_l;
		}
		while(1)
		{
			int count = 0,re=0;

  newmouse(&MouseX,&MouseY,&press);

  re=pressed_anime(x_start,y_start,x_start,y_start
);
			if(re!=0)
			{
				for(i=0;i<num;i++)
				{
					if(type[i]==0)
					{
						earth_cover01(x_p[i],
y_p[i]);
						earth_cover01(x_p[i],
y_p[i] -7);
					}
					if(type[i]==1)
					{

  earth_cover02(x_p[i],y_p[i]);

  earth_cover02(x_p[i]+7,y_p[i]);
					}
				}
				if(mode==0)
				{
					return;
				}
				break;
			}
			for(i=0;i<num;i++)
			{
				if(time[i]>0)
				{
					time[i]--;
					continue;
				}
				if(type[i]==2)
				{
					count++;
					continue;
				}
				if(type[i]==0)
				{
					earth_cover01(x_p[i],y_p[i]);
					y_p[i]--;

init_tracktor02_f(x_p[i],y_p[i]);
					if(y_p[i]<=des_y)
					{

earth_cover01(x_p[i],y_p[i]);

earth_cover01(x_p[i],y_p[i]-7);
						type[i]=1;
						x_p[i]+=tracktor_l;

init_tracktor02_r(x_p[i],y_p[i]);
						continue;
					}
				}
				if(type[i]==1)
				{
					earth_cover02(x_p[i],y_p[i]);
					x_p[i]++;

init_tracktor02_r(x_p[i],y_p[i]);

if(x_p[i]>=des_x+(num-i-1)*distance-tracktor_l)
					{

earth_cover02(x_p[i],y_p[i]);

earth_cover02(x_p[i]+7,y_p[i]);
						type[i]=2;

//init_tracktor02_f(x_p[i],y_p[i]-35);
```

```
                        continue;
                    }
                }
            }
            if(count>=num)
            {
                break;
            }
            delay(delaytime);
        }
    }

    //start_x>des_x&&start_y<des_y
    void    tracktor_set_off003(int    start_x,int
start_y,int des_x,int des_y,int distance,int num)
    {
        int
x_p[tracktor_num_max],y_p[tracktor_num_max],
type[tracktor_num_max],time[tracktor_num_max
],i;
        for(i=0;i<num;i++)
        {
            type[i]=0;
            time[i]=i*distance;

  x_p[i]=(2*start_x+tra_start_l)/2-tracktor_w/2;
            y_p[i]=start_y+tra_start_d+1;
        }
        while(1)
        {
            int count = 0,re=0;

  newmouse(&MouseX,&MouseY,&press);

  re=pressed_anime(x_start,y_start,x_start,y_start
);
            if(re!=0)
            {
                for(i=0;i<num;i++)
                {
                    if(type[i]==0)
                    {
                        earth_cover01(x_p[i],
y_p[i]);
                        earth_cover01(x_p[i],
y_p[i] + 5);
                    }
                    if(type[i]==1)
                    {

earth_cover02(x_p[i]+2,y_p[i]);

earth_cover02(x_p[i]-5,y_p[i]);
                    }
                }
                if(mode==0)
                {
                    return;
                }
                break;
            }
            for(i=0;i<num;i++)
            {
                if(time[i]>0)
                {
                    time[i]--;
                    continue;
                }
                if(type[i]==2)
                {
                    count++;
                    continue;
                }
                if(type[i]==0)
                {
                    earth_cover01(x_p[i],y_p[i]);
                    y_p[i]++;

init_tracktor02_b(x_p[i],y_p[i]);
                    if(y_p[i]>=des_y)
                    {

earth_cover01(x_p[i],y_p[i]);

earth_cover01(x_p[i],y_p[i]+5);
                        type[i]=1;
                        x_p[i]+=tracktor_l;

init_tracktor02_l(x_p[i],y_p[i]);
                        continue;
                    }
                }
                if(type[i]==1)
                {
```

```
  earth_cover02(x_p[i]+5,y_p[i]);
                            x_p[i]--;

  init_tracktor02_l(x_p[i],y_p[i]);

  if(x_p[i]<=des_x+i*distance+tracktor_l)
                            {

  earth_cover02(x_p[i]+2,y_p[i]);

  earth_cover02(x_p[i]-5,y_p[i]);
                                   type[i]=2;

  //init_tracktor02_f(x_p[i]-50,y_p[i]-35);
                                   continue;
                            }
                     }
              }
              if(count>=num)
              {
                     break;
              }
              delay(delaytime);
       }
     }

     //start_x>=des_x&&start_y>=des_y
     void     tracktor_set_off004(int     start_x,int
start_y,int des_x,int des_y,int distance,int num)
     {
       int
x_p[tracktor_num_max],y_p[tracktor_num_max],
type[tracktor_num_max],time[tracktor_num_max
],i;
       for(i=0;i<num;i++)
       {
              type[i]=0;
              time[i]=i*distance;

  x_p[i]=(2*start_x+tra_start_l)/2-tracktor_w/2;
              y_p[i]=start_y-tra_start_d-1-tracktor_l;
       }
       while(1)
       {
              int count = 0,re=0;

  newmouse(&MouseX,&MouseY,&press);

  re=pressed_anime(x_start,y_start,x_start,y_start
);
                     if(re!=0)
                     {
                            for(i=0;i<num;i++)
                            {
                                   if(type[i]==0)
                                   {
                                          earth_cover01(x_p[i],
y_p[i]);
                                          earth_cover01(x_p[i],
y_p[i] -7);
                                   }
                                   if(type[i]==1)
                                   {

  earth_cover02(x_p[i]+2,y_p[i]);

  earth_cover02(x_p[i]-5,y_p[i]);
                                   }
                            }
                            if(mode==0)
                            {
                                   return;
                            }
                            break;
                     }
                     for(i=0;i<num;i++)
                     {
                            if(time[i]>0)
                            {
                                   time[i]--;
                                   continue;
                            }
                            if(type[i]==2)
                            {
                                   count++;
                                   continue;
                            }
                            if(type[i]==0)
                            {
                                   earth_cover01(x_p[i],y_p[i]);
                                   y_p[i]--;

  init_tracktor02_f(x_p[i],y_p[i]);
```

```
                    if(y_p[i]<=des_y)
                    {
 earth_cover01(x_p[i],y_p[i]);
 earth_cover01(x_p[i],y_p[i]-7);
                        type[i]=1;
                        x_p[i]+=tracktor_l;
 init_tracktor02_r(x_p[i],y_p[i]);
                        continue;
                    }
                }
                if(type[i]==1)
                {
 earth_cover02(x_p[i]+7,y_p[i]);
                    x_p[i]--;
 init_tracktor02_l(x_p[i],y_p[i]);
 if(x_p[i]<=des_x+i*distance+tracktor_l)
                    {
 earth_cover02(x_p[i]+2,y_p[i]);
 earth_cover02(x_p[i]-5,y_p[i]);
                        type[i]=2;
 //init_tracktor02_f(x_p[i]-50,y_p[i]-35);
                        continue;
                    }
                }
            }
            if(count>=num)
            {
                break;
            }
            delay(delaytime);
        }
    }

    void tracktor_return0(int start_x,int start_y,int des_x,int des_y,int distance,int num)
    {
      if(start_x<des_x&&start_y>des_y)
      {
 tracktor_return001(start_x,start_y,des_x,des_y,distance,num);
        }
        else if(start_x>des_x&&start_y>des_y)
        {
 tracktor_return002(start_x,start_y,des_x,des_y,distance,num);
        }
        else if(start_x<des_x&&start_y<des_y)
        {
 tracktor_return003(start_x,start_y,des_x,des_y,distance,num);
        }
        else
        {
 tracktor_return004(start_x,start_y,des_x,des_y,distance,num);
        }
      }

    void tracktor_return001(int start_x,int start_y,int des_x,int des_y,int distance,int num)
    {
        int x_p[tracktor_num_max],y_p[tracktor_num_max],type[tracktor_num_max],i;
        if(distance<tracktor_l)
        {
            distance=tracktor_l;
        }
        for(i=0;i<num;i++)
        {
            type[i]=0;
            x_p[i]=des_x+i*distance;
            y_p[i]=des_y;
            earth_fill03(x_p[i],y_p[i]);
            x_p[i]-=tracktor_l;
            y_p[i]-=tracktor_w;
        }
        while(1)
        {
            int count = 0,re=0;
```

