1.OJ排名中的个人总耗时

#include<stdio.h>

struct problemScore

{

int hour;

int minute;

int second;

int errorCount;

}score[11];

int main(void)

{

int i,hourSum=0,minuteSum=0,secondSum=0;

//依次输入A题--J题ＡＣ时的“时、分、秒、错误提交次数“, 并存储在score[1]--score[10]中。

for(i=1;i<=10;i++)

{

scanf("%d %d %d %d",&score[i].hour,&score[i].minute,&score[i].second,&score[i].errorCount);

}

//在begin和end之间编写代码

//实现根据 A题--J题的“时、分、秒、错误提交次数“ 累加计算出 总耗时penalty的“时、分、秒“，并存入score[0]中。

/\*\*\*\*\*\*\*\*\*\*\*\* begin \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

int submit;

for(i=1; i<=10; i++)

{

submit=0;

if(score[i].hour!=0||score[i].minute!=0||score[i].second!=0)

{

submit=20\*score[i].errorCount;

}

hourSum+=score[i].hour;

minuteSum+=score[i].minute+submit;

secondSum+=score[i].second;

if( secondSum>=60)

{

secondSum=secondSum-60;minuteSum++;

}

if(minuteSum>=60&&minuteSum<120)

{

minuteSum=minuteSum-60;hourSum++;

}

if(minuteSum>=120)

{

hourSum=hourSum+minuteSum/60;minuteSum=minuteSum%60;

}

}

score[0].hour=hourSum;

score[0].minute=minuteSum;

score[0].second=secondSum;

/\*\*\*\*\*\*\*\*\*\*\*\* end \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//第1行，输出计算的总耗时penalty，按"时:分:秒“的格式表示。

printf("Penalty: %02d:%02d:%02d\n",score[0].hour,score[0].minute,score[0].second);

//第2行至第11行，输出A题至J题的耗时，按"时:分:秒（－错误提交次数）“的格式表示。

//但若“时、分、秒”全为０，则不输出“时、分、秒”。若错误提交次数为０，则不输出错误提交次数。

for(i=1;i<=10;i++)

{

printf("%c: ",'A'+i-1);

if( score[i].hour || score[i].minute || score[i].second )

printf("%02d:%02d:%02d",score[i].hour,score[i].minute,score[i].second );

if(score[i].errorCount)

printf("(-%d)",score[i].errorCount );

printf("\n");

}

return 0;

}

2.通讯录排序

#include <stdio.h>

#include<string.h>

struct adult{

char birth[50];

char name[50];

char num[50];

};

int main(void)

{

int i,n,j,index,k;

struct adult s1[50],temp;

scanf("%d",&n);

for(i=0;i<n;i++)

{

scanf("%s %s %s",s1[i].name,s1[i].birth,s1[i].num);

}

for(j=0;j<n-1;j++)

{

index=j;

for(k=j+1;k<n;k++)

{

if(strcmp(s1[index].birth,s1[k].birth)>0)

{

index=k;

}

}

temp=s1[index];

s1[index]=s1[j];

s1[j]=temp;

}

for(i=0;i<n;i++)

{

printf("%s %s %s\n",s1[i].name,s1[i].birth,s1[i].num);

}

return 0;

}

3.单链表结点删除

#include <stdio.h>

#include <stdlib.h>

struct ListNode {

int data;

struct ListNode \*next;

};

struct ListNode \*readlist();

struct ListNode \*deletem( struct ListNode \*L, int m );

void printlist( struct ListNode \*L )

{

struct ListNode \*p = L;

while (p) {

printf("%d ", p->data);

p = p->next;

}

printf("\n");

}

int main()

{

int m;

struct ListNode \*L = readlist();

scanf("%d", &m);

L = deletem(L, m);

printlist(L);

return 0;

}

4.建立学生信息链表

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

struct stud\_node {

int num;

char name[20];

int score;

struct stud\_node \*next;

};

struct stud\_node \*head, \*tail;

void input()

{

struct stud\_node \*p;

int num;

while(scanf("%d", &num)!=EOF&&num) {

p = (struct stud\_node \*)malloc(sizeof(struct stud\_node));

p -> num = num;

scanf("%s%d", p -> name, &(p -> score));

if(head==NULL) {

head = tail = p;

} else {

tail -> next = p;

tail = p;

}

}

}

int main()

