ใบงานที่ 4 Menu

# **วัตถุประสงค์**

1. สร้าง Menu

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\_04)

· 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>  #include "resource.h"  // Global Variables:  HINSTANCE hInst; // current instance  TCHAR szTitle[]="L9"; // 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 = NULL;  wcex.lpszMenuName = MAKEINTRESOURCE(IDR\_MYMENU);  wcex.lpszClassName = szWindowClass;  wcex.hIconSm = LoadIcon (NULL, IDI\_APPLICATION);  // wcex.hIcon = LoadIcon(GetModuleHandle(NULL), MAKEINTRESOURCE(IDI\_MYICON));  // wcex.hIconSm = (HICON)LoadImage(GetModuleHandle(NULL), MAKEINTRESOURCE(IDI\_MYICON), IMAGE\_ICON, 16, 16, 0);  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!";    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\_COMMAND:  switch(LOWORD(wParam))  {  case ID\_FILE\_EXIT:  PostMessage(hWnd, WM\_CLOSE, 0, 0);  break;  case ID\_STUFF\_GO:  break;  }  break;  case WM\_CLOSE:  DestroyWindow(hWnd);  break;  case WM\_DESTROY:  PostQuitMessage(0);  break;  default:  return DefWindowProc(hWnd, message, wParam, lParam);  }  return 0;  } |

5. เพิ่มไฟล์ resource.h

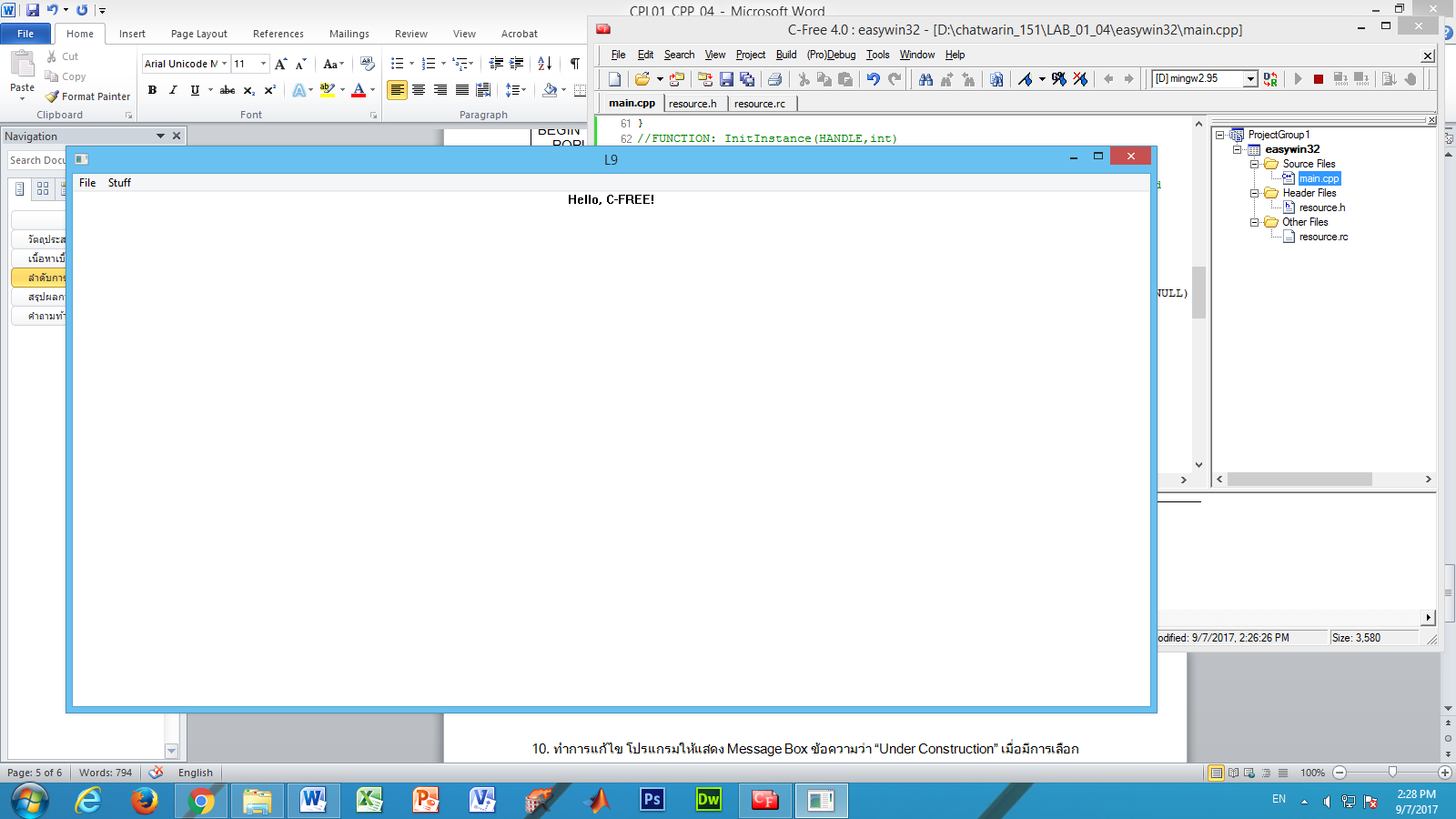
|  |
| --- |
| #define IDR\_MYMENU 101  #define IDI\_MYICON 201  #define ID\_FILE\_EXIT 9001  #define ID\_STUFF\_GO 9002 |