```
  newmouse(&MouseX,&MouseY,&press);

  re=pressed_anime(x_start,y_start,x_start,y_start
);
              if(re!=0)
              {
                     for(i=0;i<num;i++)
                     {
                            if(type[i]==0)
                            {

  earth_cover02(x_p[i]+2,y_p[i]);

  earth_cover02(x_p[i]-6,y_p[i]);
                            }
                            if(type[i]==1)
                            {

  earth_cover01(x_p[i],y_p[i]);

  earth_cover01(x_p[i],y_p[i]+5);
                            }
                     }
                     if(mode==0)
                     {
                            return;
                     }
                     break;
              }
              for(i=0;i<num;i++)
              {
                     if(type[i]==2)
                     {
                            count++;
                            continue;
                     }
                     if(type[i]==0)
                     {

  earth_cover02(x_p[i]+2,y_p[i]);
                            x_p[i]--;

  init_tracktor02_l(x_p[i],y_p[i]);

  if(x_p[i]<=((2*start_x+tra_start_l)/2-tracktor_w/
2)+50)
                            {

  earth_cover02(x_p[i]+2,y_p[i]);

  earth_cover02(x_p[i]-6,y_p[i]);

  x_p[i]=(2*start_x+tra_start_l)/2-tracktor_w/2;
                                   type[i]=1;
                                   continue;
                            }
                     }
                     if(type[i]==1)
                     {
                            earth_cover01(x_p[i],y_p[i]);
                            y_p[i]++;

  init_tracktor02_b(x_p[i],y_p[i]);
                            if(y_p[i]>=start_y-48)
                            {

  earth_cover01(x_p[i],y_p[i]);

  earth_cover01(x_p[i],y_p[i]+5);
                                   type[i]=2;
                                   continue;
                            }
                     }
              }
              if(count>=num)
              {
                     break;
              }
              delay(delaytime);
       }
     }

     void tracktor_return002(int start_x,int
start_y,int des_x,int des_y,int distance,int num)
     {
       int
x_p[tracktor_num_max],y_p[tracktor_num_max],
type[tracktor_num_max],i;
       if(distance<tracktor_l)
       {
              distance=tracktor_l;
       }
       for(i=0;i<num;i++)
       {
```

```
                type[num-1-i]=0;
                x_p[num-1-i]=des_x+i*distance;
                y_p[num-1-i]=des_y;
                earth_fill03(x_p[i],y_p[i]);
                x_p[num-1-i]+=tracktor_l;
                y_p[num-1-i]-=tracktor_w;
            }
            while(1)
            {
                int count = 0,re=0;

    newmouse(&MouseX,&MouseY,&press);

    re=pressed_anime(x_start,y_start,x_start,y_start
);
                if(re!=0)
                {
                    for(i=0;i<num;i++)
                    {
                        if(type[i]==0)
                        {

    earth_cover02(x_p[i]+6,y_p[i]);

    earth_cover02(x_p[i]-2,y_p[i]);
                        }
                        if(type[i]==1)
                        {

    earth_cover01(x_p[i],y_p[i]);

    earth_cover01(x_p[i],y_p[i]+5);
                        }
                    }
                    if(mode==0)
                    {
                        return;
                    }
                    break;
                }
                for(i=0;i<num;i++)
                {
                    if(type[i]==2)
                    {
                        count++;
                        continue;
                    }
                    if(type[i]==0)
                    {

    earth_cover02(x_p[i]+2,y_p[i]);
                        x_p[i]++;

    init_tracktor02_r(x_p[i],y_p[i]);

    if(x_p[i]>=((2*start_x+tra_start_l)/2-tracktor_w/
2))
                        {

    earth_cover02(x_p[i]+6,y_p[i]);

    earth_cover02(x_p[i]-2,y_p[i]);

    x_p[i]=(2*start_x+tra_start_l)/2-tracktor_w/2;
                            type[i]=1;
                            continue;
                        }
                    }
                    if(type[i]==1)
                    {
                        earth_cover01(x_p[i],y_p[i]);
                        y_p[i]++;

    init_tracktor02_b(x_p[i],y_p[i]);
                        if(y_p[i]>=start_y-48)
                        {

    earth_cover01(x_p[i],y_p[i]);

    earth_cover01(x_p[i],y_p[i]+5);
                            type[i]=2;
                            continue;
                        }
                    }
                }
                if(count>=num)
                {
                    break;
                }
                delay(delaytime);
            }
        }

        void    tracktor_return003(int    start_x,int
start_y,int des_x,int des_y,int distance,int num)
	{
	  int
x_p[tracktor_num_max],y_p[tracktor_num_max],
type[tracktor_num_max],i;
	  if(distance<tracktor_l)
	  {
		distance=tracktor_l;
	  }
	  for(i=0;i<num;i++)
	  {
		type[i]=0;
		x_p[i]=des_x+i*distance;
		y_p[i]=des_y;
		earth_fill03(x_p[i],y_p[i]);
		x_p[i]-=tracktor_l;
		y_p[i]-=tracktor_w;
	  }
	  while(1)
	  {
		int count = 0,re=0;

  newmouse(&MouseX,&MouseY,&press);

  re=pressed_anime(x_start,y_start,x_start,y_start
);
		if(re!=0)
		{
			for(i=0;i<num;i++)
			{
				if(type[i]==0)
				{

  earth_cover02(x_p[i]+2,y_p[i]);

  earth_cover02(x_p[i]-6,y_p[i]);
				}
				if(type[i]==1)
				{

  earth_cover01(x_p[i],y_p[i]-7);

  earth_cover01(x_p[i],y_p[i]+3);
				}
			}
			if(mode==0)
			{
				return;
			}
			break;
		}
		for(i=0;i<num;i++)
		{
			if(type[i]==2)
			{
				count++;
				continue;
			}
			if(type[i]==0)
			{

  earth_cover02(x_p[i]+2,y_p[i]);
				x_p[i]--;

  init_tracktor02_l(x_p[i],y_p[i]);

  if(x_p[i]<=((2*start_x+tra_start_l)/2-tracktor_w/
2)+50)
				{

  earth_cover02(x_p[i]+2,y_p[i]);

  earth_cover02(x_p[i]-6,y_p[i]);

  x_p[i]=(2*start_x+tra_start_l)/2-tracktor_w/2;
					y_p[i]-=tracktor_l;
					type[i]=1;
					continue;
				}
			}
			if(type[i]==1)
			{
				earth_cover01(x_p[i],y_p[i]);
				y_p[i]--;

  init_tracktor02_f(x_p[i],y_p[i]);

  if(y_p[i]<=start_y+tra_start_l-6)
				{

  earth_cover01(x_p[i],y_p[i]-7);

  earth_cover01(x_p[i],y_p[i]+3);
					type[i]=2;
```

```
                        continue;
                    }
                }
            }
            if(count>=num)
            {
                break;
            }
            delay(delaytime);
        }
    }

