ใบงานที่ 5 Input

# **วัตถุประสงค์**

1. สร้างโปรแกรมรับอินพุตแบบต่างๆ

2. ใช้เครื่องมือต่างๆ ในการสร้าง Application

# **เนื้อหาเบื้องต้น**

https://github.com/Desktop-Programming-Lab-2559/LAB-02/blob/master/LabIntro.md

# **ลำดับการทดลอง**

1. เรียกโปรแกรม Microsoft Visual Studio

2. สร้าง Project ใหม่ โดยเลือกเมนู File >> New >> Project… (Ctrl+Shift+N)

· ช่อง Templates: ให้เลือก Visual C++ และเลือกชนิด project เป็น Empty Project

· ช่อง Name: ให้ใส่ชื่อของ Project เป็น LAB4

· ช่อง Location: ให้เลือกตำแหน่งที่จะสร้าง Project (D:\Student\_Code\LAB\_01\_05)

· ส่วนที่เหลือ ให้คงไว้ตามที่ปรากฏ กด OK

3. เพิ่ม source code ให้กับ project โดยการเลือกเมนู PROJECT >> Add New Item…

· ตั้งชื่อไฟล์เป็น main.cpp

· กด Add เพื่อเพิ่มไฟล์

4. พิมพ์โปรแกรมดังต่อไปนี้ลงในไฟล์ main.cpp

|  |
| --- |
| #include <windows.h>  #include <tchar.h>  // Global Variables:  HINSTANCE hInst; // current instance  TCHAR szTitle[]="20170920\_L4"; // The title bar text  TCHAR szWindowClass[]="WinApp"; // the class name  // Forward declarations of functions included in this code module:  ATOM MyRegisterClass(HINSTANCE hInstance);  BOOL InitInstance(HINSTANCE, LPSTR, int);  LRESULT CALLBACK WndProc(HWND, UINT, WPARAM, LPARAM);  int APIENTRY WinMain(HINSTANCE hInstance,  HINSTANCE hPrevInstance,  LPSTR lpCmdLine,  int nCmdShow)  {  // TODO: Place code here.  MSG msg;  // Register Class  MyRegisterClass(hInstance);  // Perform application initialization:  if (!InitInstance (hInstance, lpCmdLine, nCmdShow))  {  return FALSE;  }  // Main message loop:  while (GetMessage(&msg, NULL, 0, 0))  {  TranslateMessage(&msg);  DispatchMessage(&msg);  }  return msg.wParam;  }  //  // FUNCTION: MyRegisterClass()  //  // PURPOSE: Registers the window class.  //  // COMMENTS:  //  // This function and its usage is only necessary if you want this code  // to be compatible with Win32 systems prior to the 'RegisterClassEx'  // function that was added to Windows 95.  //  ATOM MyRegisterClass(HINSTANCE hInstance)  {  WNDCLASSEX wcex;  wcex.cbSize = sizeof(WNDCLASSEX);  wcex.style = CS\_HREDRAW | CS\_VREDRAW;  wcex.lpfnWndProc = (WNDPROC)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 = szWindowClass;  wcex.hIconSm = LoadIcon (NULL, IDI\_APPLICATION);  return RegisterClassEx(&wcex);  }  //  // FUNCTION: InitInstance(HANDLE, int)  //  // PURPOSE: Saves instance handle and creates main window  //  // COMMENTS:  //  // In this function, we save the instance handle in a global variable and  // create and display the main program window.  //  BOOL InitInstance(HINSTANCE hInstance, LPSTR lpCmdLine, int nCmdShow)  {  HWND hWnd;  hInst = hInstance; // Store instance handle in our global variable  hWnd = CreateWindow(szWindowClass, lpCmdLine, WS\_OVERLAPPEDWINDOW,  CW\_USEDEFAULT, 0, CW\_USEDEFAULT, 0, NULL, NULL, hInstance, NULL);  if (!hWnd)  {p  return FALSE;  }  ShowWindow(hWnd, nCmdShow);  UpdateWindow(hWnd);  return TRUE;  }  //  // FUNCTION: WndProc(HWND, unsigned, WORD, LONG)  //  // PURPOSE: Processes messages for the main window.  //  // WM\_COMMAND - process the application menu  // WM\_PAINT - Paint the main window  // WM\_DESTROY - post a quit message and return  //  //  LRESULT CALLBACK WndProc(HWND hWnd, UINT message, WPARAM wParam, LPARAM lParam)  {  PAINTSTRUCT ps;  HDC hdc;  RECT rt;  char szHello[]="Hello, C-Free!";    switch (message)  {  case WM\_PAINT:  hdc = BeginPaint(hWnd, &ps);  // TODO: Add any drawing code here...    GetClientRect(hWnd, &rt);  DrawText(hdc, szHello, strlen(szHello), &rt, DT\_CENTER);  EndPaint(hWnd, &ps);  break;  case WM\_CLOSE:  DestroyWindow(hWnd);  break;  case WM\_DESTROY:  PostQuitMessage(0);  break;  default:  return DefWindowProc(hWnd, message, wParam, lParam);  }  return 0;  } |