6. เพิ่มไฟล์ resource.rc

|  |
| --- |
| #include "resource.h"  IDR\_MYMENU MENU  BEGIN  POPUP "&File"  BEGIN  MENUITEM "E&xit", ID\_FILE\_EXIT  END  POPUP "&Stuff"  BEGIN  MENUITEM "&Go", ID\_STUFF\_GO  MENUITEM "G&o somewhere else", 0, GRAYED  END  END  //IDI\_MYICON ICON "menu\_one.ico" |

7. ทำการทดสอบโปรแกรม บันทึกผล

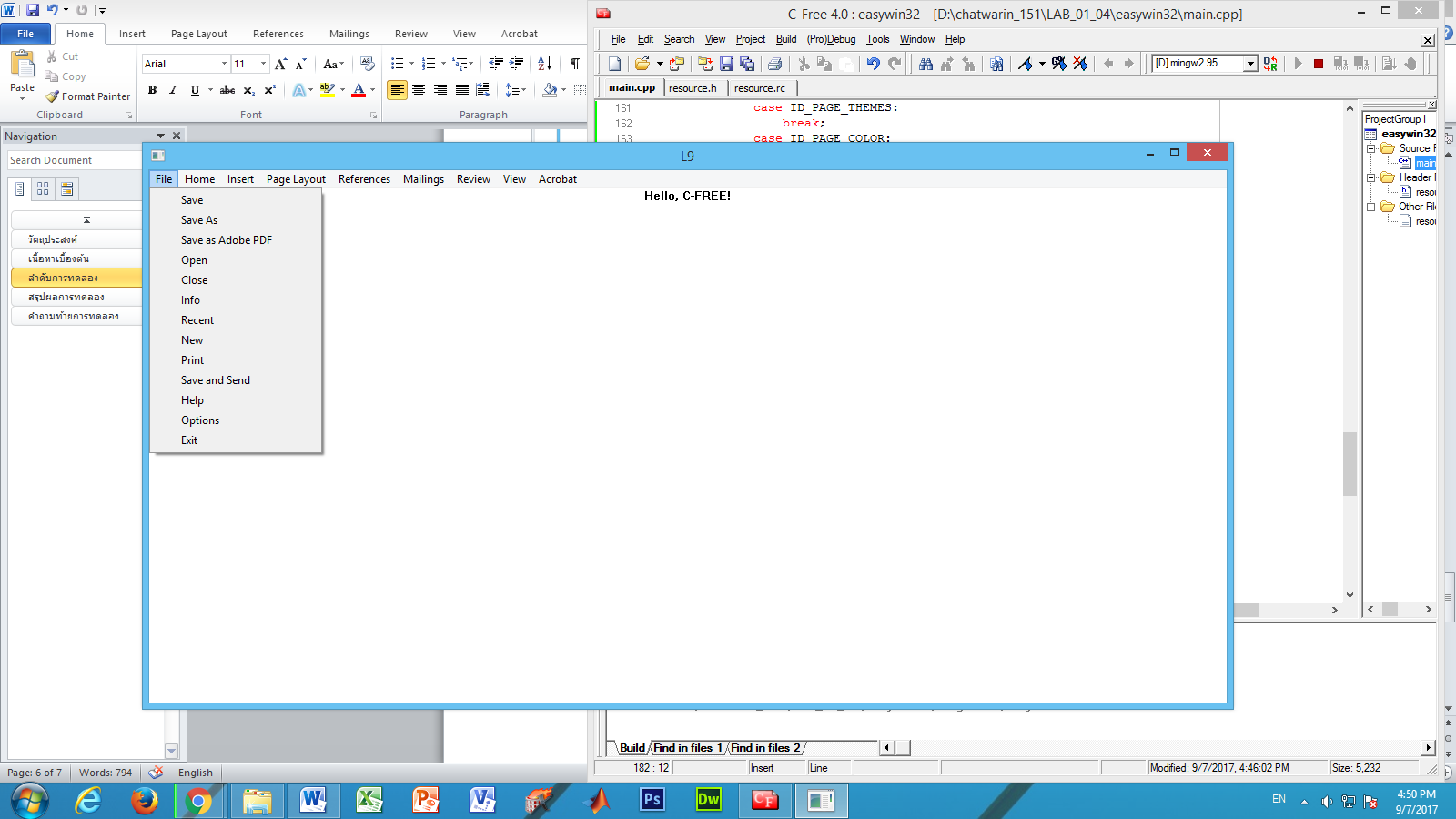
Ans. เป็นการเขียนโปรแกรมให้มีการขึ้นแถบเมนูด้านบนโดยใช้คำสั่ง case WM\_COMMAND ในการกำหนดตัวแปรชื่อเมนูที่ต้องการ ในส่วนของเมนู File จะมีการทำงานของปุ่ม Exit อยุ่ด้วย โดยกำหนดการทำงานให้ปุ่ม Exit เป็นการกดปิดหน้าต่าง window ด้วยคำสั่ง PostMessage(hWnd, WM\_CLOSE, 0, 0) ส่วนการเพิ่มไฟล์ resource.h เป็นการกำหนดตัวแปรที่เราต้องการเรียกใช้ในไฟล์ main การเพิ่มไฟล์ resource.rc เป็นการกำหนดให้มีปุ่มเมนูให้เลือกในแถบเมนูด้านบนที่เราได้กำหนดไว้แล้ว

