**期末考核项目——源代码**

**说明1：不得更改本文档的排版格式包括字体、大小、间距等。**

**说明2：按程序执行顺序将完整源代码粘贴于此。**

**说明3**：行号自动编号：每行源代码前有连续行号并去掉空行，保留注释、说明等。

**说明4**：不为凑代码行数，故意将代码断行（一行代码不恰当地分开在两行或以上）。

**说明5**：每个模块代码前可以添加算法、功能、技术等简要说明文字。

============================================================================================

1. //CGrid.h格子类头文件
2. #pragma once
3. class CGrid
4. {
5. public:
6. bool bFlaged;
7. };
8. //help.h 帮助对话框头文件
9. class help : public CDialogEx
10. {
11. DECLARE\_DYNAMIC(help)
12. public:
13. help(CWnd\* pParent = nullptr); // 标准构造函数
14. virtual ~help();
15. // 对话框数据
16. #ifdef AFX\_DESIGN\_TIME
17. enum { IDD = IDD\_help };
18. #endif
19. protected:
20. virtual void DoDataExchange(CDataExchange\* pDX); // DDX/DDV 支持
21. DECLARE\_MESSAGE\_MAP()
22. };
23. //framework.h头文件
24. #pragma once
25. #ifndef VC\_EXTRALEAN
26. #define VC\_EXTRALEAN // 从 Windows 头中排除极少使用的资料
27. #endif
28. #include "targetver.h"
29. #define \_ATL\_CSTRING\_EXPLICIT\_CONSTRUCTORS // 某些 CString 构造函数将是显式的
30. // 关闭 MFC 的一些常见且经常可放心忽略的隐藏警告消息
31. #define \_AFX\_ALL\_WARNINGS
32. #include <afxwin.h> // MFC 核心组件和标准组件
33. #include <afxext.h> // MFC 扩展
34. #include <afxdisp.h> // MFC 自动化类
35. #ifndef \_AFX\_NO\_OLE\_SUPPORT
36. #include <afxdtctl.h> // MFC 对 Internet Explorer 4 公共控件的支持
37. #endif
38. #ifndef \_AFX\_NO\_AFXCMN\_SUPPORT
39. #include <afxcmn.h> // MFC 对 Windows 公共控件的支持
40. #endif // \_AFX\_NO\_AFXCMN\_SUPPORT
41. #include <afxcontrolbars.h> // MFC 支持功能区和控制条
42. #ifdef \_UNICODE
43. #if defined \_M\_IX86
44. #pragma comment(linker,"/manifestdependency:\"type='win32' name='Microsoft.Windows.Common-Controls' version='6.0.0.0' processorArchitecture='x86' publicKeyToken='6595b64144ccf1df' language='\*'\"")
45. #elif defined \_M\_X64
46. #pragma comment(linker,"/manifestdependency:\"type='win32' name='Microsoft.Windows.Common-Controls' version='6.0.0.0' processorArchitecture='amd64' publicKeyToken='6595b64144ccf1df' language='\*'\"")
47. #else
48. #pragma comment(linker,"/manifestdependency:\"type='win32' name='Microsoft.Windows.Common-Controls' version='6.0.0.0' processorArchitecture='\*' publicKeyToken='6595b64144ccf1df' language='\*'\"")
49. #endif
50. #endif
51. // MyTagPaneDlg.h: 主对话框头文件
52. #pragma once
53. // CMyTagPaneDlg 对话框
54. class CMyTagPaneDlg : public CDialogEx
55. {
56. // 构造
57. public:
58. CMyTagPaneDlg(CWnd\* pParent = nullptr); // 标准构造函数
59. // 对话框数据
60. #ifdef AFX\_DESIGN\_TIME
61. enum { IDD = IDD\_MYTAGPANE\_DIALOG };
62. #endif