    void    tracktor_return004(int    start_x,int
start_y,int des_x,int des_y,int distance,int num)
    {
        int
x_p[tracktor_num_max],y_p[tracktor_num_max],
type[tracktor_num_max],i;
        if(distance<tracktor_l)
        {
            distance=tracktor_l;
        }
        for(i=0;i<num;i++)
        {
            type[num-1-i]=0;

  x_p[num-1-i]=des_x+i*distance+tracktor_l;
            y_p[num-1-i]=des_y-tracktor_w;
            earth_fill03(x_p[i],y_p[i]);
        }
        while(1)
        {
            int count = 0,re=0;

  newmouse(&MouseX,&MouseY,&press);

  re=pressed_anime(x_start,y_start,x_start,y_start
);
            if(re!=0)
            {
                for(i=0;i<num;i++)
                {
                    if(type[i]==0)
                    {

  earth_cover02(x_p[i]+6,y_p[i]);
  earth_cover02(x_p[i]-2,y_p[i]);
                    }
                    if(type[i]==1)
                    {

  earth_cover01(x_p[i],y_p[i]-7);

  earth_cover01(x_p[i],y_p[i]+3);
                    }
                }
                if(mode==0)
                {
                    return;
                }
                break;
            }
            for(i=0;i<num;i++)
            {
                if(type[i]==2)
                {
                    count++;
                    continue;
                }
                if(type[i]==0)
                {

  earth_cover02(x_p[i]+2,y_p[i]);
                    x_p[i]++;

  init_tracktor02_r(x_p[i],y_p[i]);

  if(x_p[i]>=((2*start_x+tra_start_l)/2-tracktor_w/
2))
                    {

  earth_cover02(x_p[i]+6,y_p[i]);

  earth_cover02(x_p[i]-2,y_p[i]);

  x_p[i]=(2*start_x+tra_start_l)/2-tracktor_w/2;
                        y_p[i]-=tracktor_w;
                        type[i]=1;
                        continue;
                    }
                }
                if(type[i]==1)
                {
                        earth_cover01(x_p[i],y_p[i]);
                        y_p[i]--;

  init_tracktor02_f(x_p[i],y_p[i]);

  if(y_p[i]<=start_y+tra_start_l-6)
                        {

  earth_cover01(x_p[i],y_p[i]-7);

  earth_cover01(x_p[i],y_p[i]+3);
                            type[i]=2;
                            continue;
                        }
                    }
                }
                if (count >= num)
                {
                    break;
                }
                delay(delaytime);
            }
        }


        void  picker_anime(int  start_x,int  start_y,int
*des_x,int *des_y,int distance,int num)
        {
        #define cost_time 50
         int                          x_p[tracktor_num_max],
y_p[tracktor_num_max],
type[tracktor_num_max],\

time[tracktor_num_max],count[tracktor_num_ma
x], i,co_time[tracktor_num_max],xy[2];
         xy[0]=start_x,xy[1]=start_y;
         // if(des_x[0]<=0)
         //    return;
         for (i = 0; i < num; i++)
         {
                type[i] = 0;
                count[i]=0;
                co_time[i]=0;
                time[i] = i * distance;
                x_p[i] = (2 * start_x + tra_start_l) / 2 -
tracktor_w / 2;
                y_p[i] = start_y + tra_start_d + 1;
                draw_copak(des_x[i],des_y[i]);
            }
            while(1)
            {
                int total=0,re=0;

  newmouse(&MouseX,&MouseY,&press);

  re=pressed_anime(x_start,y_start,x_start,y_start
);
                if(re!=0)
                {
                    for(i=0;i<num;i++)
                    {
                        if(type[i]==0)
                        {
                            earth_cover01(x_p[i],
y_p[i]);
                            earth_cover01(x_p[i],
y_p[i] + 5);
                        }
                        if(type[i]==1)
                        {

  earth_cover02(x_p[i],y_p[i]);

  earth_cover02(x_p[i]+7,y_p[i]);
                        }
                    }
                    if(mode==0)
                    {
                        return;
                    }
                    break;
                }
                for(i=0;i<num;i++)
                {
                    if(des_x[i]<=0)
                    {
                        count[i]=3;
                    }
                    if(count[i]==0)
                    {

  picker_set_off(x_p,y_p,start_x,start_y,des_x[i],d
es_y[i]+12,time,i,count,type);
                        continue;
            }
            else if(count[i]==1)
            {
                // if(co_time[i]==0)
                // {
                //
  init_picker_f(des_x[i],des_y[i]);
                //    co_time[i]++;
                //    continue;
                // }
                if(co_time[i]<cost_time)
                {

  init_picker_f(des_x[i],des_y[i]);
                    co_time[i]++;
                    continue;
                }
                else
if(co_time[i]>=cost_time)
                {
                    type[i]=0;
                    count[i]++;

  x_p[i]=des_x[i],y_p[i]=des_y[i];
                    earth_cover01(x_p[i],
y_p[i]-7);
                    earth_cover01(x_p[i],
y_p[i] + 5);
                    continue;
                }
            }
            else if(count[i]==2)
            {

  picker_return(x_p,y_p,start_x,start_y,des_x[i],de
s_y[i]-5-tracktor_l,i,count,type);
                continue;
            }
            else if(count[i]>=3)
            {
                total++;
                continue;
            }
        }
        if(total==num)
        {
            break;
            }
            draw_setoff(xy);
            delay(delaytime);
        }
        return;
    }

    void   picker_set_off(int   *x_p,int   *y_p,int
start_x,int start_y,int des_x,int des_y,int *time,int
num,int *count,int* type)
    {
        if (start_x < des_x && start_y < des_y)
        {
            picker_set_off01( x_p,y_p,des_x, des_y,
time, num,count,type);
        }
        else if (start_x < des_x && start_y > des_y)
        {
            picker_set_off02(x_p,y_p, des_x, des_y,
time, num,count,type);
        }
        else if (start_x > des_x && start_y < des_y)
        {
            picker_set_off03(x_p,y_p, des_x, des_y,
time, num,count,type);
        }
        else
        {
            picker_set_off04(x_p,y_p, des_x, des_y,
time, num,count,type);
        }
    }

    // start_x<des_x&&start_y<des_y
    void  picker_set_off01(  int  *x_p,int  *y_p,int
des_x,  int  des_y,  int  *time,  int  i,int  *count,int
*type)
    {
        //          int          x_p[tracktor_num_max],
y_p[tracktor_num_max],
type[tracktor_num_max],
time[tracktor_num_max], i;
        // for (i = 0; i < num; i++)
        // {
        //    type[i] = 0;
        //    time[i] = i * distance;
        //    x_p[i] = (2 * start_x + tra_start_l) / 2 -
```

```
tracktor_w / 2;
	//	y_p[i] = start_y + tra_start_d + 1;
	// }
	// //clrmous(MouseX,MouseY);
	// while (1)
	// {
	// int re=0;
	// newmouse(&MouseX,&MouseY,&press);
	//
re=pressed_anime(x_start,y_start,x_start,y_start);
	// if(re!=0)
	// {
	//	if(type[i]==0)
	//	{
	//		earth_cover01(x_p[i], y_p[i]);
	//		earth_cover01(x_p[i], y_p[i] + 5);
	//	}
	//	if(type[i]==1)
	//	{
	//		earth_cover02(x_p[i],y_p[i]);
	//		earth_cover02(x_p[i]+7,y_p[i]);
	//	}
	//	if(mode==0)
	//	{
	//		return;
	//	}
	// }
	if (time[i] > 0)
	{
		time[i]--;
		return;
	}
	if (type[i] == 2)
	{
		count[i]++;
		return;
	}
	if (type[i] == 0)
	{
		earth_cover01(x_p[i], y_p[i]);
		y_p[i]++;
		init_picker_b(x_p[i], y_p[i]);
		if (y_p[i] >= des_y)
		{
			earth_cover01(x_p[i], y_p[i]);
			earth_cover01(x_p[i], y_p[i] + 5);
			type[i] = 1;
			x_p[i] += tracktor_l;
			init_picker_r(x_p[i], y_p[i]);
		}
		return;
	}
	if (type[i] == 1)
	{
		earth_cover02(x_p[i], y_p[i]);
		x_p[i]++;
		init_picker_r(x_p[i], y_p[i]);
		if (x_p[i] >= des_x-tracktor_l)
		{
			earth_cover02(x_p[i],y_p[i]);
			earth_cover02(x_p[i]+7,y_p[i]);
			//x_p[i]=des_x,y_p[i]=des_y;
			type[i]=2;