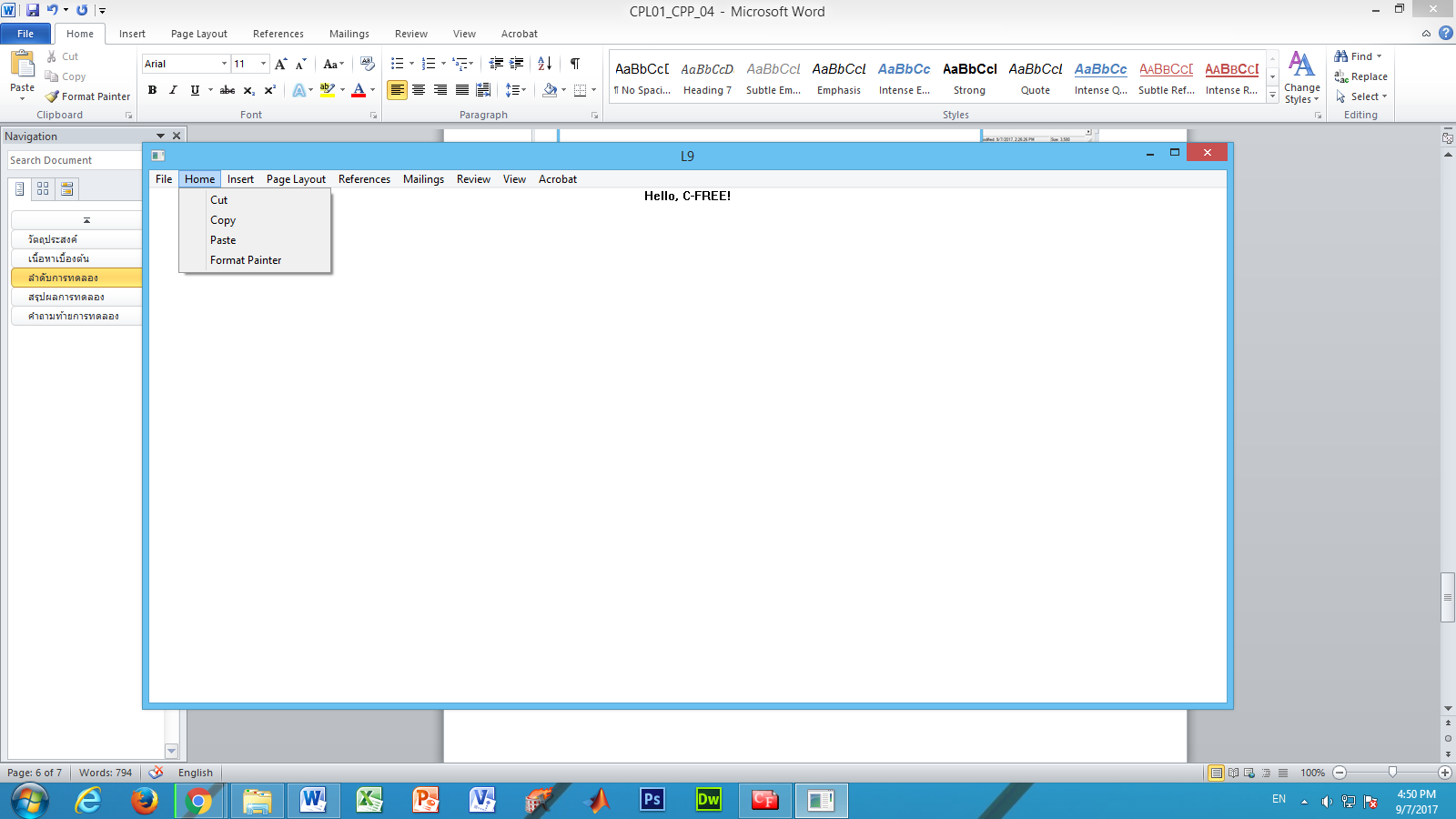


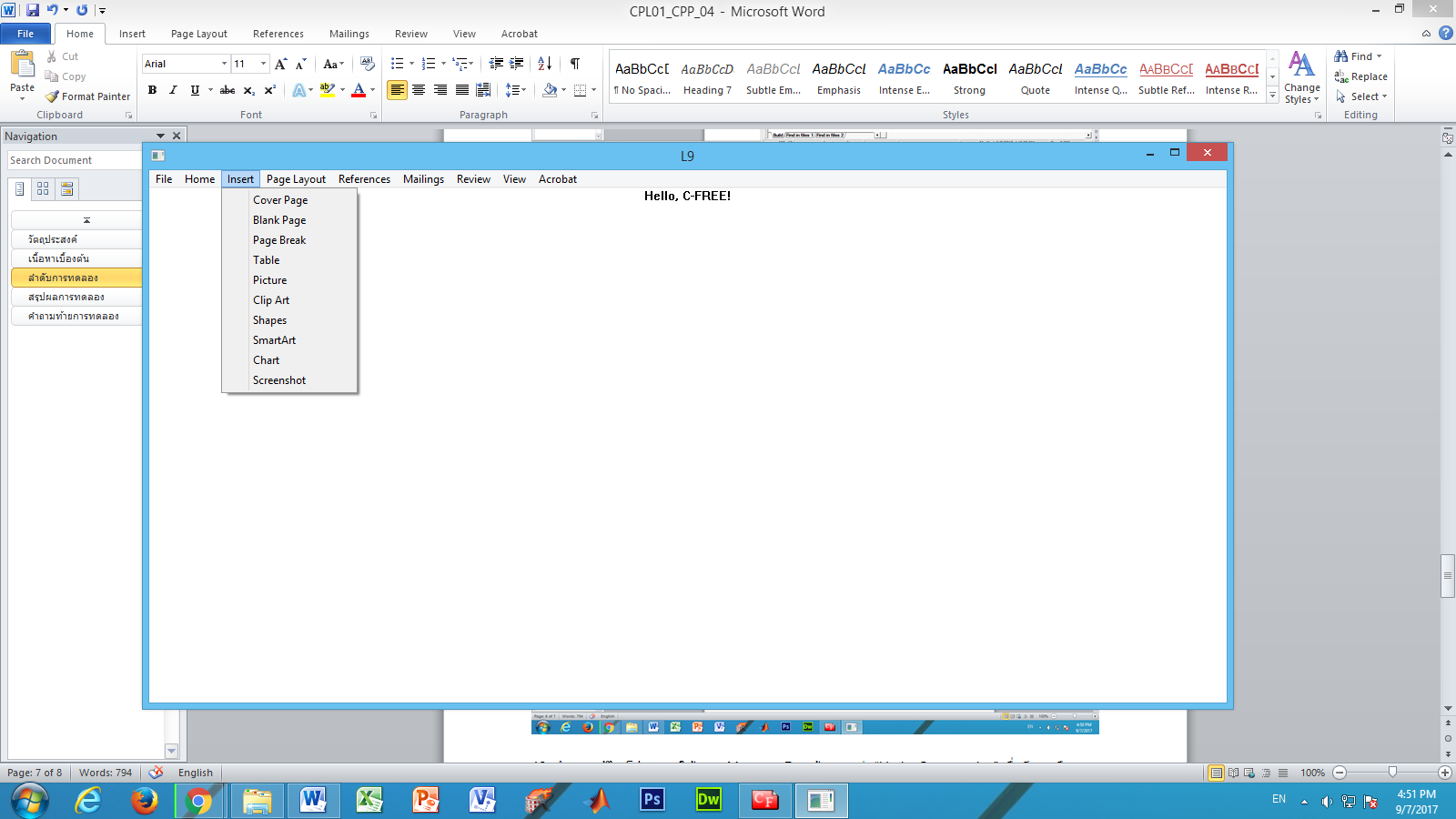
8. ทำการสร้างเมนูเลียนแบบ Microsoft Word