{

struct stud\_node \*p;

head = tail = NULL;

input();

for ( p = head; p != NULL; p = p->next )

printf("%d %s %d\n", p->num, p->name, p->score);

return 0;

}

5.链表逆置

#include <stdio.h>

#include <stdlib.h>

struct ListNode {

int data;

struct ListNode \*next;

};

struct ListNode \*createlist(); /\*裁判实现，细节不表\*/

struct ListNode \*reverse( struct ListNode \*head );

void printlist( struct ListNode \*head )

{

struct ListNode \*p = head;

while (p) {

printf("%d ", p->data);

p = p->next;

}

printf("\n");

}

int main()

{

struct ListNode \*head;

head = createlist();

head = reverse(head);

printlist(head);

return 0;

}

struct ListNode \*reverse( struct ListNode \*head )

{

struct ListNode \*newhead=NULL,\*ptr,\*p=head;

for(;p!=NULL;)

{

ptr=(struct ListNode \*)malloc(sizeof(struct ListNode));

ptr->data=p->data;ptr->next=NULL;

if(newhead==NULL)

newhead=ptr;

else

{

ptr->next=newhead;

newhead=ptr;

}

p=p->next;

}

return newhead;

}

1. 链表拼接

#include <stdio.h>

#include <stdlib.h>

struct ListNode {

int data;

struct ListNode \*next;

};

struct ListNode \*createlist(); /\*裁判实现，细节不表\*/

struct ListNode \*mergelists(struct ListNode \*list1, struct ListNode \*list2);

void printlist( struct ListNode \*head )

{

struct ListNode \*p = head;

while (p) {

printf("%d ", p->data);

p = p->next;

}

printf("\n");

}

int main()

{

struct ListNode \*list1, \*list2;

list1 = createlist();

list2 = createlist();

list1 = mergelists(list1, list2);

printlist(list1);

return 0;

}

struct ListNode \*mergelists(struct ListNode \*list1, struct ListNode \*list2)

{

int data;

struct ListNode \*c=(struct ListNode \*)malloc(sizeof(struct ListNode));

struct ListNode \*head=c;

while(list1!=NULL&&list2!=NULL)

{

if(list1->data<list2->data)

{

head->next=list1;

list1=list1->next;

}

else{

head->next=list2;

list2=list2->next;

}

head=head->next;

}

if(list1!=NULL) head->next=list1;

else head->next=list2;

return c->next;

}

7.逆序建立链表

#include <stdio.h>

#include <stdlib.h>

struct ListNode {

int data;

struct ListNode \*next;

};

struct ListNode \*createlist();

int main()

{

struct ListNode \*p, \*head = NULL;

head = createlist();

for ( p = head; p != NULL; p = p->next )

printf("%d ", p->data);

printf("\n");

return 0;

}

struct ListNode \*createlist()

{

struct ListNode \*p,\*head=NULL,\*tail=NULL;

int data;

scanf("%d",&data);

while(data>0)

{

p=(struct ListNode \*)malloc(sizeof(struct ListNode));

p->data=data;

p->next=NULL;

if(tail==NULL)

{

tail=p;

}

else{

p->next=head;

}

head=p;

scanf("%d",&data);

}

return head;

}

8.奇数值节点链表

#include <stdio.h>

#include <stdlib.h>

struct ListNode {

int data;

struct ListNode \*next;

};

struct ListNode \*readlist();

struct ListNode \*getodd( struct ListNode \*\*L );

void printlist( struct ListNode \*L )

{

struct ListNode \*p = L;

while (p) {

printf("%d ", p->data);

p = p->next;

}

printf("\n");

}

int main()

{

struct ListNode \*L, \*Odd;

L = readlist();

Odd = getodd(&L);

printlist(Odd);

printlist(L);

return 0;

}

struct ListNode \*mergelists(struct ListNode \*list1, struct ListNode \*list2)

{

int data;

struct ListNode \*c=(struct ListNode \*)malloc(sizeof(struct ListNode));

struct ListNode \*head=c;

while(list1!=NULL&&list2!=NULL)

{

if(list1->data<list2->data)

{

head->next=list1;

list1=list1->next;

}

else{

head->next=list2;

list2=list2->next;

}

head=head->next;

}

if(list1!=NULL) head->next=list1;

else head->next=list2;

return c->next;

}

9.数组元素和

#include<stdio.h>

int s[1005][1005],t[1005][1005];

int main()

