**《Windows应用程序原理》实验报告**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **姓名** | | 姚凡 | | | **年级** | 2022 |
| **学号** | | 20221879 | | | **专业、班级** | 信息安全02班 |
| **实验名称** | 实验三 编写Windows窗口应用程序实现键盘鼠标消息响应 | | | | | |
| **实验时间** | 2024.4.23 | | **实验地点** | DS3304 | | |
| **实验性质** | | | **□验证性 □设计性 ☑综合性** | | | |
| 1. 实验目的 2. 熟悉Windows窗口应用程序的开发环境和基本框架。 3. 学习如何响应键盘和鼠标消息。 4. 掌握如何在窗口应用程序中绘制图形界面，包括分割客户区域、绘制表格等。 5. 理解如何生成随机数，并将其应用于简易纸牌游戏中。 6. 通过简易纸牌游戏设计和实现，加深对Windows程序设计编程的理解和应用能力。 | | | | | | |
| 二、实验项目内容  1.编程实现简易纸牌游戏。  2.将客户区分割为上下两部分，上半部分根据高度和宽度画出2\*5的表格，下半部分划10个子窗口。  3.用户按回车或者空格键后重新让每个子窗口产生一个0-9的随机数，但不显示，每个窗口内的数字各不相同，并清空上部表格中的数字。  4.在子窗口上单击鼠标左键后在该窗口上显示自己的随机数，并将该数据依此填入上部分的表格中，以备后面检查猜测成果。  5.游戏的目标是尽量按由小到大的顺序猜出随机数。 | | | | | | |
| 1. 实验过程或算法（源程序）   #include <windows.h>  #include <stdio.h>  #include <stdlib.h>  #include <time.h>  #define WINDOW\_WIDTH 800  #define WINDOW\_HEIGHT 600  #define GRID\_ROWS 2  #define GRID\_COLS 5  #define SUBWINDOW\_COUNT 10  #define SUBWINDOW\_WIDTH 50  #define SUBWINDOW\_HEIGHT 50  LRESULT CALLBACK WndProc(HWND, UINT, WPARAM, LPARAM);  void DrawGrid(HDC);  void GenerateRandomNumbers(void);  void DisplayRandomNumbers(HDC);  void HandleSubWindowClick(int, int);  int randomNumbers[SUBWINDOW\_COUNT];  int guessedNumbers[GRID\_ROWS][GRID\_COLS];  int currentGuessIndex = 0;  int WINAPI WinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance, LPSTR lpCmdLine, int nCmdShow) {  WNDCLASSEX wcex;  HWND hWnd;  MSG msg;  wcex.cbSize = sizeof(WNDCLASSEX);  wcex.style = CS\_HREDRAW | CS\_VREDRAW;  wcex.lpfnWndProc = WndProc;  wcex.cbClsExtra = 0;  wcex.cbWndExtra = 0;  wcex.hInstance = hInstance;  wcex.hIcon = LoadIcon(NULL, IDI\_APPLICATION);  wcex.hCursor = LoadCursor(NULL, IDC\_ARROW);  wcex.hbrBackground = (HBRUSH)(COLOR\_WINDOW + 1);  wcex.lpszMenuName = NULL;  wcex.lpszClassName = L"SimpleCardGame";  wcex.hIconSm = LoadIcon(NULL, IDI\_APPLICATION);  SetConsoleOutputCP(CP\_UTF8);  SetConsoleCP(CP\_UTF8);  if (!RegisterClassEx(&wcex)) {  MessageBox(NULL, L"Window Registration Failed!", L"Error!", MB\_ICONEXCLAMATION | MB\_OK);  return 0;  }  hWnd = CreateWindowEx(  WS\_EX\_CLIENTEDGE,  L"SimpleCardGame",  L"Simple Card Game",  WS\_OVERLAPPEDWINDOW,  CW\_USEDEFAULT, CW\_USEDEFAULT,  WINDOW\_WIDTH, WINDOW\_HEIGHT,  NULL, NULL, hInstance, NULL);  if (hWnd == NULL) {  MessageBox(NULL, L"Window Creation Failed!", L"Error!", MB\_ICONEXCLAMATION | MB\_OK);  return 0;  }  ShowWindow(hWnd, nCmdShow);  UpdateWindow(hWnd);  while (GetMessage(&msg, NULL, 0, 0)) {  TranslateMessage(&msg);  DispatchMessage(&msg);  }  return (int)msg.wParam;  }  LRESULT CALLBACK WndProc(HWND hWnd, UINT message, WPARAM wParam, LPARAM lParam) {  HDC hdc;  PAINTSTRUCT ps;  switch (message) {  case WM\_CREATE:  srand((unsigned int)time(NULL));  GenerateRandomNumbers();  break;  case WM\_PAINT:  hdc = BeginPaint(hWnd, &ps);  DrawGrid(hdc);  DisplayRandomNumbers(hdc);  EndPaint(hWnd, &ps);  break;  case WM\_KEYDOWN:  if (wParam == VK\_RETURN || wParam == VK\_SPACE) {  GenerateRandomNumbers();  InvalidateRect(hWnd, NULL, TRUE);  }  break;  case WM\_LBUTTONDOWN:  HandleSubWindowClick(LOWORD(lParam), HIWORD(lParam));  InvalidateRect(hWnd, NULL, TRUE);  