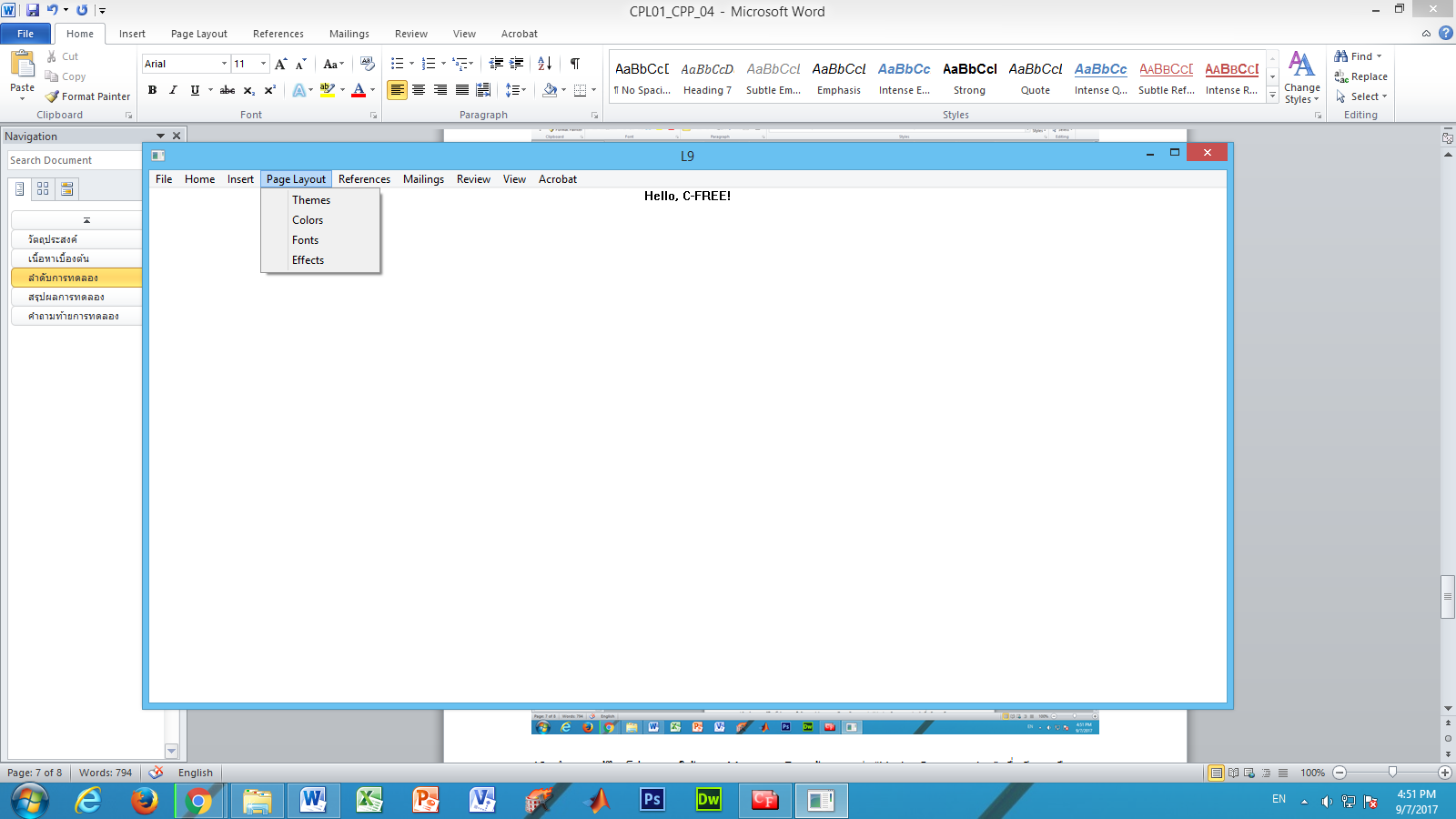
9. ทำการทดสอบโปรแกรม บันทึกผล

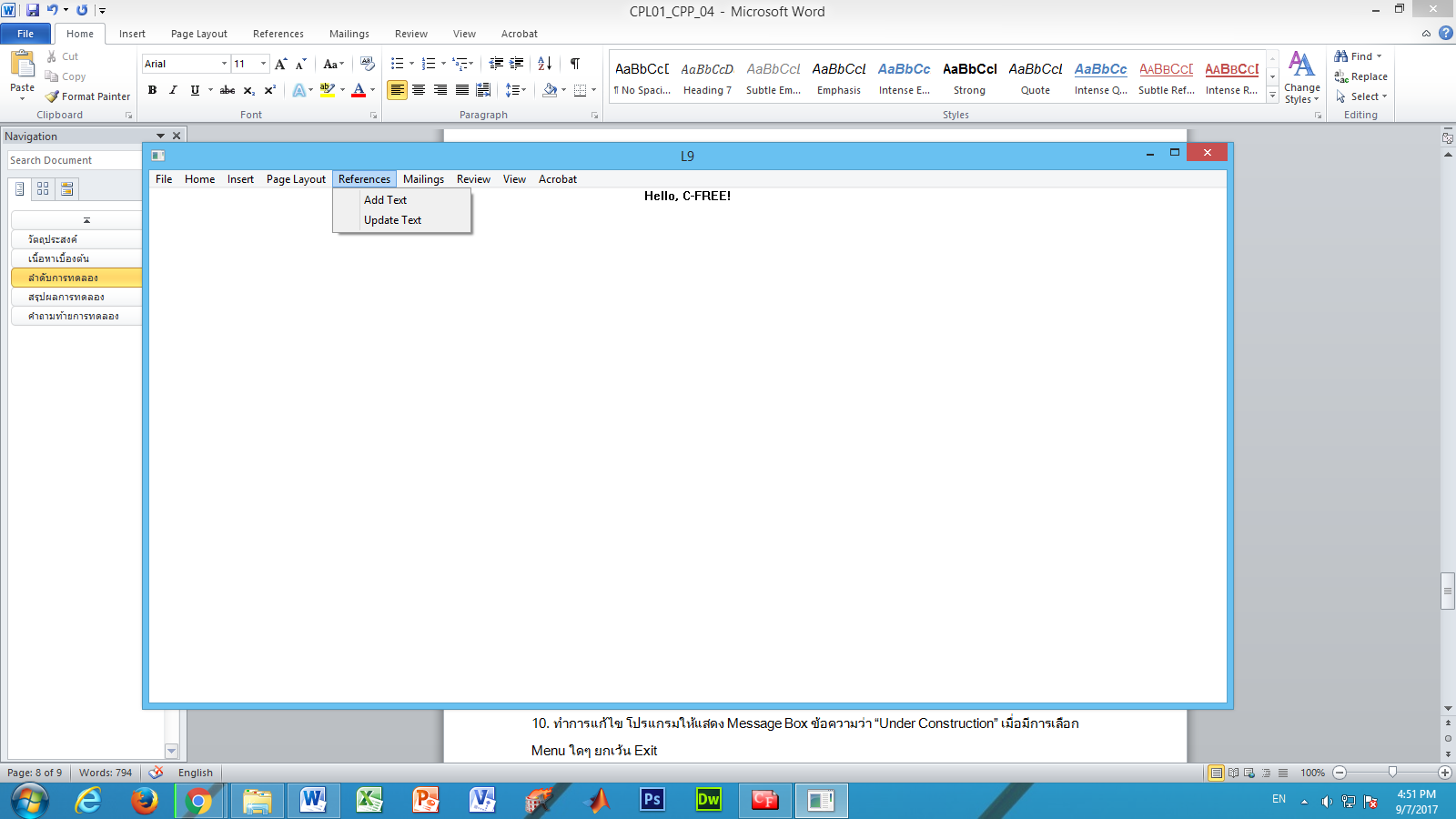
Ans. เป้นการเขียนโปรแกรมเลียนแบบ Microsoft word 2010 ด้วยการเพิ่มการกำหนดตัวแปรในไฟล์ resource.h แล้วเรียกใช้ในไฟล์ resource.rc ในการเพิ่มปุ่มเมนูที่ต้องการ แล้วจึงไปกำหนดการทำงานของปุ่มแต่ละปุ่มที่ได้เพิ่มเข้ามาในไฟล์ main ต่อจากส่วนของใน case WM\_COMMAND