5. ทำการทดสอบโปรแกรม และบันทึกผล

|  |
| --- |
| Ans. ก่อนทดสอบโปรแกรม ถ้าหากเลือกกดแถบเมนู project แล้วเลือกที่ settings สามารถกำหนดค่า parameters ได้ โดยหากกำหนดใส่ค่า parameters ค่านั้นจะปรากฎที่แถบ title bar ของหน้าต่าง window ของโปรแกรมที่เขียน |

6. แก้ไขโปรแกรมตามที่กำหนด

|  |
| --- |
| case WM\_CLOSE:  MessageBeep(10);  DestroyWindow(hWnd);  break; |

7. ทำการทดสอบโปรแกรม และบันทึกผล

|  |
| --- |
| Ans. หลังจากทดสอบโปรแกรมแล้วหน้าต่าง window เด้งขึ้น ถ้ากดปิดจะมีเสียงเกิดขึ้นหลังจากกดปิดไปแล้ว |

8. New Project

9. พิมพ์โปรแกรมตามที่กำหนด

|  |
| --- |
| #include <windows.h>  #include <tchar.h>  #include "DrawBall.h"  // Global Variables:  HINSTANCE hInst; // current instance  TCHAR szTitle[]="20170920\_L6"; // The title bar text  TCHAR szWindowClass[]="WinApp"; // the class name  BOOL \_bDrawOn = FALSE;  int \_nXSize,\_nYSize;  HWND hWnd;  // Foward declarations of functions included in this code module:  ATOM MyRegisterClass(HINSTANCE hInstance);  BOOL InitInstance(HINSTANCE, int);  LRESULT CALLBACK WndProc(HWND, UINT, WPARAM, LPARAM);  void DrawBall(HWND, int, int);  int APIENTRY WinMain(HINSTANCE hInstance,  HINSTANCE hPrevInstance,  LPSTR lpCmdLine,  int nCmdShow)  {  // TODO: Place code here.  MSG msg;  // Register Class  MyRegisterClass(hInstance);  // Perform application initialization:  if (!InitInstance (hInstance, nCmdShow))  {  return FALSE;  }  while(TRUE)  {  if(PeekMessage(&msg, NULL, 0, 0, PM\_REMOVE))  {  if(msg.message==WM\_QUIT)  {  return msg.wParam;  }else  {  TranslateMessage(&msg);  DispatchMessage(&msg);  }  }else  {  if(\_bDrawOn)  {  DrawBall(hWnd,\_nXSize, \_nYSize);  }  }    }  }  //  // FUNCTION: MyRegisterClass()  //  // PURPOSE: Registers the window class.  //  // COMMENTS:  //  // This function and its usage is only necessary if you want this code  // to be compatible with Win32 systems prior to the 'RegisterClassEx'  // function that was added to Windows 95.  //  ATOM MyRegisterClass(HINSTANCE hInstance)  {  WNDCLASSEX wcex;  wcex.cbSize = sizeof(WNDCLASSEX);  wcex.style = CS\_HREDRAW | CS\_VREDRAW;  wcex.lpfnWndProc = (WNDPROC)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 = MAKEINTRESOURCE(MyMenu);  wcex.lpszClassName = szWindowClass;  wcex.hIconSm = LoadIcon (NULL, IDI\_APPLICATION);  return RegisterClassEx(&wcex);  }  //  // FUNCTION: InitInstance(HANDLE, int)  //  // PURPOSE: Saves instance handle and creates main window  //  // COMMENTS:  //  // In this function, we save the instance handle in a global variable and  // create and display the main program window.  //  BOOL InitInstance(HINSTANCE hInstance, int nCmdShow)  {  hInst = hInstance; // Store instance handle in our global variable  hWnd = CreateWindow(szWindowClass, szTitle, WS\_OVERLAPPEDWINDOW,  CW\_USEDEFAULT, 0, CW\_USEDEFAULT, 0, NULL, NULL, hInstance, NULL);  if (!hWnd)  {  return FALSE;  }  ShowWindow(hWnd, nCmdShow);  UpdateWindow(hWnd);  return TRUE;  }  //  // FUNCTION: WndProc(HWND, unsigned, WORD, LONG)  //  // PURPOSE: Processes messages for the main window.  //  // WM\_COMMAND - process the application menu  // WM\_PAINT - Paint the main window  // WM\_DESTROY - post a quit message and return  //  //  LRESULT CALLBACK WndProc(HWND hWnd, UINT message, WPARAM wParam, LPARAM lParam)  {  PAINTSTRUCT ps;  HDC hdc;  RECT rt;  char szHello[]="Hello, C-Free!";    switch (message)  {  case WM\_CREATE:  \_bDrawOn=FALSE;  break;  case WM\_SIZE:  \_nXSize=LOWORD(lParam);  \_nYSize=HIWORD(lParam);  break;  case WM\_COMMAND:  switch(wParam)  {  case IDM\_SHOW:  if(\_bDrawOn)  {  \_bDrawOn=FALSE;  }else  {  \_bDrawOn=TRUE;  }  break;  case IDM\_QUIT:  DestroyWindow(hWnd);  break;  }  break;  case WM\_CLOSE:  DestroyWindow(hWnd);  break;  case WM\_DESTROY:  PostQuitMessage(0);  break;  default:  return DefWindowProc(hWnd, message, wParam, lParam);  }  return 0;  } |