break;  case WM\_DESTROY:  PostQuitMessage(0);  break;  default:  return DefWindowProc(hWnd, message, wParam, lParam);  }  return 0;  }  void DrawGrid(HDC hdc) {  int x, y, i, j;  RECT rect;  // 绘制上半部分  for (i = 0; i < GRID\_ROWS; i++) {  for (j = 0; j < GRID\_COLS; j++) {  x = j \* (WINDOW\_WIDTH / GRID\_COLS);  y = i \* (WINDOW\_HEIGHT / 2 / GRID\_ROWS);  rect.left = x;  rect.top = y;  rect.right = x + (WINDOW\_WIDTH / GRID\_COLS);  rect.bottom = y + (WINDOW\_HEIGHT / 2 / GRID\_ROWS);  DrawEdge(hdc, &rect, EDGE\_RAISED, BF\_RECT);  if (guessedNumbers[i][j] != -1) {  wchar\_t buffer[10];  swprintf\_s(buffer, 10, L"%d", guessedNumbers[i][j]);  DrawText(hdc, buffer, -1, &rect, DT\_SINGLELINE | DT\_CENTER | DT\_VCENTER);  }  }  }  // 绘制下半部分  for (i = 0; i < SUBWINDOW\_COUNT; i++) {  x = (i % GRID\_COLS) \* (WINDOW\_WIDTH / GRID\_COLS);  y = WINDOW\_HEIGHT / 2 + (i / GRID\_COLS) \* (WINDOW\_HEIGHT / 2 / GRID\_ROWS);  rect.left = x;  rect.top = y;  rect.right = x + SUBWINDOW\_WIDTH;  rect.bottom = y + SUBWINDOW\_HEIGHT;  DrawEdge(hdc, &rect, EDGE\_RAISED, BF\_RECT);  }  }  void GenerateRandomNumbers(void) {  int i, j, temp;  int numbers[SUBWINDOW\_COUNT];  // 初始化数组  for (i = 0; i < SUBWINDOW\_COUNT; i++) {  numbers[i] = i;  }  // 洗牌  for (i = SUBWINDOW\_COUNT - 1; i > 0; i--) {  j = rand() % (i + 1);  temp = numbers[i];  numbers[i] = numbers[j];  numbers[j] = temp;  }  // 记牌  for (i = 0; i < SUBWINDOW\_COUNT; i++) {  randomNumbers[i] = numbers[i];  guessedNumbers[i / GRID\_COLS][i % GRID\_COLS] = -1;  }  currentGuessIndex = 0;  }  void DisplayRandomNumbers(HDC hdc) {  int x, y, i, j;  RECT rect;  for (i = 0; i < SUBWINDOW\_COUNT; i++) {  x = (i % GRID\_COLS) \* (WINDOW\_WIDTH / GRID\_COLS);  y = WINDOW\_HEIGHT / 2 + (i / GRID\_COLS) \* (WINDOW\_HEIGHT / 2 / GRID\_ROWS);  rect.left = x;  rect.top = y;  rect.right = x + SUBWINDOW\_WIDTH;  rect.bottom = y + SUBWINDOW\_HEIGHT;  if (randomNumbers[i] != -1) {  // 点击后才显示数字  continue;  }  if (guessedNumbers[i / GRID\_COLS][i % GRID\_COLS] != -1) {  wchar\_t buffer[10];  swprintf\_s(buffer, 10, L"%d", guessedNumbers[i / GRID\_COLS][i % GRID\_COLS]);  DrawText(hdc, buffer, -1, &rect, DT\_SINGLELINE | DT\_CENTER | DT\_VCENTER);  }  }  }  void HandleSubWindowClick(int x, int y) {  int subWindowIndex = (y - (WINDOW\_HEIGHT / 2)) / (WINDOW\_HEIGHT / 2 / GRID\_ROWS) \* GRID\_COLS + x / (WINDOW\_WIDTH / GRID\_COLS);  if (subWindowIndex >= 0 && subWindowIndex < SUBWINDOW\_COUNT && randomNumbers[subWindowIndex] != -1) {  wchar\_t buffer[10];  swprintf\_s(buffer, 10, L"%d", randomNumbers[subWindowIndex]);  guessedNumbers[currentGuessIndex / GRID\_COLS][currentGuessIndex % GRID\_COLS] = randomNumbers[subWindowIndex];  randomNumbers[subWindowIndex] = -1;  currentGuessIndex++;  }  } | | | | | | |
| 1. 实验结果及分析和（或）源程序调试过程   1.将客户区分割为上下两部分，上半部分根据高度和宽度画出2\*5的表格，下半部分划10个子窗口，10个子窗口不显示生成的随机数。    图1 弹出窗口如图所示  2.在子窗口上单击鼠标左键后在该窗口上显示自己的随机数，并将该数据依此填入上部分的表格中。    图2 点击子窗口后上方依次填入数据    图3 继续点击子窗口  3.用户按回车或者空格键后重新让每个子窗口产生一个0-9的随机数，但不显示，每个窗口内的数字各不相同，并清空上部表格中的数字。    图4 按下空格或回车键，表格清零，子窗口重新生成0-9的随机数  4.将数据依此填入上部分的表格中，以备后面检查猜测成果。游戏的目标是尽量按由小到大的顺序猜出随机数。    图5 round 1    图6 round 2    图7 从小到大猜测失败    图8 表格填满测试图 | | | | | | |