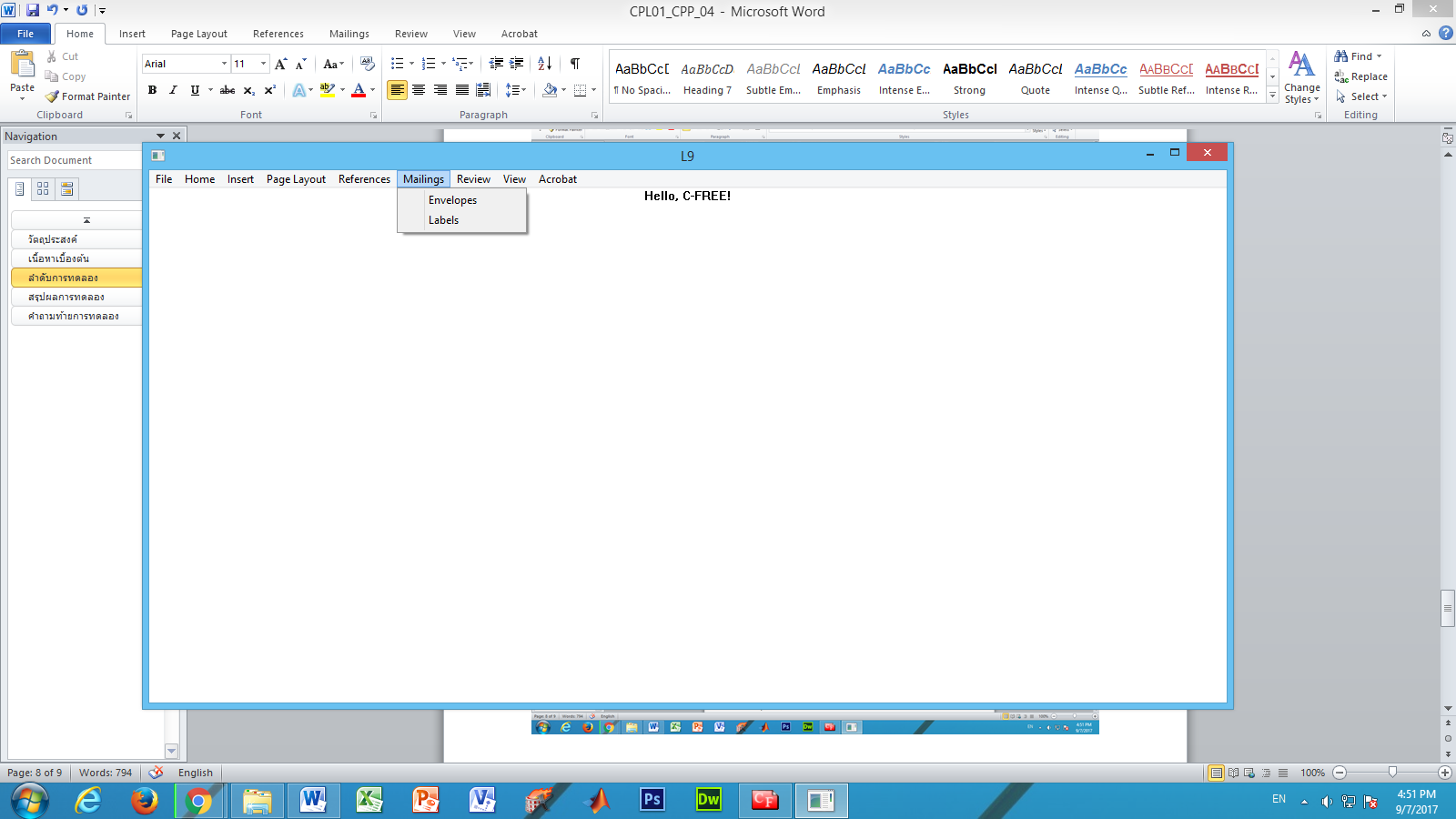