63. protected:
64. virtual void DoDataExchange(CDataExchange\* pDX); // DDX/DDV 支持
65. // 实现
66. protected:
67. HICON m\_hIcon;
68. // 生成的消息映射函数
69. virtual BOOL OnInitDialog();
70. afx\_msg void OnSysCommand(UINT nID, LPARAM lParam);
71. afx\_msg void OnPaint();
72. afx\_msg HCURSOR OnQueryDragIcon();
73. DECLARE\_MESSAGE\_MAP()
74. public:
75. void CalCurrentFlag();
76. void InitMap();
77. afx\_msg void OnRButtonDown(UINT nFlags, CPoint point);
78. afx\_msg void OnClearAll();
79. afx\_msg void OnStnClickedHelp();
80. afx\_msg void OnStnClickedMusic();
81. afx\_msg void OnStnClickedMusic2();
82. afx\_msg void OnAbout();
83. afx\_msg void OnTimer(UINT\_PTR nIDEvent);
84. afx\_msg BOOL OnSetCursor(CWnd\* pWnd, UINT nHitTest, UINT message);
85. afx\_msg void OnStnClickedqiehuan();
86. afx\_msg void OnStnClickedtansuo();
87. afx\_msg HBRUSH OnCtlColor(CDC\* pDC, CWnd\* pWnd, UINT nCtlColor);
88. };
89. //help对话框头文件
90. #pragma once
91. class help : public CDialogEx
92. {
93. DECLARE\_DYNAMIC(help)
94. public:
95. help(CWnd\* pParent = nullptr); // 标准构造函数
96. virtual ~help();
97. // 对话框数据
98. #ifdef AFX\_DESIGN\_TIME
99. enum { IDD = IDD\_help };
100. #endif
101. protected:
102. virtual void DoDataExchange(CDataExchange\* pDX); // DDX/DDV 支持
103. DECLARE\_MESSAGE\_MAP()
104. };
105. // MyTagPane.h: PROJECT\_NAME 应用程序的主头文件
106. #pragma once
107. #ifndef \_\_AFXWIN\_H\_\_
108. #error "在包含此文件之前包含 'pch.h' 以生成 PCH"
109. #endif
110. #include "resource.h" // 主符号
111. // CMyTagPaneApp:
112. // 有关此类的实现，请参阅 MyTagPane.cpp
113. class CMyTagPaneApp : public CWinApp
114. {
115. public:
116. CMyTagPaneApp();
117. // 重写
118. public:
119. virtual BOOL InitInstance();
120. // 实现
121. DECLARE\_MESSAGE\_MAP()
122. };
123. extern CMyTagPaneApp theApp;
124. // help.cpp: 实现文件
125. #include "pch.h"
126. #include "MyTagPane.h"
127. #include "help.h"
128. #include "afxdialogex.h"
129. IMPLEMENT\_DYNAMIC(help, CDialogEx)
130. help::help(CWnd\* pParent /\*=nullptr\*/)
131. : CDialogEx(IDD\_help, pParent)
132. {
133. }
134. help::~help()
135. {
136. }
137. void help::DoDataExchange(CDataExchange\* pDX)
138. {
139. CDialogEx::DoDataExchange(pDX);
140. }
141. BEGIN\_MESSAGE\_MAP(help, CDialogEx)
142. END\_MESSAGE\_MAP()
143. // MyTagPane.cpp: 定义应用程序的类行为。
144. #include "pch.h"
145. #include "framework.h"
146. #include "MyTagPane.h"
147. #include "MyTagPaneDlg.h"
148. #ifdef \_DEBUG
149. #define new DEBUG\_NEW
150. #endif
151. // CMyTagPaneApp