 //init_tracktor01_f(x_p[i],y_p[i]-35);
		}
		return;
	}
	// if (count >= num)
	// {
	//	break;
	// }
	//}
}


// start_x<des_x&&start_y>des_y
void picker_set_off02(int *x_p,int *y_p, int
des_x, int des_y, int *time, int i,int *count,int*
type)
{
	//	int	x_p[tracktor_num_max],
y_p[tracktor_num_max],
type[tracktor_num_max],
time[tracktor_num_max], i;
	// for (i = 0; i < num; i++)
	// {
	//	type[i] = 0;
	//	time[i] = i * distance;
	//	x_p[i] = (2 * start_x + tra_start_l) / 2 -
tracktor_w / 2;
	//	y_p[i] = start_y - tra_start_d - 1 -
tracktor_l;
	// }
	// //clrmous(MouseX,MouseY);
	// while (1)
	// {
	// int re=0;
	// newmouse(&MouseX,&MouseY,&press);
	//
re=pressed_anime(x_start,y_start,x_start,y_start);
	// if(re!=0)
	// {
	//	if(type[i]==0)
	//	{
	//		earth_cover01(x_p[i], y_p[i]);
	//		earth_cover01(x_p[i], y_p[i] -7);
	//	}
	//	if(type[i]==1)
	//	{
	//		earth_cover02(x_p[i],y_p[i]);
	//		earth_cover02(x_p[i]+7,y_p[i]);
	//	}
	//	if(mode==0)
	//	{
	//		return;
	//	}
	// }
	if (time[i] > 0)
	{
		time[i]--;
		return;
	}
	if (type[i] == 2)
	{
		count[i]++;
		return;
	}
	if (type[i] == 0)
	{
		earth_cover01(x_p[i], y_p[i]);
		y_p[i]--;
		init_picker_f(x_p[i], y_p[i]);
		if (y_p[i] <= des_y)
		{
			earth_cover01(x_p[i], y_p[i]);
			earth_cover01(x_p[i], y_p[i] - 7);
			type[i] = 1;
			x_p[i] += tracktor_l;
			init_picker_r(x_p[i], y_p[i]);
		}
		return;
	}
	if (type[i] == 1)
	{
		earth_cover02(x_p[i], y_p[i]);
		x_p[i]++;
		init_picker_r(x_p[i], y_p[i]);
		if (x_p[i] >= des_x -tracktor_l)
		{
			earth_cover02(x_p[i],y_p[i]);
			earth_cover02(x_p[i]+7,y_p[i]);
			//x_p[i]=des_x,y_p[i]=des_y;
			type[i]=2;

  //init_tracktor01_f(x_p[i],y_p[i]-35);
		}
		return;
	}
	// if (count >= num)
	// {
	//	break;
	// }
}

// start_x>des_x&&start_y<des_y
void picker_set_off03(int *x_p,int *y_p, int des_x, int des_y, int *time, int i,int *count,int* type)
{
	//	int	x_p[tracktor_num_max], y_p[tracktor_num_max], type[tracktor_num_max], time[tracktor_num_max], i;
	// for (i = 0; i < num; i++)
	// {
	//	type[i] = 0;
	//	time[i] = i * distance;
	//	x_p[i] = (2 * start_x + tra_start_l) / 2 - tracktor_w / 2;
	//	y_p[i] = start_y + tra_start_d + 1;
	// }
	// //clrmous(MouseX,MouseY);
	// while (1)
	// {
	// int re=0;
	// newmouse(&MouseX,&MouseY,&press);
	//
re=pressed_anime(x_start,y_start,x_start,y_start);
```

```
// if(re!=0)
// {
//     if(type[i]==0)
//     {
//             earth_cover01(x_p[i], y_p[i]);
//             earth_cover01(x_p[i], y_p[i] + 5);
//     }
//     if(type[i]==1)
//     {
//             earth_cover02(x_p[i]+2,y_p[i]);
//             earth_cover02(x_p[i]-5,y_p[i]);
//     }
//     if(mode==0)
//     {
//             return;
//     }
// }
if (time[i] > 0)
{
        time[i]--;
        return;
}
if (type[i] == 2)
{
        count[i]++;
        return;
}
if (type[i] == 0)
{
        earth_cover01(x_p[i], y_p[i]);
        y_p[i]++;
        init_picker_b(x_p[i], y_p[i]);
        if (y_p[i] >= des_y)
        {
                earth_cover01(x_p[i], y_p[i]);
                earth_cover01(x_p[i], y_p[i] + 5);
                type[i] = 1;
                x_p[i] += tracktor_l;
                init_picker_l(x_p[i], y_p[i]);
        }
        return;
}
if (type[i] == 1)
{
        earth_cover02(x_p[i] + 5, y_p[i]);
        x_p[i]--;
        init_picker_l(x_p[i], y_p[i]);
```

```
                if (x_p[i] <= des_x +tracktor_l)
                {
                        earth_cover02(x_p[i]+2,y_p[i]);
                        earth_cover02(x_p[i]-5,y_p[i]);
                        x_p[i]=des_x,y_p[i]=des_y;
                        type[i]=2;

    //init_tracktor01_f(x_p[i]-50,y_p[i]-35);
                }
                return;
        }
        // if (count >= num)
        // {
        //     break;
        // }
    }

    // start_x>=des_x&&start_y>=des_y
    void picker_set_off04( int *x_p,int *y_p,int
des_x, int des_y, int *time, int i,int *count,int*
type)
    {
        //      int     x_p[tracktor_num_max],
y_p[tracktor_num_max],
type[tracktor_num_max],
time[tracktor_num_max], i;
        // for (i = 0; i < num; i++)
        // {
        //     type[i] = 0;
        //     time[i] = i * distance;
        //     x_p[i] = (2 * start_x + tra_start_l) / 2 -
tracktor_w / 2;
        //     y_p[i] = start_y - tra_start_d - 1 -
tracktor_l;
        // }
        // //clrmous(MouseX,MouseY);
        // while (1)
        // {
        // int count = 0,re=0;
        // newmouse(&MouseX,&MouseY,&press);
        //
re=pressed_anime(x_start,y_start,x_start,y_start);
        // if(re!=0)
        // {
        //     if(type[i]==0)
        //     {
        //             earth_cover01(x_p[i], y_p[i]);
    //          earth_cover01(x_p[i], y_p[i] -7);
    //      }
    //      if(type[i]==1)
    //      {
    //          earth_cover02(x_p[i]+2,y_p[i]);
    //          earth_cover02(x_p[i]-5,y_p[i]);
    //      }
    //      if(mode==0)
    //      {
    //          return;
    //      }
    // }
    if (time[i] > 0)
    {
        time[i]--;
        return;
    }
    if (type[i] == 2)
    {
        count[i]++;
        return;
    }
    if (type[i] == 0)
    {
    earth_cover01(x_p[i], y_p[i]);
    y_p[i]--;
    init_picker_f(x_p[i], y_p[i]);
    if (y_p[i] <= des_y)
    {
        earth_cover01(x_p[i], y_p[i]);
        earth_cover01(x_p[i], y_p[i] - 7);
        type[i] = 1;
        x_p[i] += tracktor_l;
        init_picker_r(x_p[i], y_p[i]);
    }
    return;
  }
  if (type[i] == 1)
  {
    earth_cover02(x_p[i] + 7, y_p[i]);
    x_p[i]--;
    init_picker_l(x_p[i], y_p[i]);
    if (x_p[i] <= des_x + tracktor_l)
    {
        earth_cover02(x_p[i]+2,y_p[i]);
        earth_cover02(x_p[i]-5,y_p[i]);
        x_p[i]=des_x,y_p[i]=des_y;
            type[i]=2;