10. พิมพ์โปรแกรมตามที่กำหนด บันทึกผลในชื่อ DrawBall.cpp

|  |
| --- |
| #include<windows.h>  #include"DrawBall.h"  void DrawBall(HWND hWnd, int nXSize, int nYSize)  {  HDC hDC;  HPEN hPen, hOldPen;  HBRUSH hBrush, hOldBrush;  static int nX=2\*BALLRAD, nY=2\*BALLRAD, nVelX=VELOCITY, nVelY=VELOCITY;  nXSize=GetSystemMetrics(SM\_CXSCREEN);  nYSize=GetSystemMetrics(SM\_CYSCREEN);  nX=nX+nVelX;  nY=nY+nVelY;    if(nY>nYSize) nY= 2\*BALLRAD;  if(nX>nXSize) nX= 2\*BALLRAD;  if((nY<MINRAD)||(nYSize-nY<MINRAD))  {  nVelY =-1\*nVelY;  }else  {  if((nX<MINRAD)||(nXSize-nX<MINRAD))  {  nVelX =-1\*nVelX;  }  }    hDC=GetDC(hWnd);  hPen=CreatePen(PS\_SOLID, VELOCITY+VELOCITY/2, RGB(255,255,255));  hBrush=CreateSolidBrush(RGB(255, 0, 0));  hOldPen=(HPEN)SelectObject(hDC,hPen);  hOldBrush=(HBRUSH)SelectObject(hDC,hBrush);  Ellipse(hDC, nX-BALLRAD, nY-BALLRAD, nX+BALLRAD,nY+BALLRAD);    SelectObject(hDC,hOldPen);  DeleteObject(hPen);  hPen=CreatePen(PS\_SOLID,3,RGB(255, 255, 255));  hOldPen=(HPEN)SelectObject(hDC,hPen);    Rectangle(hDC, nX+BALLRAD/3, nY-BALLRAD/3, nX+2+BALLRAD/3, nY-2-BALLRAD/3);  SelectObject(hDC,hOldPen);  DeleteObject(hPen);  SelectObject(hDC,hOldBrush);  DeleteObject(hBrush);  ReleaseDC(hWnd, hDC);  } |

11. เพิ่มไฟล์ชื่อ DrawBall.cpp ไปยัง Project ในข้อที่ 8

12. พิมพ์โปรแกรมตามที่กำหนด บันทึกในชื่อ DrawBall.rc

|  |
| --- |
| #include "DrawBall.h"  MyIcon ICON DrawBall.ico  MyMenu MENU  BEGIN  MENUITEM "&Ball", IDM\_SHOW  MENUITEM "&Quit", IDM\_QUIT  END |

13. พิมพ์โปรแกรมตามที่กำหนด บันทึกในชื่อ DrawBall.h

|  |
| --- |
| #define MyMenu 101  #define IDM\_SHOW 1  #define IDM\_QUIT 10  #define VELOCITY 5  #define BALLRAD 20  #define MINRAD 15 |

14.ทำการทดสอบโปรแกรม และบันทึกผล

|  |
| --- |
| Ans. เป็นโปรแกรมสร้างลูกบอลให้วิ่งในหน้าต่าง window โดยมีคำสั่งให้กดตรงแถบเมนูด้านบน ถ้าหากกดที่ Ball ก็จะมีลูกบอลสีแดงวิ่งไปมาบนหน้าต่าง window แต่ถ้าหากกด Quit ก็จะเป็นการปิดหน้าต่าง window |

15. New Project

16. แก้ไขโปรแกรมตามที่กำหนด

|  |
| --- |
| LRESULT CALLBACK WndProc(HWND hWnd, UINT message, WPARAM wParam, LPARAM lParam)  {  PAINTSTRUCT ps;  HDC hdc;  RECT rt;  char szHello[]="Hello, C-Free!";  static BOOL bFirstTime=TRUE;  static HWND hStaticText, hStaticIcon, hStaticRect;  switch (message)  {  case WM\_CREATE:  if(bFirstTime)  {  bFirstTime=FALSE;  hStaticText=CreateWindow("Static","This is a static text(LABEL)", WS\_CHILD|SS\_BLACKFRAME| SS\_CENTER,10,10,130,60, hWnd, NULL, hInst, NULL);  ShowWindow(hStaticText,SW\_SHOWNORMAL);  }  break;  case WM\_PAINT:  hdc = BeginPaint(hWnd, &ps);  // TODO: Add any drawing code here...  GetClientRect(hWnd, &rt);  DrawText(hdc, szHello, strlen(szHello), &rt, DT\_CENTER);  EndPaint(hWnd, &ps);  break;  case WM\_CLOSE:  DestroyWindow(hWnd);  break;  case WM\_DESTROY:  PostQuitMessage(0);  break;  default:  return DefWindowProc(hWnd, message, wParam, lParam);  }  return 0;  } |