152. BEGIN\_MESSAGE\_MAP(CMyTagPaneApp, CWinApp)
153. ON\_COMMAND(ID\_HELP, &CWinApp::OnHelp)
154. END\_MESSAGE\_MAP()
155. // CMyTagPaneApp 构造
156. CMyTagPaneApp::CMyTagPaneApp()
157. {
158. // 支持重新启动管理器
159. m\_dwRestartManagerSupportFlags = AFX\_RESTART\_MANAGER\_SUPPORT\_RESTART;
160. // TODO: 在此处添加构造代码，
161. // 将所有重要的初始化放置在 InitInstance 中
162. }
163. // 唯一的 CMyTagPaneApp 对象
164. CMyTagPaneApp theApp;
165. // CMyTagPaneApp 初始化
166. BOOL CMyTagPaneApp::InitInstance()
167. {
168. // 如果一个运行在 Windows XP 上的应用程序清单指定要
169. // 使用 ComCtl32.dll 版本 6 或更高版本来启用可视化方式，
170. //则需要 InitCommonControlsEx()。 否则，将无法创建窗口。
171. INITCOMMONCONTROLSEX InitCtrls;
172. InitCtrls.dwSize = sizeof(InitCtrls);
173. // 将它设置为包括所有要在应用程序中使用的
174. // 公共控件类。
175. InitCtrls.dwICC = ICC\_WIN95\_CLASSES;
176. InitCommonControlsEx(&InitCtrls);
177. CWinApp::InitInstance();
178. AfxEnableControlContainer();
179. // 创建 shell 管理器，以防对话框包含
180. // 任何 shell 树视图控件或 shell 列表视图控件。
181. CShellManager \*pShellManager = new CShellManager;
182. // 激活“Windows Native”视觉管理器，以便在 MFC 控件中启用主题
183. CMFCVisualManager::SetDefaultManager(RUNTIME\_CLASS(CMFCVisualManagerWindows));
184. // 标准初始化
185. // 如果未使用这些功能并希望减小
186. // 最终可执行文件的大小，则应移除下列
187. // 不需要的特定初始化例程
188. // 更改用于存储设置的注册表项
189. // TODO: 应适当修改该字符串，
190. // 例如修改为公司或组织名
191. SetRegistryKey(\_T("应用程序向导生成的本地应用程序"));
192. CMyTagPaneDlg dlg;
193. m\_pMainWnd = &dlg;
194. INT\_PTR nResponse = dlg.DoModal();
195. if (nResponse == IDOK)
196. {
197. // TODO: 在此放置处理何时用
198. // “确定”来关闭对话框的代码
199. }
200. else if (nResponse == IDCANCEL)
201. {
202. // TODO: 在此放置处理何时用
203. // “取消”来关闭对话框的代码
204. }
205. else if (nResponse == -1)
206. {
207. TRACE(traceAppMsg, 0, "警告: 对话框创建失败，应用程序将意外终止。\n");
208. TRACE(traceAppMsg, 0, "警告: 如果您在对话框上使用 MFC 控件，则无法 #define \_AFX\_NO\_MFC\_CONTROLS\_IN\_DIALOGS。\n");
209. }
210. // 删除上面创建的 shell 管理器。
211. if (pShellManager != nullptr)
212. {
213. delete pShellManager;
214. }
215. #if !defined(\_AFXDLL) && !defined(\_AFX\_NO\_MFC\_CONTROLS\_IN\_DIALOGS)
216. ControlBarCleanUp();
217. #endif
218. // 由于对话框已关闭，所以将返回 FALSE 以便退出应用程序，
219. // 而不是启动应用程序的消息泵。
220. return FALSE;
221. }
222. //MyTagPaneDlg.cpp 主对话框程序
223. #include "pch.h"
224. #include "framework.h"
225. #include "MyTagPane.h"
226. #include "MyTagPaneDlg.h"
227. #include "afxdialogex.h"
228. #include"CGrid.h"
229. #include<mmsystem.h>
230. #include"help.h"
231. #ifdef \_DEBUG
232. #define new DEBUG\_NEW
233. #endif
234. #define MAX\_ROW 20
235. #define MAX\_COL 20