  //init_tracktor01_f(x_p[i]-50,y_p[i]-35);
      }
      return;
    }
    //      if (count >= num)
    //      {
    //          break;
    //      }
    //      delay(delaytime);
    // }
    }

    void picker_return(int *x_p,int *y_p,int start_x,int start_y,int des_x,int des_y,int num,int *count,int* type)
    {
    if(start_x<des_x&&start_y>des_y)
    {
      picker_return01(x_p,y_p,start_x,start_y ,num,count,type);
    }
    else if(start_x>des_x&&start_y>des_y)
    {
      picker_return02( x_p,y_p,start_x,start_y ,num,count,type);
    }
    else if(start_x<des_x&&start_y<des_y)
    {
      picker_return03(x_p,y_p,start_x,start_y ,num,count,type);
    }
    else
    {
      picker_return04(x_p,y_p,start_x,start_y ,num,count,type);
    }
    }

    void picker_return01( int *x_p,int *y_p,int start_x,int start_y, int i,int *count,int *type)
    {
    //  int x_p[tracktor_num_max],y_p[tracktor_num_max],type[tracktor_num_max],i;
    // if(distance<tracktor_l)
```

```
	// {
	//	distance=tracktor_l;
	// }
	// for(i=0;i<num;i++)
	// {
	//	type[i]=0;
	//	x_p[i]=des_x+i*distance;
	//	y_p[i]=des_y;
	//	earth_fill03(x_p[i],y_p[i]);
	//	x_p[i]-=tracktor_l;
	//	y_p[i]-=tracktor_w;
	// }
	// while(1)
	// {
	//	int count = 0,re=0;
	//
  newmouse(&MouseX,&MouseY,&press);
	//
  re=pressed_anime(x_start,y_start,x_start,y_start
);
	//	if(re!=0)
	//	{
	//		for(i=0;i<num;i++)
	//		{
	//			if(type[i]==0)
	//			{
	//
  earth_cover02(x_p[i]+2,y_p[i]);
	//
  earth_cover02(x_p[i]-6,y_p[i]);
	//			}
	//			if(type[i]==1)
	//			{
	//
  earth_cover01(x_p[i],y_p[i]);
	//
  earth_cover01(x_p[i],y_p[i]+5);
	//			}
	//		}
	//		if(mode==0)
	//		{
	//			return;
	//		}
	//		break;
	//	}
	  // for(i=0;i<num;i++)
	  // {
		if(type[i]==2)
		{
			count[i]++;
			return;
		}
		if(type[i]==0)
		{
			earth_cover02(x_p[i]+2,y_p[i]);
			x_p[i]--;
			init_picker01_l(x_p[i],y_p[i]);

  if(x_p[i]<=((2*start_x+tra_start_l)/2-tracktor_w/
2)+50)
			{
				earth_cover02(x_p[i]+2,y_p[i]);
				earth_cover02(x_p[i]-6,y_p[i]);

  x_p[i]=(2*start_x+tra_start_l)/2-tracktor_w/2;
				type[i]=1;
			}
			return;
		}
		if(type[i]==1)
		{
			earth_cover01(x_p[i],y_p[i]);
			y_p[i]++;
			init_picker01_b(x_p[i],y_p[i]);
			if(y_p[i]>=start_y-48)
			{
				earth_cover01(x_p[i],y_p[i]);
				earth_cover01(x_p[i],y_p[i]+5);
				type[i]=2;
			}
			return;
		}
			// if(count>=num)
			// {
			//	break;
			// }
			// delay(delaytime);
	}

	void picker_return02(int *x_p,int *y_p,int
start_x,int start_y, int i,int *count,int *type)
	{
	  //					int
x_p[tracktor_num_max],y_p[tracktor_num_max],
type[tracktor_num_max],i;
        // if(distance<tracktor_l)
        // {
        //     distance=tracktor_l;
        // }
        // for(i=0;i<num;i++)
        // {
        //     type[num-1-i]=0;
        //     x_p[num-1-i]=des_x+i*distance;
        //     y_p[num-1-i]=des_y;
        //     earth_fill03(x_p[i],y_p[i]);
        //     x_p[num-1-i]+=tracktor_l;
        //     y_p[num-1-i]-=tracktor_w;
        // }
        // while(1)
        // {
        //     int count = 0,re=0;
        //
  newmouse(&MouseX,&MouseY,&press);
        //
  re=pressed_anime(x_start,y_start,x_start,y_start
);
        //     if(re!=0)
        //     {
        //         for(i=0;i<num;i++)
        //         {
        //             if(type[i]==0)
        //             {
        //
  earth_cover02(x_p[i]+6,y_p[i]);
        //
  earth_cover02(x_p[i]-2,y_p[i]);
        //             }
        //             if(type[i]==1)
        //             {
        //
  earth_cover01(x_p[i],y_p[i]);
        //
  earth_cover01(x_p[i],y_p[i]+5);
        //             }
        //         }
        //         if(mode==0)
        //         {
        //             return;
        //         }
        //         break;
        //     }
```

```
    //     for(i=0;i<num;i++)
    //     {
    if(type[i]==2)
    {
        count[i]++;
        return;
    }
    if(type[i]==0)
    {
        earth_cover02(x_p[i]+2,y_p[i]);
        x_p[i]++;
        init_picker01_r(x_p[i],y_p[i]);

  if(x_p[i]>=((2*start_x+tra_start_l)/2-tracktor_w/
2))
        {
            earth_cover02(x_p[i]+6,y_p[i]);
            earth_cover02(x_p[i]-2,y_p[i]);

  x_p[i]=(2*start_x+tra_start_l)/2-tracktor_w/2;
            type[i]=1;
        }
        return;
    }
    if(type[i]==1)
    {
        earth_cover01(x_p[i],y_p[i]);
        y_p[i]++;
        init_picker01_b(x_p[i],y_p[i]);
        if(y_p[i]>=start_y-48)
        {
            earth_cover01(x_p[i],y_p[i]);
            earth_cover01(x_p[i],y_p[i]+5);
            type[i]=2;
        }
        return;
    }
        // }
    //     if(count>=num)
    //     {
    //         break;
    //     }
    //     delay(delaytime);
    // }
}