17. ทดสอบโปรแกรม บันทึกผล

|  |
| --- |
| Ans. เป็นคำสั่งสร้างกรอบข้อความในหน้าต่าง window โดยเรียกใช้ผ่านคำสั่ง hStaticText ตรงส่วน CREATE เรียกใช้ในส่วนของฟังก์ชั่น static เพื่อกำหนดลักษณะของกรอบข้อความในหน้าต่าง window |

18. ทดลองปรับแต่ง ตาม <https://msdn.microsoft.com/en-us/library/windows/desktop/bb760773(v=vs.85).aspx>

19. ทดสอบโปรแกรม บันทึกผล

|  |
| --- |
| Ans. SS\_NOPREFIX เป็นคำสั่งจำกัดเครื่องหมายพิเศษ (&) ถ้ามีข้อความเข้ามาแล้วมีเครื่องหมายพิเศษฟังก์ชันนี้จะจำกัดไปและถ้าหากพิมพ์ข้อความเกินขอบเขตของกรอบก็จะตัดส่วนของข้อความที่เกินทิ้งไป |

20. แก้ไขโปรแกรม

|  |
| --- |
| case WM\_CREATE:  if(bFirstTime)  {  bFirstTime=FALSE;  hStaticText=CreateWindow("Static","This is a static text(LABEL)", WS\_CHILD| SS\_CENTER,10,10,130,60, hWnd, NULL, hInst, NULL);  hStaticIcon=CreateWindow("Static","MyIcon", WS\_CHILD| SS\_ICON ,150,10,0,0, hWnd, NULL, hInst, NULL);  hStaticRect=CreateWindow("Static","", WS\_CHILD| SS\_GRAYRECT ,10,80,200,30, hWnd, NULL, hInst, NULL);  ShowWindow(hStaticText,SW\_SHOWNORMAL);  ShowWindow(hStaticIcon,SW\_SHOWNORMAL);  ShowWindow(hStaticRect,SW\_SHOWNORMAL);  }  break; |

21.เพิ่มไฟล์ static.rc (ใช้ไฟล์ ico ที่อยู่ภายในเครื่อง)

|  |
| --- |
| MyIcon ICON DrawBall.ico |

22. ทดสอบโปรแกรม และบันทึกผล

|  |
| --- |
| Ans. เป็นการสร้างช่อง text ในหน้าต่าง window จากคำสั่ง hStaticText สร้างด้วยฟังก์ชัน static [SS\_CENTER] ให้เกิดช่อง text จากนั้นทำคำสั่ง hStaticIcon เพื่อใส่รูป icon ในหน้าต่าง window ด้วย การใส่รูป icon ต้องกำหนดไฟล์รูปที่ต้องการให้ชัดเจนและตรงกับโค้ดโปรแกรม โดยใส่คำสั่งเรียกใช้รูป icon ในไฟล์ static.rc |

23 เพิ่มไฟล์ Static.h

|  |
| --- |
| #define MyMenu 101  #define IDM\_CHANGE 100  #define IDM\_SEND 200  #define IDM\_QUIT 300 |

23. แก้ไขไฟล์ Static.rc

|  |
| --- |
| #include "Static.h"  MyIcon ICON DrawBall.ico  MyMenu MENU  BEGIN  MENUITEM "&Change Title", IDM\_CHANGE  MENUITEM "&Send Message", IDM\_SEND  MENUITEM "&Quit", IDM\_QUIT  END |

24. Include “Static.h” ในไฟล์ main.cpp

25. เพิ่มเติมโปรแกรม

|  |
| --- |
| case WM\_COMMAND:  switch(wParam)  {  case IDM\_CHANGE:  SetWindowText(hStaticText,"New text from SetWindowText()");  break;  case IDM\_SEND:  SendMessage(hStaticText,WM\_SETTEXT,0,(LONG)(LPSTR)"New Text Via SendMessage()");  break;  case IDM\_QUIT:  DestroyWindow(hWnd);  break;  }  break; |