{

int n,m,i=1,j=1,q;

scanf("%d%d",&n,&m);

while(i<=n)

{

for(j=1;j<=m;j++)

{

scanf("%d",&s[i][j]);

t[i][j]=s[i][j]+t[i-1][j]+t[i][j-1]-t[i-1][j-1];

}

i++;

}

scanf("%d",&q);

while(q!=0)

{

int x1,y1,x2,y2;

scanf("%d%d%d%d",&x1,&y1,&x2,&y2);

x1++,x2++,y1++,y2++;

printf("%d\n",t[x2][y2]-t[x2][y1-1]-t[x1-1][y2]+t[x1-1][y1-1]);

q--;

}

}

10.学生链表处理

#include <stdio.h>

#include <stdlib.h>

struct stud\_node {

int num;

char name[20];

int score;

struct stud\_node \*next;

};

struct stud\_node \*createlist();

struct stud\_node \*deletelist( struct stud\_node \*head, int min\_score );

int main()

{

int min\_score;

struct stud\_node \*p, \*head = NULL;

head = createlist();

scanf("%d", &min\_score);

head = deletelist(head, min\_score);

for ( p = head; p != NULL; p = p->next )

printf("%d %s %d\n", p->num, p->name, p->score);

return 0;

}

struct stud\_node \*createlist(){

int num;

struct stud\_node \*p,\*head=NULL,\*tail=NULL;

scanf("%d",&num);

while(num){

p=(struct stud\_node\*)malloc(sizeof(struct stud\_node));

p->num=num;

scanf("%s %d",p->name,&p->score);

if(head==NULL){

head=tail=p;

}else{

tail->next=p;

tail=tail->next;

}

tail->next=NULL;

scanf("%d",&num);

}

return head;

}

struct stud\_node \*deletelist( struct stud\_node \*head, int min\_score ){

struct stud\_node \*p,\*tail;

while(head!=NULL&&head->score<min\_score) head=head->next;//表头的score小于min\_score,再往下找。

if(head==NULL) return NULL;//没有信息。

tail=head;

p=tail->next;

while(p!=NULL){

if(p->score<min\_score){

tail->next=p->next;

free(p);

}else tail=p;//往下找。

p=tail->next;

}

return head;

}

1. 最小的x

#include"stdio.h"

int main()

{

long long int a,b,c,d,i,j,n,k,sum,flag=0;

while(scanf("%lld %lld %lld %lld",&a,&b,&c,&d)!=EOF)

{

for(i=1;i<82;i++)

{

n=(i+a)\*(i+b)\*(i+c)\*(i+d);

k=n;

sum=0;

flag=0;

while(k!=0)

{

sum+=k%10;

k=k/10;

}

if(sum==i)

{

flag=1;

break;

}

}

if(flag==1)

printf("%lld\n",n);

if(flag==0)

printf("no\n");

}return 0;

}

12.统计专业人数

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

struct ListNode {

char code[8];

struct ListNode \*next;

};

struct ListNode \*createlist(); /\*裁判实现，细节不表\*/

int countcs( struct ListNode \*head );

int main()

{

struct ListNode \*head;

head = createlist();

printf("%d\n", countcs(head));

return 0;

}

int countcs( struct ListNode \*head )

{

struct ListNode \*p;

int count=0;

p=head;

while(p!=NULL)

{

if((p->code[1]=='0')&&(p->code[2]=='2'))

{

count++;

}

p=p->next;

}

return count;

}

13.隐私管理系统

#include<stdio.h>

#include<string.h>

#include<stdlib.h>

typedef struct yinsi

{

int number;

char position[50];

char description[20];

char name[30];

char key[20];

struct yinsi \*next;

}guanli;

void menu\_select();

guanli \*input(guanli \*head);

void search(guanli \*head);

void modifyimformation(guanli \*head);

void lookup(guanli \*head);

guanli \*delect(guanli \*head);

void fileWrite(guanli \*head);

void fileRead(guanli \*head);

void secrect\_addition(guanli \*head);

void encrypt(guanli \*pwd);

void menu\_select()

{

char s[3]; int c;

printf("\n \* \* \* \* \* \* \* \*主菜单\* \* \* \* \* \*\n");

printf(" 1.输入记录\n");

printf(" 2.浏览记录\n");

printf(" 3.按账号名查找记录并显示\n");

printf(" 4.修改记录\n");

printf(" 5.删除记录\n");

printf(" 6.将记录存入文件\n");

printf(" 7.从文件中读入所有记录\n");

printf(" 8.退出\n");