void   picker_return03(int   *x_p,int   *y_p,int
start_x,int start_y,    int i,int *count,int *type)
        {
          //                                                   int
x_p[tracktor_num_max],y_p[tracktor_num_max],
type[tracktor_num_max],i;
          // if(distance<tracktor_l)
          // {
          //     distance=tracktor_l;
          // }
          // for(i=0;i<num;i++)
          // {
          //     type[i]=0;
          //     x_p[i]=des_x+i*distance;
          //     y_p[i]=des_y;
          //     earth_fill03(x_p[i],y_p[i]);
  //     x_p[i]-=tracktor_l;
          //     y_p[i]-=tracktor_w;
          // }
          // while(1)
          // {
          //     int count = 0,re=0;
          //
  newmouse(&MouseX,&MouseY,&press);
          //
  re=pressed_anime(x_start,y_start,x_start,y_start
);
          //     if(re!=0)
          //     {
          //            for(i=0;i<num;i++)
          //            {
          //                   if(type[i]==0)
          //                   {
          //
  earth_cover02(x_p[i]+2,y_p[i]);
          //
  earth_cover02(x_p[i]-6,y_p[i]);
          //                   }
          //                   if(type[i]==1)
          //                   {
          //
  earth_cover01(x_p[i],y_p[i]-7);
          //
  earth_cover01(x_p[i],y_p[i]+3);
          //                   }
          //            }
          //            if(mode==0)
          //            {
          //                          return;
          //            }
          //            break;
          //     }
          //     for(i=0;i<num;i++)
          //     {
          if(type[i]==2)
          {
                 count[i]++;
                 return;
          }
          if(type[i]==0)
          {
                 earth_cover02(x_p[i]+2,y_p[i]);
                 x_p[i]--;
                 init_picker01_l(x_p[i],y_p[i]);

  if(x_p[i]<=((2*start_x+tra_start_l)/2-tracktor_w/
2)+50)
                 {
                        earth_cover02(x_p[i]+2,y_p[i]);
                        earth_cover02(x_p[i]-6,y_p[i]);

  x_p[i]=(2*start_x+tra_start_l)/2-tracktor_w/2;
                        y_p[i]-=tracktor_l;
                        type[i]=1;
                 }
                 return;
          }
          if(type[i]==1)
          {
                 earth_cover01(x_p[i],y_p[i]);
                 y_p[i]--;
                 init_picker01_f(x_p[i],y_p[i]);
                 if(y_p[i]<=start_y+tra_start_l-6)
                 {
                        earth_cover01(x_p[i],y_p[i]-7);
                        earth_cover01(x_p[i],y_p[i]+3);
                        type[i]=2;
                 }
                 return;
          }
          //     }
          //     if(count>=num)
          //     {
          //            break;
          //     }
```

```
    //    delay(delaytime);
    // }
   }

   void   picker_return04(int   *x_p,int   *y_p,int
start_x,int start_y,    int i,int *count,int *type)
   {
    //                                               int
x_p[tracktor_num_max],y_p[tracktor_num_max],
type[tracktor_num_max],i;
    // if(distance<tracktor_l)
    // {
    //    distance=tracktor_l;
    // }
    // for(i=0;i<num;i++)
    // {
    //    type[num-1-i]=0;
    //
  x_p[num-1-i]=des_x+i*distance+tracktor_l;
    //    y_p[num-1-i]=des_y-tracktor_w;
    //    earth_fill03(x_p[i],y_p[i]);
    // }
    // while(1)
    // {
    //    int count = 0,re=0;
    //
  newmouse(&MouseX,&MouseY,&press);
    //
  re=pressed_anime(x_start,y_start,x_start,y_start
);
    //    if(re!=0)
    //    {
    //          for(i=0;i<num;i++)
    //          {
    //                if(type[i]==0)
    //                {
    //
  earth_cover02(x_p[i]+6,y_p[i]);
    //
  earth_cover02(x_p[i]-2,y_p[i]);
    //                }
    //                if(type[i]==1)
    //                {
    //
  earth_cover01(x_p[i],y_p[i]-7);
    //
  earth_cover01(x_p[i],y_p[i]+3);
    //                }
    //          }
    //          if(mode==0)
    //          {
    //                return;
    //          }
    //          break;
    //    }
    //    for(i=0;i<num;i++)
    //    {
    if(type[i]==2)
    {
          count[i]++;
          return;
    }
    if(type[i]==0)
    {
          earth_cover02(x_p[i]+2,y_p[i]);
          x_p[i]++;
          init_picker01_r(x_p[i],y_p[i]);

  if(x_p[i]>=((2*start_x+tra_start_l)/2-tracktor_w/
2))
          {
                earth_cover02(x_p[i]+6,y_p[i]);
                earth_cover02(x_p[i]-2,y_p[i]);

  x_p[i]=(2*start_x+tra_start_l)/2-tracktor_w/2;
                y_p[i]-=tracktor_w;
                type[i]=1;
          }
          return;
    }
    if(type[i]==1)
    {
          earth_cover01(x_p[i],y_p[i]);
          y_p[i]--;
          init_picker01_f(x_p[i],y_p[i]);
          if(y_p[i]<=start_y+tra_start_l-6)
          {
                earth_cover01(x_p[i],y_p[i]-7);
                earth_cover01(x_p[i],y_p[i]+3);
                type[i]=2;
          }
          return;
    }
    //    }
//      if(count>=num)
//      {
//              break;
//      }
//      delay(delaytime);
// }
}
```

13、 WELCOME.C

```
#include "COMMON.H"
#include "WELCOME.H"
#include "mouse.h"
#include "PARAMETE.H"
#include "HOME.H"
#include "LOGFUN.H"
/*void main()
{
    int gdriver,gmode;
    gdriver=DETECT;
    initgraph(&gdriver,&gmode,"..\\BORLANDC\\BGI
");
    draw_wel();
    closegraph();
}*/

// draw the buttons in welcome page
void draw_wel_buttons(void)
{
    clrmous(MouseX, MouseY);
    cleardevice();
    setbkcolor(WHITE);
    puthz(180, 30, "棉花采摘模拟系统", 32, 32,
BLUE);
    setlinestyle(0, 0, 1);
    setcolor(DARKGRAY);

    setfillstyle(1, 13);
    rectangle(450, 110, 550, 170);
    puthz(470, 130, "编辑参数", 16, 16, BLUE);
    rectangle(450, 210, 550, 270);
    puthz(470, 230, "开始模拟", 16, 16, BLUE);
    rectangle(450, 310, 550, 370);
    puthz(470, 330, "参数列表", 16, 16, BLUE);
    rectangle(450, 410, 550, 470);
    puthz(460, 430, "帮助及说明", 16, 16, BLUE);

    line(300, 130, 360, 100);
    line(360, 100, 420, 130);
    line(300, 130, 420, 130);
    rectangle(310, 130, 410, 200);
    puthz(330, 150, "仓库管理", 16, 16, BLUE);

    quit();
    // last();
}

// draw the tractor in welcome page
void draw_wel_tractors()
{
    // The wheels
    int i, d;
    setcolor(BROWN);
    setfillstyle(1, BLUE);
    bar(94, 185, 100, 215);
    bar(200, 185, 206, 215);
    bar(94, 265, 100, 295);
    bar(200, 265, 206, 295);
    setfillstyle(1, RED);

    // The rectangle of the machine
    bar(100, 150, 200, 310);
    rectangle(105, 155, 195, 305);
    setfillstyle(1, YELLOW);
    setlinestyle(0, 0, 1);

    // The small bar
    for (i = 0, d = 80; i < 8; i++)
    {
        bar(d - 2, 125, d + 2, 155);
        rectangle(d - 3, 125, d + 3, 155);
        d += 20;
    }
    setlinestyle(0, 0, 3);
    bar(80, 130, 220, 150);
    rectangle(80, 130, 220, 150);
    for (i = 0, d = 100; i < 6; i++)
    {
        line(d, 130, d, 150);
        d += 20;
    }
    setfillstyle(1, RED);
    bar(130, 120, 170, 160);
    rectangle(130, 120, 170, 160);
}
```