236. CGrid map[MAX\_ROW][MAX\_COL];
237. int row = 20;//行数
238. int col = 20;//列数
239. int nCurFlag = 0;//用力记录标记数
240. CRect ClientRect;
241. bool ModeFlag = true;//用来表示当前模式状态
242. int x=9, y=0;//之后演示模式中会用到
243. // 用于应用程序“关于”菜单项的 CAboutDlg 对话框
244. class CAboutDlg : public CDialogEx
245. {
246. public:
247. CAboutDlg();
248. // 对话框数据
249. #ifdef AFX\_DESIGN\_TIME
250. enum { IDD = IDD\_ABOUTBOX };
251. #endif
252. protected:
253. virtual void DoDataExchange(CDataExchange\* pDX); // DDX/DDV 支持
254. // 实现
255. protected:
256. DECLARE\_MESSAGE\_MAP()
257. };
258. CAboutDlg::CAboutDlg() : CDialogEx(IDD\_ABOUTBOX)
259. {
260. }
261. void CAboutDlg::DoDataExchange(CDataExchange\* pDX)
262. {
263. CDialogEx::DoDataExchange(pDX);
264. }
265. BEGIN\_MESSAGE\_MAP(CAboutDlg, CDialogEx)
266. END\_MESSAGE\_MAP()
267. // CMyTagPaneDlg 对话框
268. CMyTagPaneDlg::CMyTagPaneDlg(CWnd\* pParent /\*=nullptr\*/)
269. : CDialogEx(IDD\_MYTAGPANE\_DIALOG, pParent)
270. {
271. m\_hIcon = AfxGetApp()->LoadIcon(IDR\_MAINFRAME);
272. }
273. void CMyTagPaneDlg::DoDataExchange(CDataExchange\* pDX)
274. {
275. CDialogEx::DoDataExchange(pDX);
276. }
277. BEGIN\_MESSAGE\_MAP(CMyTagPaneDlg, CDialogEx)
278. ON\_WM\_SYSCOMMAND()
279. ON\_WM\_PAINT()
280. ON\_WM\_QUERYDRAGICON()
281. ON\_WM\_RBUTTONDOWN()
282. ON\_COMMAND(ID\_32771, &CMyTagPaneDlg::OnClearAll) //菜单的清空按钮
283. ON\_STN\_CLICKED(IDC\_Help, &CMyTagPaneDlg::OnStnClickedHelp) / /帮助框
284. ON\_STN\_CLICKED(IDC\_Music, &CMyTagPaneDlg::OnStnClickedMusic) //音乐开关
285. ON\_STN\_CLICKED(IDC\_Mute, &CMyTagPaneDlg::OnStnClickedMusic2) //静音开关
286. ON\_COMMAND(ID\_32772, &CMyTagPaneDlg::OnAbout) //关于按钮
287. ON\_WM\_MOUSEMOVE() //鼠标移动事件
288. ON\_WM\_TIMER() //计时器
289. ON\_WM\_SETCURSOR() //设置鼠标样式
290. ON\_STN\_CLICKED(IDC\_qiehuan, &CMyTagPaneDlg::OnStnClickedqiehuan) //切换模式按钮
291. ON\_WM\_LBUTTONDOWN() //左点击事件
292. ON\_WM\_CTLCOLOR()
293. END\_MESSAGE\_MAP()
294. // CMyTagPaneDlg 消息处理程序
295. BOOL CMyTagPaneDlg::OnInitDialog()
296. {
297. CDialogEx::OnInitDialog();
298. // 将“关于...”菜单项添加到系统菜单中。
299. // IDM\_ABOUTBOX 必须在系统命令范围内。
300. ASSERT((IDM\_ABOUTBOX & 0xFFF0) == IDM\_ABOUTBOX);
301. ASSERT(IDM\_ABOUTBOX < 0xF000);
302. CMenu\* pSysMenu = GetSystemMenu(FALSE);
303. if (pSysMenu != nullptr)
304. {
305. BOOL bNameValid;
306. CString strAboutMenu;
307. bNameValid = strAboutMenu.LoadString(IDS\_ABOUTBOX);
308. ASSERT(bNameValid);
309. if (!strAboutMenu.IsEmpty())
310. {
311. pSysMenu->AppendMenu(MF\_SEPARATOR);