}

guanli \*head=NULL;

int main()

{

int flag=2;

int i,q;

char d[7],e[50],f[30];

printf("欢迎使用隐私管理系统！\n\n\n\n");

printf("请输入密码（初始密码为txp123）:");

char a[7]="txp123";

printf("\n输入密码:");

scanf("%s",d);

for(i=1;i<=3;i++)

{

if(strcmp(a,d)==0) break;

else {printf("密码错误！\n");}

if(i>2) {printf("您是非法用户！");exit(0); }

printf("输入密码:");

scanf("%s",d);

}

while(1)

{

menu\_select();

printf(" \n请选择1-8的操作：");

scanf("%d",&q);

switch(q)

{

case 1:head=input(head);break;

case 2:lookup(head);break;

case 3:search(head);break;

case 4:modifyimformation(head);break;

case 5:head=delect(head);break;

case 6:fileWrite(head);break;

case 7:fileRead(head);break;

case 8:exit(0);

}

}

}

guanli \*input(guanli \*head)

{

guanli \*p,\*tail=NULL;

int y;

int num;

printf("最多只能录入100个人的信息!\n");

while(printf("输入数字编号:"),scanf("%d",&num)!=EOF)

{

p=(guanli\*)malloc(sizeof(guanli));

p->number=num;

printf("\n请输入账号位置（50字符以内）:");

scanf("%s",p->position);

printf("\n请输入账号描述（20字符以内）:");

scanf("%s",p->description);

printf("\n请输入账号名称（30字符以内）:");

scanf("%s",p->name);

printf("\n请输入账号密码（20字符以内）:");

scanf("%s",p->key);

if(head==NULL)

head=tail=p;

else

{

tail->next=p;

tail=p;

}

printf("\n\n是否结束录入，'1'继续，'0'结束\n");

scanf("%d",&y);

if(y==0) break;

}

tail->next=NULL;

printf("\n\n自动返回菜单!!!!!\n\n\n");

return head;

}

void search(guanli \*head)

{

char k[50];

guanli \*p=head;

printf("\n\n请输入你要查找的账号名称:");

scanf("%s",k);

while(p!=NULL)

{

if(strcmp(k,p->name)==0)

{

printf("- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - \n");

printf("编号 账号位置 账号描述 账号名称 密码\n");

printf("%d %s %s %s %s\n",p->number,p->position,p->description,p->name,p->key);

printf("- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - \n");

printf("\n\n为您查找成功!!!!!\n\n\n");

break;

}

else p=p->next;

}

if(p==NULL)

{

printf("对不起，没有该账号的资料!\n");

}

printf("\n\n自动返回菜单!!!!!\n\n\n");

}

void lookup(guanli \*head)

{

guanli \*p=head;

printf("为您查找中，请稍后.....\n");

printf("- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - \n");

while(p!=NULL)

{

printf("编号 账号位置 账号描述 账号名称 密码\n");

printf("%d %s %s %s %s\n",p->number,p->position,p->description,p->name,p->key);

p=p->next;

}

printf("- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - \n");

printf("\n\n自动返回菜单!!!!!\n\n\n");

}

guanli \*delect(guanli \*head)

{

guanli \*p=head,\*tail=NULL;

int a;

printf("请输入你要删除的信息数字编号:\n");

scanf("%d",&a);

head=NULL;

while(p!=NULL)

{

if(p->number!=a)

{

if(head==NULL)

head=p;

else

tail->next=p;

tail=p;

}

p=p->next;

if(tail!=NULL) tail->next=NULL;

}

printf("信息删除成功!!!!!\n\n");

printf("\n\n自动返回菜单!!!!!\n\n\n");

return head;

}

void modifyimformation(guanli \*head)

{

guanli \*p=head;

int a,c;

char b[50];

printf("请输入你要修改的信息数字编号:\n");

scanf("%d",&a);

while(p->next!=NULL)

{

if(p->number=a) break;

else

{

p=p->next;

}

}

printf("修改账号位置请按'1'\n");

printf("修改账号描述请按'2'\n");

printf("修改账号名称请按'3'\n");

printf("修改密码请按'4'\n");

scanf("%d",&c);

printf("输入修改的信息:");

scanf("%s",b);

switch(c)