```
// draw the cotton field in welcome page
void draw_wel_cofield()
{
	// The cotton field
	int i, d;
	setcolor(BROWN);
	rectangle(50, 100, 300, 400);
	setlinestyle(0, 0, 3);
	for (d = 58; d <= 300; d += 20)
	{
		for (i = 100; i <= 400; i += 2)
		{
			int temp = rand() % 2;
			line(d + temp, i, d + temp, i);
		}
	}
	for (i = 0; i < 1000; i++)
	{
		int t1 = rand() % 250, t2 = rand() % 300;
		line(t1 + 50, t2 + 100, t1 + 50, t2 + 100);
	}
	setfillstyle(1, BROWN);
	bar(78, 130, 222, 400);
	setcolor(WHITE);
	for (i = 0; i < 100; i++)
	{
		int t1 = rand() % 144, t2 = rand() % 270;
		line(t1 + 78, t2 + 130, t1 + 78, t2 + 130);
	}
	/*setfillstyle(1,BROWN);
	bar(50,100,300,400);
	setlinestyle(0,0,3);
	setcolor(WHITE);
	for(i=50;i<=300;i+=6)
	{
		for(d=100;d<=400;d+=2)
		{
			int temp=rand()%2;
			line(i+temp,d,i+temp,d);
			//temp=rand()%2;
			//line(i+temp,d,i+temp,d);
		}
	}*/
}

// enter the edit page
void enter_next()
{
	static int flag = 0, flag1 = 1;
	int dian[8] = {300, 130, 360, 100, 420, 130, 300,
130};
	if (mouse_press(450, 110, 550, 170) == 0 ||
mouse_press(450, 210, 550, 270) == 0 ||
mouse_press(450, 310, 550, 370) == 0 ||
mouse_press(450, 410, 550, 470) == 0 ||
		mouse_press(310, 130, 410, 200) == 0 ||
mouse_press(0, 0, 40, 30) == 0 || mouse_press(0, 450, 40,
480) == 0)
	{
		MouseS = 0;
	}
	if (mouse_press(450, 110, 550, 170) == 2 ||
mouse_press(450, 210, 550, 270) == 2 ||
mouse_press(450, 310, 550, 370) == 2 ||
mouse_press(450, 410, 550, 470) == 2 ||
		mouse_press(310, 130, 410, 200) == 2 ||
mouse_press(0, 0, 40, 30) == 2 || mouse_press(0, 450, 40,
480) == 2)
	{
		MouseS = 1;
	}

	if (mouse_press(450, 110, 550, 170) == 2)
	{
		if (flag1 == 1)
		{
			clrmous(MouseX, MouseY);
			setfillstyle(1, CYAN);
			bar(450, 110, 550, 170);
			puthz(470, 130, "编辑参数", 16, 16,
BLUE);

			flag = 0;
			flag1 = 0;
		}
	}
	else if (mouse_press(450, 210, 550, 270) == 2)
	{
		if (flag1 == 1)
		{
			clrmous(MouseX, MouseY);
			setfillstyle(1, CYAN);
			bar(450, 210, 550, 270);
```

```
                puthz(470, 230, "开始模拟", 16, 16,
BLUE);
                flag = 0;
                flag1 = 0;
            }
        }
        else if (mouse_press(450, 310, 550, 370) == 2)
        {
            if (flag1 == 1)
            {
                clrmous(MouseX, MouseY);
                setfillstyle(1, CYAN);
                bar(450, 310, 550, 370);
                puthz(470, 330, "参数列表", 16, 16,
BLUE);
                flag = 0;
                flag1 = 0;
            }
        }
        else if (mouse_press(450, 410, 550, 470) == 2)
        {
            if (flag1 == 1)
            {
                clrmous(MouseX, MouseY);
                setfillstyle(1, CYAN);
                bar(450, 410, 550, 470);
                puthz(460, 430, "帮助及说明", 16, 16,
BLUE);
                flag = 0;
                flag1 = 0;
            }
        }
        else if (mouse_press(310, 130, 410, 200) == 2)
        {
            if (flag1 == 1)
            {
                clrmous(MouseX, MouseY);
                setfillstyle(1, CYAN);
                bar(310, 130, 410, 200);
                puthz(330, 150, "仓库管理", 16, 16,
BLUE);
                fillpoly(4, dian);
                rectangle(310, 130, 410, 200);
                flag = 0;
                flag1 = 0;
            }
        }
```

```
        else if (flag == 0)
        {
            clrmous(MouseX, MouseY);
            setfillstyle(1, 0);
            bar(450, 110, 550, 170);
            bar(450, 210, 550, 270);
            bar(450, 310, 550, 370);
            bar(450, 410, 550, 470);

            rectangle(450, 110, 550, 170);
            puthz(470, 130, "编辑参数", 16, 16, BLUE);
            rectangle(450, 210, 550, 270);
            puthz(470, 230, "开始模拟", 16, 16, BLUE);
            rectangle(450, 310, 550, 370);
            puthz(470, 330, "参数列表", 16, 16, BLUE);
            rectangle(450, 410, 550, 470);
            puthz(460, 430, "帮助及说明", 16, 16,
BLUE);

            setfillstyle(1, 0);
            bar(300, 100, 420, 200);
            puthz(330, 150, "仓库管理", 16, 16, BLUE);
            rectangle(310, 130, 410, 200);
            line(300, 130, 360, 100);
            line(360, 100, 420, 130);
            line(300, 130, 420, 130);

            flag = 1;
            flag1 = 1;
        }

        // Enter the edit page
        if (mouse_press(450, 110, 550, 170) == 1)
        {
            mode = 2;
        }

        // Enter the simulation page
        if (mouse_press(450, 210, 550, 270) == 1)
        {
            mode = 3;
        }

        // Enter the past arguments
        if (mouse_press(450, 310, 550, 370) == 1)
        {
            mode = 4;
            // draw_past01();
```

```
	}

	// Enter the help arguments
	if (mouse_press(450, 410, 550, 470) == 1)
	{
		mode = 5;
		// draw_help01();
	}

	// Enter the home page
	if (mouse_press(310, 130, 410, 200) == 1)
	{
		// draw_home01();
		mode = 1;
		/*else
		{
			draw_home00();
		}*/
	}

	// Exit the program
	if (mouse_press(0, 0, 40, 30) == 1)
	{
		wr_h();
		free(h);
		exit(0);
	}
}

/*Draw the whole welcome page*/
void draw_wel()
{
	draw_wel_buttons();
	draw_wel_cofield();
	draw_wel_tractors();
	quit();
}

void quit(void)
{
	setfillstyle(1,LIGHTBLUE);
	bar(0,0,40,30);
	puthz(3,10,"退出",16,16,WHITE);
}

void skip(void)
{
		setfillstyle(1,LIGHTBLUE);
		bar(585,450,625,480);
		puthz(625-40+3,480-30+10,"          跳          过
",16,16,WHITE);
	}

	void next(void)
	{
		setfillstyle(1,LIGHTBLUE);
		bar(585,450,625,480);
		puthz(625-40+3,480-30+10,"          下          页
",16,16,WHITE);
	}

	void last(void)
	{
		setfillstyle(1,LIGHTBLUE);
		bar(0,450,40,480);
		puthz(4,480-30+10,"返回",16,16,WHITE);
	}

	/*void   text_input(char   *str,int   x1,int   y1,int   x2,int
y2,int t_x,int t_y,int t_size)
	{
		char temp,*p;
		int                                                    i,
n=t_x,get,arr[10]={p_0,p_1,p_2,p_4,p_5,p_6,p_7,p_8,p_9}
;
		clrmous(MouseX,MouseY);
		p=str;
		setfillstyle(1,WHITE);
		setcolor(DARKGRAY);
		bar(x1,y1,x2,y2);
		while(bioskey(1))
		{
			get=bioskey(0);
		}
		while(*p!='\0')
		{
			if (get==p_Enter)
			{
				break;
			}
			for(i=0;i<10;i++)
			{
				if(arr[i]==get)
				{
```