312. pSysMenu->AppendMenu(MF\_STRING, IDM\_ABOUTBOX, strAboutMenu);
313. }
314. }
315. // 设置此对话框的图标。 当应用程序主窗口不是对话框时，框架将自动
316. // 执行此操作
317. SetIcon(m\_hIcon, TRUE); // 设置大图标
318. SetIcon(m\_hIcon, FALSE); // 设置小图标
319. // TODO: 在此添加额外的初始化代码
320. PlaySound((LPCTSTR)IDR\_WAVE1, AfxGetInstanceHandle(), SND\_RESOURCE | SND\_ASYNC);//默认初始状态是播放音乐
321. GetDlgItem(IDC\_Music)->ShowWindow(SW\_HIDE);//当点击到音乐开关控件时，切换不同的图片
322. GetDlgItem(IDC\_Mute)->ShowWindow(SW\_SHOW);
323. SetWindowText(L"标记方格II【刘斌琰】"); //设置窗口标题
324. CClientDC dc(this);
325. CRect rect;
326. GetClientRect(&rect);
327. CDC dcMem;
328. dcMem.CreateCompatibleDC(&dc);
329. return TRUE; // 除非将焦点设置到控件，否则返回 TRUE
330. }
331. void CMyTagPaneDlg::OnSysCommand(UINT nID, LPARAM lParam)
332. {
333. if ((nID & 0xFFF0) == IDM\_ABOUTBOX)
334. {
335. CAboutDlg dlgAbout;
336. dlgAbout.DoModal();
337. }
338. else
339. {
340. CDialogEx::OnSysCommand(nID, lParam);
341. }
342. }
343. void CMyTagPaneDlg::OnPaint()
344. {
345. CPaintDC dc(this);
346. if (IsIconic())
347. {
348. CPaintDC dc(this); // 用于绘制的设备上下文
349. SendMessage(WM\_ICONERASEBKGND, reinterpret\_cast<WPARAM>(dc.GetSafeHdc()), 0);
350. // 使图标在工作区矩形中居中
351. int cxIcon = GetSystemMetrics(SM\_CXICON);
352. int cyIcon = GetSystemMetrics(SM\_CYICON);
353. CRect rect;
354. GetClientRect(&rect);
355. int x = (rect.Width() - cxIcon + 1) / 2;
356. int y = (rect.Height() - cyIcon + 1) / 2;
357. // 绘制图标
358. dc.DrawIcon(x, y, m\_hIcon);
359. }
360. if (ModeFlag) //对当前的模式状态进行判断，如果是真则是探索模式
361. {
362. GetDlgItem(IDC\_tansuoD)->ShowWindow(SW\_SHOW);
363. GetDlgItem(IDC\_tansuo)->ShowWindow(SW\_HIDE);
364. GetDlgItem(IDC\_yanshi)->ShowWindow(SW\_SHOW);
365. GetDlgItem(IDC\_yanshiD)->ShowWindow(SW\_HIDE);
366. KillTimer(1);
367. }
368. if (!ModeFlag) //如果是假则是演示模式
369. {
370. GetDlgItem(IDC\_tansuoD)->ShowWindow(SW\_HIDE);
371. GetDlgItem(IDC\_tansuo)->ShowWindow(SW\_SHOW);
372. GetDlgItem(IDC\_yanshi)->ShowWindow(SW\_HIDE);
373. GetDlgItem(IDC\_yanshiD)->ShowWindow(SW\_SHOW);
374. SetTimer(1, 1000, NULL);
375. }
376. CDC memDC;
377. memDC.CreateCompatibleDC(&dc);
378. GetClientRect(&ClientRect);//获得用户区域
379. CalCurrentFlag();
380. CFont font;
381. CString s;
382. s.Format(L"标记数：%d", nCurFlag);
383. SetDlgItemText( IDC\_FlagNum, s);
384. /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/
385. CBitmap bitEmpty, bitTag, bitBoom, bitMusic, bitMute, bitBkpic;
386. //MemDC.CreateCompatibleDC(0);
387. bitEmpty.LoadBitmapW(IDB\_BITMAP1);