{

case 1: strcpy(p->position,b);break;

case 2: strcpy(p->description,b);break;

case 3: strcpy(p->name,b);break;

case 4: strcpy(p->key,b);break;

}

printf("\n信息修改成功!!!!");

printf("- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - \n");

printf("编号 账号位置 账号描述 账号名称 密码\n");

printf("%d %s %s %s %s\n",p->number,p->position,p->description,p->name,p->key);

printf("- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - \n");

printf("\n\n自动返回菜单!!!!!\n\n\n");

}

void fileWrite(guanli \*head)

{

guanli \*p=head;

FILE \*w =fopen("baocun.txt","w");

if(w==NULL)

{

printf("打开文件失败!");

return;

}

while(p)

{

fprintf(w,"%d %s %s %s %s\n",p->number,p->position,p->description,p->name,p->key);

p=p->next;

}

fclose(w);

printf("- - -所有记录已经成功保存至文件中!- - -\n");

printf("\n\n自动返回菜单!!!!!\n\n\n");

}

void fileRead(guanli \*head)

{

guanli \*newhead =(guanli\*)malloc(sizeof(guanli)),\*x;

int number;

char position[50];

char description[20];

char name[30];

char key[20];

guanli \*p;

guanli \*q;

p=q=newhead;

FILE \* r= fopen("baocun.txt","r");

if(r==NULL)

{

printf("打开文件失败!");

}

while(fscanf(r,"%d%s%s%s%s\n",&number,position,description,name,key)!=EOF)

{

q= (guanli\*)malloc(sizeof(guanli));

q->number=number;

strcpy(q->position,position);

strcpy(q->description,description);

strcpy(q->name,name);

strcpy(q->key,key);

p->next=q;

p=q;

}

p->next=NULL;

x=newhead;

printf("编号 账号位置 账号描述 账号名称 密码\n");

while(x!=NULL)

{

printf("%d %s %s %s %s\n",x->number,x->position,x->description,x->name,x->key);

x=x->next;

}

printf("- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - \n");

printf("已成功从文件导入数据！！！\n");

printf("\n\n自动返回菜单!!!!!\n\n\n");

}

void secrect\_addition(guanli \*head)

{

FILE \*fp;

int i;

guanli \*su=head;

if((fp=fopen("baocun.txt","w"))==NULL){

printf("File open error!\n");

exit(0);

}

while(su->next!=NULL)

scanf("%s%s",su->position,su->description);

encrypt(su);

fprintf(fp,"%s %s\n",su->name,su->key);

if(fclose(fp))

{

printf("Can not close the file!\n");

exit(0);

}

}

void encrypt(guanli \*pwd)

{

int i;

for(i=0;i<strlen(pwd->key);i++)

pwd->key[i]=pwd->key[i]^15;

}

14.编程实现根据指定文本生成电子印章

//#include <CONIO.H>

#include <stdio.h>

#include <stdlib.h>

#include <math.h>

#include <malloc.h>

#include <string.h>

typedef unsigned char BYTE;

typedef unsigned short WORD;

typedef unsigned long DWORD;

/\*\*\*\* The file header of bmp file 位图文件头\*\*\*\*\*/

//#include <pshpack2.h> //This file turns 2 byte packing of structures on, then sizeof(BITMAPFILEHEADER)=14, otherwise sizeof(BITMAPFILEHEADER)=16

typedef struct tagBITMAPFILEHEADER {

WORD bfType;

DWORD bfSize;

WORD bfReserved1;

WORD bfReserved2;

DWORD bfoffBits;

} BITMAPFILEHEADER;

//#include <poppack.h> //This file turns packing of structures off

/\*\*\*\* The information header of bmp file 位图信息头\*\*\*\*\*/

typedef struct tagBITMAPINFOHEADER {

DWORD biSize;

DWORD biWidth;

DWORD biHeight;

WORD biPlanes;

WORD biBitCount;

DWORD biCompress;

DWORD biSizeImage;

DWORD biXPeIsPerMeter;

DWORD biYPeIsPerMeter;

DWORD biCIrUsed;

DWORD biClrImprotant;

} BITMAPINFOHEADER;

/\*\*\*\* The RGB data of bmp file 图像RGB数据\*\*\*\*\*/

typedef struct tagRGBDATA{

BYTE rgbBlue;