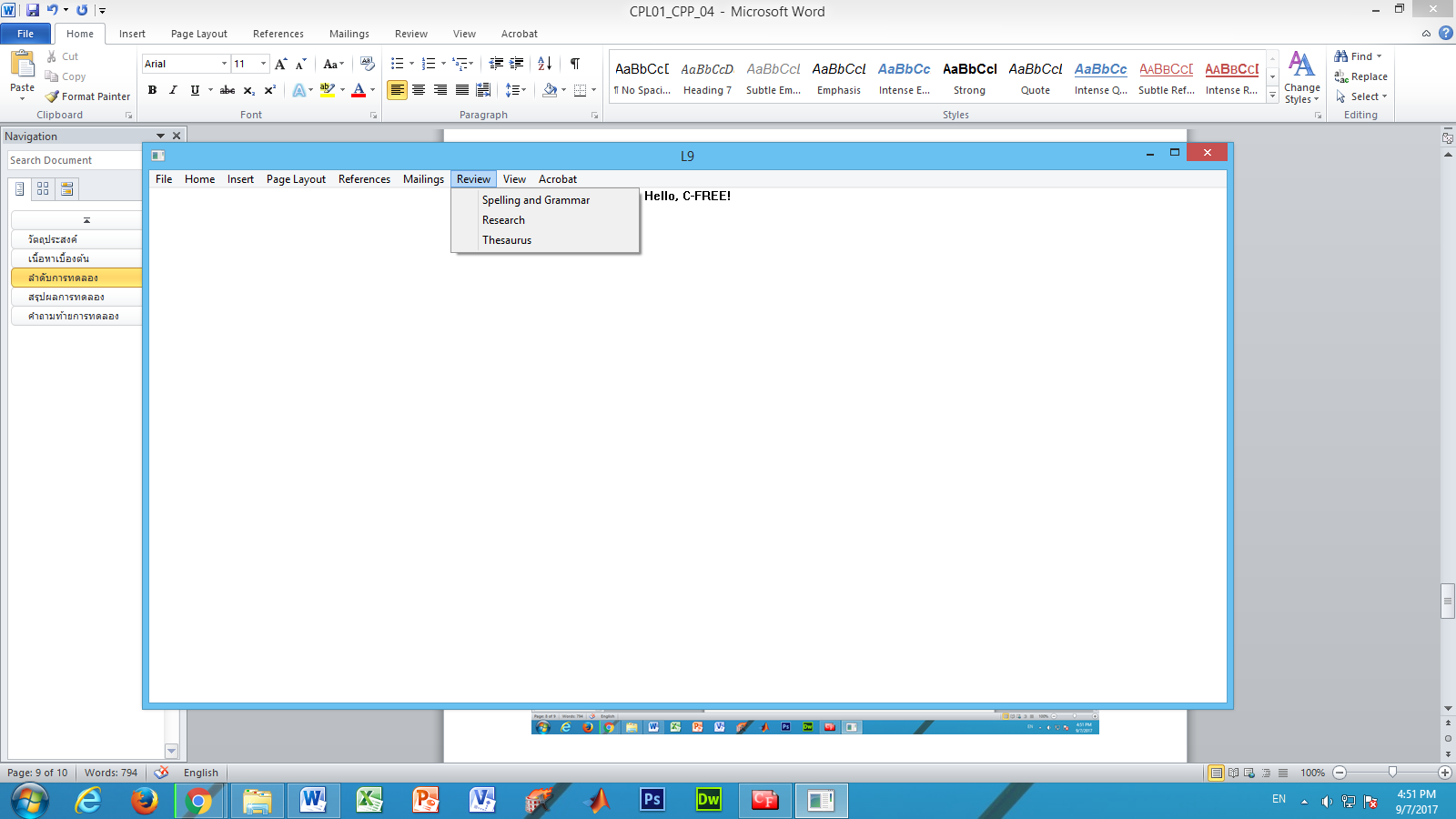


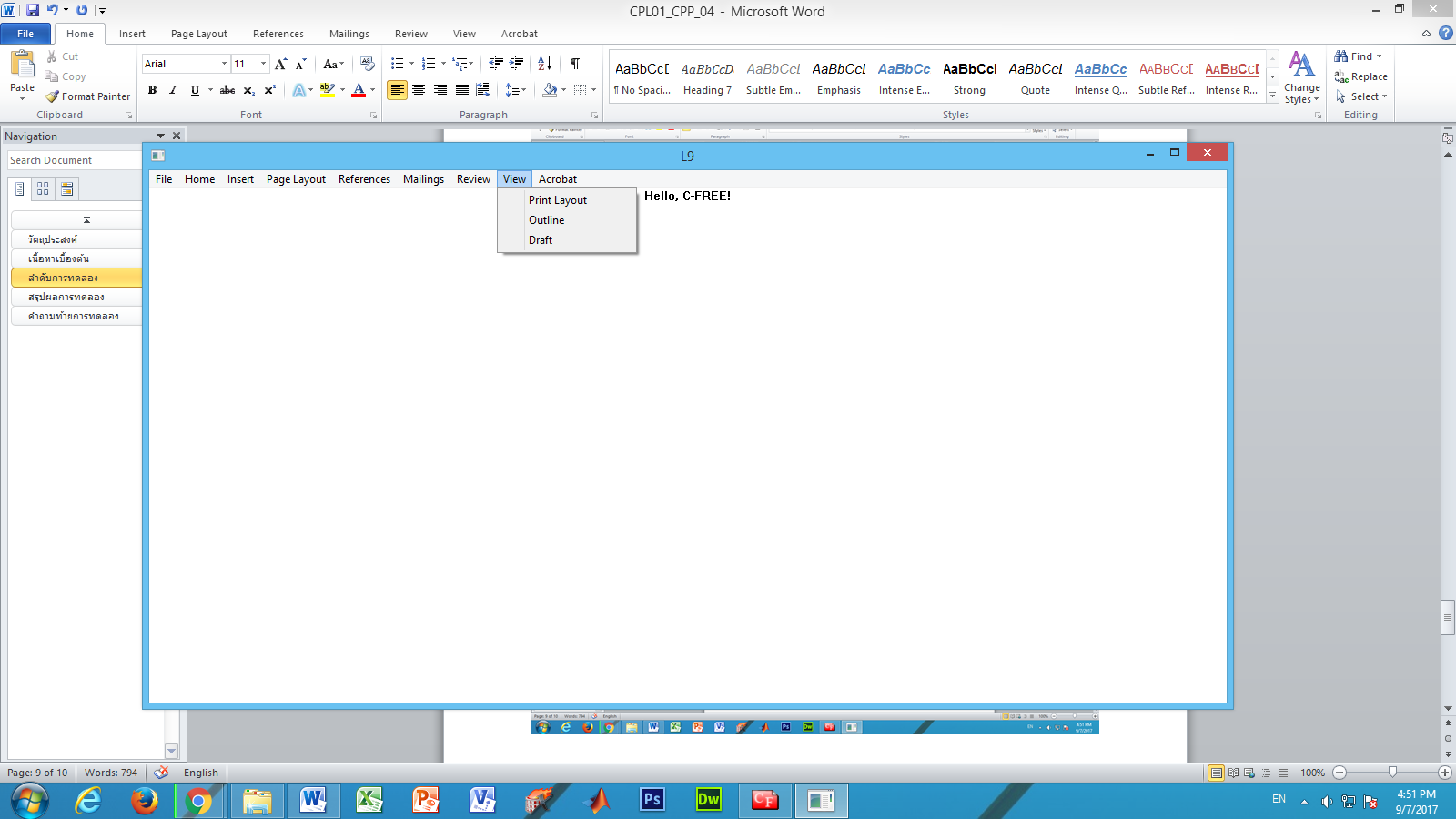