```
                temp=i+'0';
            }
        }
        *p=temp;
        p++;
        outtextxy(n,t_y,&temp);
        get=bioskey(0);
        n+=t_size;
    }
}*/

void input_text(char *id, int x, int y, int charnum, int
color, int flag)
{ // flag==1    显示

#define h 32
#define w 18
#define space 0
#define SX x + 5 // START X
#define SY y - 5

    //int k = 0;

    int i = 0;
    char t;
    clrmous(MouseX, MouseY);
    setfillstyle(SOLID_FILL, color);
    setlinestyle(SOLID_LINE, 0, NORM_WIDTH);
    setcolor(DARKGRAY);
    settextstyle(TRIPLEX_FONT, HORIZ_DIR, 4);
    settextjustify(LEFT_TEXT, TOP_TEXT);

    while (bioskey(1))
    {
        t = bioskey(0);
    }

    while (*(id + i) != '\0')
        i++;
    line(SX+i*w,SY,SX+i*w,SY+h);
    while (1)
    {
        setfillstyle(1,WHITE);
        t = bioskey(0);
        if (i<charnum)
        {
            if  (t!='\n'&&  t!='\r'&&t  !='  '&&  t  !=
033)
            { // 033:Esc
                if (t != '\b')
                {
                    *(id + i) = t;
                    *(id + i + 1) = '\0';
                    bar(SX+i*w-1+space,SY - 1,SX
+i*w+1+space,SY+h); //遮盖光标
                    if (flag)
                        outtextxy(SX + i * 18, SY,
id + i); //输出刚输入的字符 t
                    else
                    {
                        outtextxy(SX + i * 18, SY,
"*");
                    }
                    i++;
                    line(SX + i * w + space, SY, SX
+i*w +space,SY+h);
                }
                else if (t == '\b' && i > 0)
                {

bar(SX+i*w-1+space,SY-1,SX+i*w+1+space,SY+h);   // 遮 盖
光标
                    i--;
//减少一个字数
                    bar(SX+i*w,SY,SX + i * w + w,
SY + h);                    //遮盖文字
                    line(SX+i*w+space, SY, SX + i
* w + space, SY + h); //绘制光标
                    *(id + i) = '\0';
                    *(id + i + 1) = '\0';
                }
            }
            else
            {
                bar(SX+i * w - 1 + space,SY - 1, SX
+ i * w + 1+ space, SY+ h); //遮盖光标
                break;
            }
        }
        else
        {
            if (t!='\n'&&t!='\r' && t != ' '&&t!=033)
            { // 033:Esc
                if (t == '\b' && i > 0)
```

```
                    {
                        bar(SX+i*w-1+space,SY-1,SX
+ i * w + 1 + space, SY + h); //遮盖光标
                        i--;
//减少一个字数
                        bar(SX  +  i  *  w,  SY,  SX  +  i
*w+w,SY+h);                              //遮盖文字
                        line(SX + i * w + space, SY, SX
+ i * w + space,SY + h);                 //绘制光标
                        *(id + i) = '\0';
                        *(id + i + 1) = '\0';
                    }
                }
                else
```

```
                    {
                        bar(SX+i*w
-1+space,SY-1,SX+i*w+1+space,SY+h); //遮盖光标
                        break;
                    }
                }
            }
            //return i;
        }
```

<1>引用代码

引用来自学长的鼠标、汉字及部分写屏代码

八、 时间安排

第一周：进行需求分析并学习主要共性知识

第二周：完成需求分析并初步掌握共性知识

第三周：完成分工，建立代码远程仓库，提交需求分析报告，开始编程

第四周：完成欢迎界面全部内容，完成各辅界面框架内容

第五周：完成全部页面基本内容，中期验收

第六周：优化界面设计，优化绘图算法

第七周：继续优化算法，做抗压调试及部分改进

第八周：程序调试，整理报告，准备最后验收

九、 工作分配及代码量

梁栢杰：

注册登录及用户管理，农田农机参数的输入及保存，仓库信息的输入及保存，参数信息处理，帮助与说明界面，以及上述对应的界面设计及画图。

有效代码行数：3301 行

冯天瑞：

棉花采摘农机行进路径规划，不同形状土地提取与生成，农田参数对应产量与用时的计算，仓库管理，以及上述对应的界面设计及画图。

有效代码行数：4252 行

十、 组员感想

组员冯天瑞感想

C 语言课程设计极大地锻炼了我的代码能力，并直观地教会了我现代计算机语言构建工程项目的基本方法，设计程序结构的基本思路以及项目管理思维。在完成 C 课设的过程中，我不仅对上个学期的 C 语言程序设计课程内容有了更加深刻的理解，也借此学习了 bc 下的图形编程和工程项目建立，并简单地学习与应用了 git 版本管理。在此过程中，有艰难 debug 的困顿，也有程序最终正常运行的喜悦，我

和队友克服一道又一道难关，从 0 开始一点一点构建作品，在多次实验中找出不足之处并加以改进，在老师和学长学姐的指导下扩展思路，设计更加人性化智能化的人机交互方式。与此同时，我在完成课设的过程中通过网络查找关于棉花机械化采摘的相关资料，收集各省不同种类棉花相关数据，尝试着构建出更加符合实际的棉花田地模式和收割机路径规划方法，并估算土地收益。

学习编程的第一步总是艰难而笨拙的，前期的 C 语言课程以及 C 语言课程设计就是这万里征途的第一步。而这第一步的回报也是极其丰厚的，由此我们体验到了程序设计的魅力，能够调整自我，敢于应对更多挑战。

最后非常感谢我的队友梁栢杰同学在文件操作上的支持与帮助，同时也要感谢各位老师与热心的学长学姐们的意见与指导，也希望在接下来能够继续提升自己的编程水平。

组员梁栢杰感想

为期近三个月的 c 语言课程设计就此落幕，在这段时间里，我受益良多。c 课设极大地提升了我的编程能力，让我从程序的黑框中跳了出来，我的程序开始有了颜色变化，也更加贴近了实际的应用。在我刚开始学编程的时候，我有想过在这个黑框中究竟能干些什么，现在，c 课设告诉了我答案。在完成 c 课设的道路上，我遇到了许多的困难，我曾因为一个不起眼的 bug 而被困扰了三天，也曾为它的功能不理想而苦恼。总的来说，从一片虚无中一步一个脚印，最终走完这

道路，可以说是我永远不会忘记的一段旅程。再者，c 课设也让我接触到了 github 这个管理代码的工具，虽然我对这个工具使用并不熟练，甚至弄丢了两次代码，但我仍然认为他是一个多人合作项目必不可少的工具。

整个任务做下来，我认为最重要的就是耐心，细致，和坚韧的心态。无论是找 bug，还是写代码，有了这些东西，往往就会事半功倍。除此之外，我认为最重要的就是经验，整个 c 课设的过程也是积累经验的过程，一个问题，你遇到过，并解决了，那他对你来说就不是问题，但对其他人来说这或许就能要了他们的命。同时，学长所提供的各种教材和祖传代码也是我们完成 c 课设的关键。

最后，非常感谢在 c 课设过程中帮助过我的老师，助教，学长和同学，他们或多或少都对我脆弱的心灵施以了援手，为我做 c 课设增加了动力。我还要感谢我的队友冯天瑞，他让我可以完全专注于我自己的任务，并在我需要帮助时对我提供帮助。

十一、参考文献

1. 王士元. C 高级实用程序设计. 北京: 清华大学出版社. 1996
2. 周纯杰，刘正林等. 标准 C 语言程序及应用. 武汉: 华中科技大学出版社. 2008
3. 周纯杰，何顶新等.程序设计教程 用 c/c++语言编程.北京：机械工业出版社.2016