BYTE rgbGreen;

BYTE rgbRed;

} RGBDATA;

int main(int argc, char \*argv[])

{

RGBDATA \*bmpData=NULL; //图像数据指针

FILE \*fp, \*fq; //BMP文件指针

long i,j,k;

//================================================================================================

long width=112; //图像宽度

long height=113; //图像高度

long dataSize=width\*height;

BITMAPFILEHEADER bmfHeader;

BITMAPINFOHEADER bmiHeader;

if(argc<2)

{

printf("\n 请指定您要生成的BMP文件名！\n");

printf("\n 方法1：在cmd窗口输入： demo xxxx.bmp　后回车执行！\n");

printf("\n 方法2：在VC的工程设置中添加参数： xxxx.bmp　后编译运行！\n");

printf("\n 功能： 根据设定红绿蓝三色分量的值，控制显示不同的彩色条纹！\n\n");

exit(0);

}

printf("\n \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("\n 感谢您使用BMP图片文件制作DEMO程序 \n");

printf("\n 作者：向德生 \n");

printf("\n 日期：2010年5月12日 \n");

printf("\n 目的：编制印章生成程序时，供参考！ \n");

printf("\n 功能：根据设定红绿蓝三色分量的值，控制显示不同的彩色条纹！\n\n");

printf("\n \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n\n");

//第一步：用命令行中给出的文件名新建一BMP文件，此时还是一个空文件

if((fp=fopen(argv[1], "wb+"))==NULL)

{

printf("Cannot open BMP file!");

exit(0);

}

if((fq=fopen("helloketty.txt", "r"))==NULL)

{

printf("Cannot open BMP file!");

exit(0);

}

//第二步：置文件头数据并写入BMP文件

bmfHeader.bfType=0x4d42;

bmfHeader.bfSize=14+40+width\*height\*3;

bmfHeader.bfReserved1=0;

bmfHeader.bfReserved2=0;

bmfHeader.bfoffBits=0x36;

fwrite(&bmfHeader, sizeof(BITMAPFILEHEADER), 1, fp);

//第三步：置信息头数据并写入BMP文件

bmiHeader.biSize=40;

bmiHeader.biWidth=width;

bmiHeader.biHeight=height;

bmiHeader.biPlanes=1;

bmiHeader.biBitCount=24;

bmiHeader.biCompress=0;

bmiHeader.biSizeImage=width\*height\*3;

bmiHeader.biXPeIsPerMeter=0;

bmiHeader.biYPeIsPerMeter=0;

bmiHeader.biCIrUsed=0;

bmiHeader.biClrImprotant=0;

fwrite(&bmiHeader, sizeof(BITMAPINFOHEADER), 1, fp);

//第四步：置图像RGB数据并写入BMP文件

//分配足够内存，让bmpData指向这块内存，用于存放图像各象素点的RGB分量值

if((bmpData=(RGBDATA\*)malloc(width\*height\*3))==NULL)

{

printf("bmpData memory malloc error!");

}

char a[112][113];//根据xds2.txt文件的行列数确定

for(i=0;i<112;i++)

for(j=0;j<113;j++){

fscanf(fq,"%c",&a[i][j]);

}//将xds2.txt的内容写入数组

fclose(fq);//记得关闭文件

//先在bmpData所指内存中置图像RGB数据，然后将所有数据写入BMP文件

for(i=0;i<112;i++)

{

for(j=0;j<113;j++)

{

k=(height-i-1)\*width + j;

if(a[i][j]=='X'||i<=2||j<=2||i>=height-3||j>=width-3)

{

bmpData[k].rgbBlue=0;

bmpData[k].rgbGreen=0;

bmpData[k].rgbRed=255;//红色分量值为：255

}

else if(a[i][j]=='\_')

{

bmpData[k].rgbBlue=255;

bmpData[k].rgbGreen=255;

bmpData[k].rgbRed=255;//白色

}

}

}

fwrite(bmpData,sizeof(RGBDATA),dataSize,fp);//将bmpData所指内存中的RGB数据一次性写入BMP文件

printf("\n 恭喜您！BMP文件已经成功生成！\n");

printf("\n 请在当前目录下查看生成的BMP文件%s\n\n",argv[1]);

free(bmpData); //释放bmpData所指的内存空间

bmpData=NULL; //置bmpData为空指针

fclose(fp); //关闭fp所指文件

}