388. bitTag.LoadBitmapW(IDB\_BITMAP2);
389. bitBoom.LoadBitmapW(IDB\_BITMAP4);
390. CClientDC Clientdc(this);
391. /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*绘制地图 双重循环加载每个小方格的位图/
392. for (int i = 0; i < row; i++) {
393. for (int j = 0; j < col; j++) {
394. if (map[i][j].bFlaged == false) {
395. SelectObject(memDC, bitEmpty);
396. BitBlt(Clientdc.m\_hDC, j \* 32 + ClientRect.Width() / 2 - 32 \* col / 2, i \* 32 + ClientRect.Height() / 2 - row \* 32 / 2, 32, 32, memDC, 0, 0, SRCCOPY);
397. }
398. else
399. {
400. SelectObject(memDC, bitTag);
401. BitBlt(Clientdc.m\_hDC, j \* 32 + ClientRect.Width() / 2 - 32 \* col / 2, i \* 32 + ClientRect.Height() / 2 - row \* 32 / 2, 32, 32, memDC, 0, 0, SRCCOPY);
402. int n = 0;
403. if (map[i - 1][j].bFlaged)
404. n++;
405. if (map[i + 1][j].bFlaged)
406. n++;
407. if (map[i][j - 1].bFlaged)
408. n++;
409. if (map[i][j + 1].bFlaged)
410. n++;
411. if (n == 2 || n == 4 || n == 0)
412. {
413. SelectObject(memDC, bitBoom);
414. BitBlt(Clientdc.m\_hDC, j \* 32 + ClientRect.Width() / 2 - 32 \* col / 2, i \* 32 + ClientRect.Height() / 2 - row \* 32 / 2, 32, 32, memDC, 0, 0, SRCCOPY);
415. }
416. }
417. }
418. }
419. memDC.DeleteDC();
420. }
421. //当用户拖动最小化窗口时系统调用此函数取得光标
422. //显示。
423. HCURSOR CMyTagPaneDlg::OnQueryDragIcon()
424. {
425. return static\_cast<HCURSOR>(m\_hIcon);
426. }
427. void CMyTagPaneDlg::CalCurrentFlag()//对所有格子遍历，记录一共有多少格子的属性为被标记
428. {
429. // TODO: 在此处添加实现代码.
430. int n = 0;
431. for (int i = 0; i < row; i++) {
432. for (int j = 0; j < col; j++) {
433. if (map[i][j].bFlaged) {
434. n++;
435. }
436. }
437. }
438. nCurFlag = n;
439. }
440. void CMyTagPaneDlg::InitMap()//初始化地图，将所有格子的属性全部重置为空标记，全部清空时会用到
441. {
442. // TODO: 在此处添加实现代码.
443. ASSERT(row <= MAX\_ROW && col <= MAX\_COL);
444. for (int i = 0; i < row; i++) {
445. for (int j = 0; j < col; j++) {
446. map[i][j].bFlaged = false;
447. }
448. }
449. Invalidate();
450. }
451. void CMyTagPaneDlg::OnRButtonDown(UINT nFlags, CPoint point)//右键点击事件，给格子添加被标记属性
452. {
453. // TODO: 在此添加消息处理程序代码和/或调用默认值
454. double map\_x, map\_y;
455. map\_x = (point.x - ClientRect.Width() / 2.0 + 32.0 \* col / 2.0);
456. map\_y = (point.y - ClientRect.Height() / 2.0 + 32.0 \* row / 2.0);
457. if (map\_x < 0.0 || map\_y < 0.0 || map\_x >= col \* 32.0 || map\_y >= row \* 32.0) {//超出界限
458. goto defProc;
459. }
460. int j = (point.x - ClientRect.Width() / 2.0 + 32.0 \* col / 2.0) / 32.0;//取整，让鼠标的位置转化为每个格子的范围