26. ทดสอบโปรแกรม และบันทึกผล

|  |
| --- |
| Ans. เป็นการสร้างแถบเมนูด้านบนขึ้นมา ในไฟล์ main.cpp ของ project แรกที่เราทำ โดยเรียกใช้การสร้างเมนูผ่านคำสั่ง WM\_COMMAND และเพิ่ม wcex.lpszMenuName = MAKEINTRESOURCE(MyMenu); ใน WNDCLASSEX แทนค่าเดิมที่เป็นค่า null |

27. New Project

28. Button.h

|  |
| --- |
| #define MyMenu 100  #define IDM\_TEST 101  #define IDM\_QUIT 102  #define DONE\_BUTTON 400  #define STATUS\_BUTTON 401  #define RAD01 402  #define RAD02 403  #define CHECK\_BOX 404 |

29. Button.rc

|  |
| --- |
| #include "button.h"  MyIcon ICON DrawBall.ico  MyMenu MENU  BEGIN  MENUITEM "&Test", IDM\_TEST  MENUITEM "&Quit", IDM\_QUIT  END |

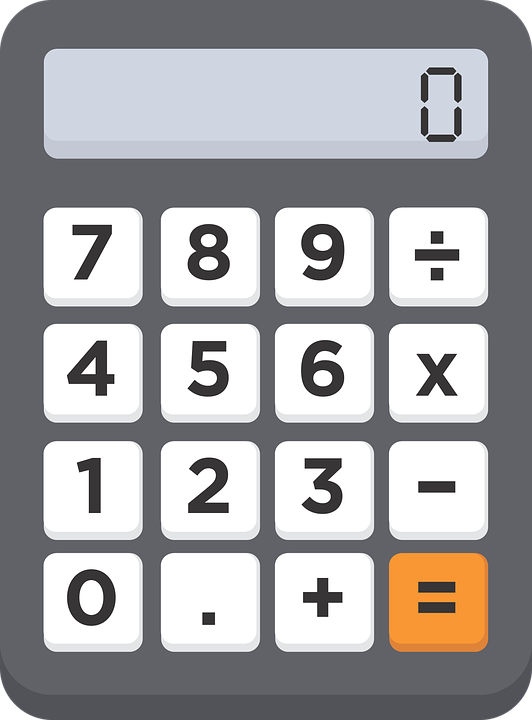
30.main.cpp

|  |
| --- |
| #include <windows.h>  #include <tchar.h>  #include "button.h"  // Global Variables:  HINSTANCE hInst; // current instance  TCHAR szTitle[]="20170921\_L2"; // The title bar text  TCHAR szWindowClass[]="WinApp"; // the class name  // Foward declarations of functions included in this code module:  ATOM MyRegisterClass(HINSTANCE hInstance);  BOOL InitInstance(HINSTANCE, int);  LRESULT CALLBACK WndProc(HWND, UINT, WPARAM, LPARAM);  int APIENTRY WinMain(HINSTANCE hInstance,  HINSTANCE hPrevInstance,  LPSTR lpCmdLine,  int nCmdShow)  {  // TODO: Place code here.  MSG msg;  // Register Class  MyRegisterClass(hInstance);  // Perform application initialization:  if (!InitInstance (hInstance, nCmdShow))  {  return FALSE;  }  // Main message loop:  while (GetMessage(&msg, NULL, 0, 0))  {  TranslateMessage(&msg);  DispatchMessage(&msg);  }  return msg.wParam;  }  //  // FUNCTION: MyRegisterClass()  //  // PURPOSE: Registers the window class.  //  // COMMENTS:  //  // This function and its usage is only necessary if you want this code  // to be compatible with Win32 systems prior to the 'RegisterClassEx'  // function that was added to Windows 95.  //  ATOM MyRegisterClass(HINSTANCE hInstance)  {  WNDCLASSEX wcex;  wcex.cbSize = sizeof(WNDCLASSEX);  wcex.style = CS\_HREDRAW | CS\_VREDRAW;  wcex.lpfnWndProc = (WNDPROC)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 = MAKEINTRESOURCE(MyMenu);  wcex.lpszClassName = szWindowClass;  wcex.hIconSm = LoadIcon (NULL, IDI\_APPLICATION);  return RegisterClassEx(&wcex);  }  //  // FUNCTION: InitInstance(HANDLE, int)  //  // PURPOSE: Saves instance handle and creates main window  //  // COMMENTS:  //  // In this function, we save the instance handle in a global variable and  // create and display the main program window.  //  BOOL InitInstance(HINSTANCE hInstance, int nCmdShow)  {  HWND hWnd;  hInst = hInstance; // Store instance handle in our global variable  hWnd = CreateWindow(szWindowClass, szTitle, WS\_OVERLAPPEDWINDOW,  CW\_USEDEFAULT, 0, CW\_USEDEFAULT, 0, NULL, NULL, hInstance, NULL);  if (!hWnd)  {  return FALSE;  }  ShowWindow(hWnd, nCmdShow);  UpdateWindow(hWnd);  return TRUE;  }  //  // FUNCTION: WndProc(HWND, unsigned, WORD, LONG)  //  // PURPOSE: Processes messages for the main window.  //  // WM\_COMMAND - process the application menu  // WM\_PAINT - Paint the main window  // WM\_DESTROY - post a quit message and return  //  //  LRESULT CALLBACK WndProc(HWND hWnd, UINT message, WPARAM wParam, LPARAM lParam)  {  PAINTSTRUCT ps;  HDC hdc;  RECT rt;  char szHello[]="Hello, C-Free!";  static BOOL bFirstTime=TRUE;  static HWND hGroup, hRad1, hRad2, hCheck, hBut1, hBut2;    switch (message)  {  case WM\_CREATE:  if(bFirstTime)  {  bFirstTime=FALSE;  hGroup=CreateWindow("BUTTON","Group Box", WS\_CHILD| BS\_GROUPBOX,10,10,120,100, hWnd, NULL, hInst, NULL);  hRad1=CreateWindow("BUTTON","Radio &1", WS\_CHILD| BS\_RADIOBUTTON,20,40,80,20, hWnd,(HMENU) RAD01, hInst, NULL);  hRad2=CreateWindow("BUTTON","Radio &2", WS\_CHILD| BS\_RADIOBUTTON,20,70,80,20, hWnd,(HMENU) RAD02, hInst, NULL);  hCheck=CreateWindow("BUTTON","&Check Box", WS\_CHILD| BS\_AUTO3STATE,20,120,100,20, hWnd,(HMENU) CHECK\_BOX, hInst, NULL);  hBut1=CreateWindow("BUTTON","&Show Status", WS\_CHILD| BS\_PUSHBUTTON,140,20,100,40, hWnd,(HMENU) RAD01, hInst, NULL);  hBut2=CreateWindow("BUTTON","&Done", WS\_CHILD| BS\_PUSHBUTTON,140,110,100,40, hWnd,(HMENU) RAD02, hInst, NULL);    ShowWindow(hGroup,SW\_SHOWNORMAL);  ShowWindow(hRad1,SW\_SHOWNORMAL);  ShowWindow(hRad2,SW\_SHOWNORMAL);  ShowWindow(hCheck,SW\_SHOWNORMAL);  ShowWindow(hBut1,SW\_SHOWNORMAL);  ShowWindow(hBut2,SW\_SHOWNORMAL);  }  break;  case WM\_COMMAND:  switch(wParam)  {  case RAD01:  SendMessage(hRad1,BM\_SETCHECK,TRUE,0L);  SendMessage(hRad2,BM\_SETCHECK,FALSE,0L);  break;  case RAD02:  SendMessage(hRad1,BM\_SETCHECK,FALSE,0L);  SendMessage(hRad2,BM\_SETCHECK,TRUE,0L);  break;  case CHECK\_BOX:  break;  case IDM\_QUIT:  DestroyWindow(hWnd);  break;  }  break;  case WM\_PAINT:  hdc = BeginPaint(hWnd, &ps);  // TODO: Add any drawing code here...    GetClientRect(hWnd, &rt);  DrawText(hdc, szHello, strlen(szHello), &rt, DT\_CENTER);  EndPaint(hWnd, &ps);  break;  case WM\_CLOSE:  DestroyWindow(hWnd);  break;  case WM\_DESTROY:  PostQuitMessage(0);  break;  default:  return DefWindowProc(hWnd, message, wParam, lParam);  }  return 0;  } |