461. int i = (point.y - ClientRect.Height() / 2.0 + 32.0 \* row / 2.0) / 32.0;
462. ASSERT(i >= 0 && i < row&& j >= 0 && j < col);//断言，只有满足条件才运行下面的代码
463. map[i][j].bFlaged = !map[i][j].bFlaged;//作标记或者取消标记
464. Invalidate();//没有
465. defProc:
466. CDialogEx::OnRButtonDown(nFlags, point);
467. }
468. void CMyTagPaneDlg::OnClearAll() //清空按钮点击函数
469. {
470. // TODO: 在此添加命令处理程序代码
471. InitMap();
472. }
473. void CMyTagPaneDlg::OnStnClickedHelp() //帮助按钮对应函数
474. {
475. // TODO: 在此添加控件通知处理程序代码
476. PlaySound((LPCTSTR)IDR\_WAVE2, AfxGetInstanceHandle(), SND\_RESOURCE | SND\_ASYNC);
477. help helpbox;
478. helpbox.DoModal(); //弹出另外一个帮助子对话框
479. }
480. void CMyTagPaneDlg::OnStnClickedMusic() //音乐开关点击函数
481. {
482. // TODO: 在此添加控件通知处理程序代码
483. PlaySound((LPCTSTR)IDR\_WAVE1, AfxGetInstanceHandle(), SND\_RESOURCE | SND\_ASYNC);
484. GetDlgItem(IDC\_Music)->ShowWindow(SW\_HIDE);
485. GetDlgItem(IDC\_Mute)->ShowWindow(SW\_SHOW);
486. }
487. void CMyTagPaneDlg::OnStnClickedMusic2()//点击静音按钮处理函数
488. {
489. // TODO: 在此添加控件通知处理程序代码
490. PlaySound(NULL, NULL, SND\_PURGE);
491. GetDlgItem(IDC\_Mute)->ShowWindow(SW\_HIDE);
492. GetDlgItem(IDC\_Music)->ShowWindow(SW\_SHOW);
493. }
494. void CMyTagPaneDlg::OnAbout()
495. {
496. // TODO: 在此添加命令处理程序代码
497. CAboutDlg Aboutdlg;
498. Aboutdlg.DoModal();
499. }
500. void CMyTagPaneDlg::OnTimer(UINT\_PTR nIDEvent)
501. {
502. // TODO: 在此添加消息处理程序代码和/或调用默认值
503. map[x][y].bFlaged = true;
504. y++;
505. Invalidate();
506. CDialogEx::OnTimer(nIDEvent);
507. }
508. BOOL CMyTagPaneDlg::OnSetCursor(CWnd\* pWnd, UINT nHitTest, UINT message)
509. {
510. if (pWnd == GetDlgItem(IDC\_Music)||pWnd==GetDlgItem(IDC\_Mute)|| pWnd == GetDlgItem(IDC\_Help))
511. {
512. SetCursor(LoadCursor(NULL, IDC\_HAND));
513. return true;
514. }
515. if (pWnd == GetDlgItem(IDC\_qiehuan))
516. {
517. SetCursor(LoadCursor(NULL, IDC\_HAND));
518. return true;
519. }
520. return CDialogEx::OnSetCursor(pWnd, nHitTest, message);
521. }
522. void CMyTagPaneDlg::OnStnClickedqiehuan()
523. {
524. // TODO: 在此添加控件通知处理程序代码
525. ModeFlag = !ModeFlag;
526. Invalidate();
527. }
528. HBRUSH CMyTagPaneDlg::OnCtlColor(CDC\* pDC, CWnd\* pWnd, UINT nCtlColor)
529. //给标记数显示框设置字体颜色和背景颜色
530. {
531. HBRUSH hbr = CDialogEx::OnCtlColor(pDC, pWnd, nCtlColor);
532. if (pWnd->GetDlgCtrlID() == IDC\_FlagNum)// IDC\_Display为所选文本框ID
534. {
535. pDC->SetTextColor(RGB(131, 175, 155));//设置字体颜色
536. return (HBRUSH)::GetStockObject(WHITE\_BRUSH);
537. }
538. return hbr;
539. }