31. ทดสอบโปรแกรม บันทึกผล

|  |
| --- |
| Ans. เป็นการสร้างปุ่มกดในหน้าต่าง โดยใช้คำสั่ง hGroup ในการสร้าง Group Box โดยมีตัวเลือก Radio1และRadio2 ด้วยคำสั่ง hRad1 กับ hRad2 ส่วนของ Check Boxเป็นการใส่ข้อมูลด้วยการติ๊ก ใช้คำสั่ง hCheck ด้วยฟังก์ชั่น BS\_AUTO3STATE และ hBut1และ2เป็นการสร้างกล่องกดรับข้อมูล |

32. เขียนโปรแกรมสร้างเครื่องคิดเลขตามรูป โดยใช้ PUSHBUTTON, STATICTEXT



# สรุปผลการทดลอง

Ans. จากการทดลอง ได้รู้คำสั่งการใส่เสียงให้โปรแกรมด้วยคำสั่ง MessageBeep, คำสั่งสร้างลูกบอลวิ่งบนหน้าจอ , การใช้ Bruch/Oldbruch การสร้างกล่องข้อความและปุ่มกดบนหน้าต่าง window คำสั่งPAINTSTRUCT เป็นฟังก์ชันโครงสร้างมีข้อมูลสำหรับแอพลิเคชัน ทำให้รู้จักฟังก์ชัน static และรู้ว่ามีความสามารถหลากหลายอย่าง จัดการกล่องข้อความในหน้าต่างหรือการปรับแต่งกล่องข้อความ การสร้างสถานะกล่องข้อความได้ด้วยการใช้คำสั่ง show window การใส่ข้อมูลด้วยการใช้ติ๊กหน้าข้อความ และการใส่รูป icon ให้กับหน้าต่างพร้อมข้อความโดยการใส่ชื่อไฟล์พร้อมสกุลไฟล์ .ico

# คำถามท้ายการทดลอง

1. บอกความแตกต่างระหว่างฟังก์ชัน GetMessage() และ PeekMessage()

Ans. GetMessage() คือ คำสั่งที่รอให้มีข้อความเข้ามาก่อนถึงจะส่งข้อความเข้ามา ส่วน PeekMessage() คือ คำสั่งที่ไม่รอให้มีข้อความเข้ามา

1. บอกลักษณะการนำฟังก์ชัน GetMessage() และ PeekMessage() ไปใช้งาน

Ans. GetMessage() ใช้งานกับงานที่ต้องการให้มีการแสดงข้อความก่อน ส่วน PeekMessage() ใช้งานกับงานที่ไม่ต้องการแสดงข้อความก่อนล่วงหน้าแต่นำข้อความไปประมวลผลเลย

1. บอกหน้าที่ GetSystemMetrics()

Ans. คำสั่ง GetSystemMetrics() เป็นคำสั่งที่มีหน้าที่จัดการกับขนาดของวัตถุ เช่น icon, cursor ฯลฯ แต่ความกว้างและความสูง วัดขนาดเป็นหน่วย pixel

1. เขียนโปรแกรมเครื่องคิดเลข