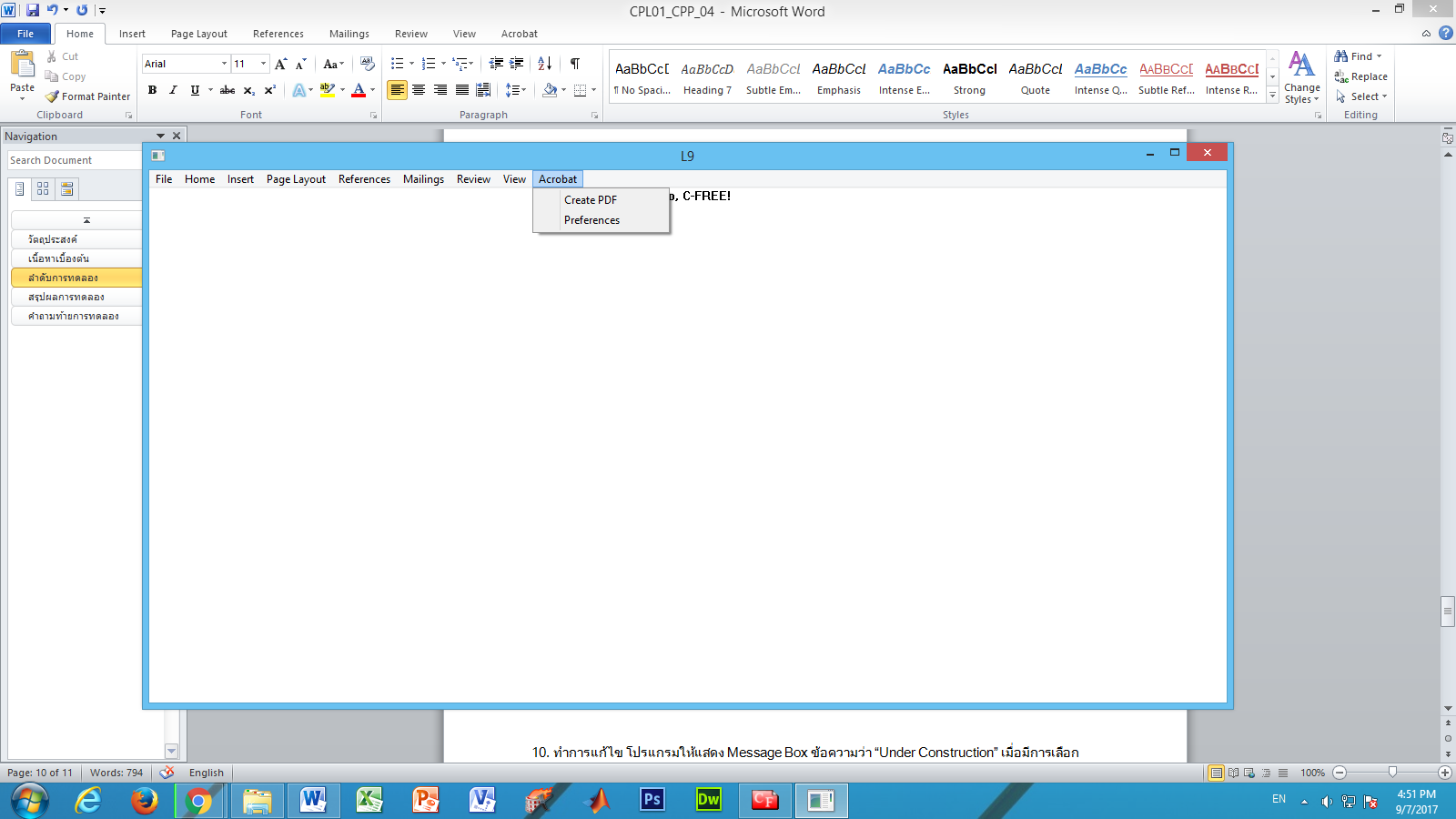




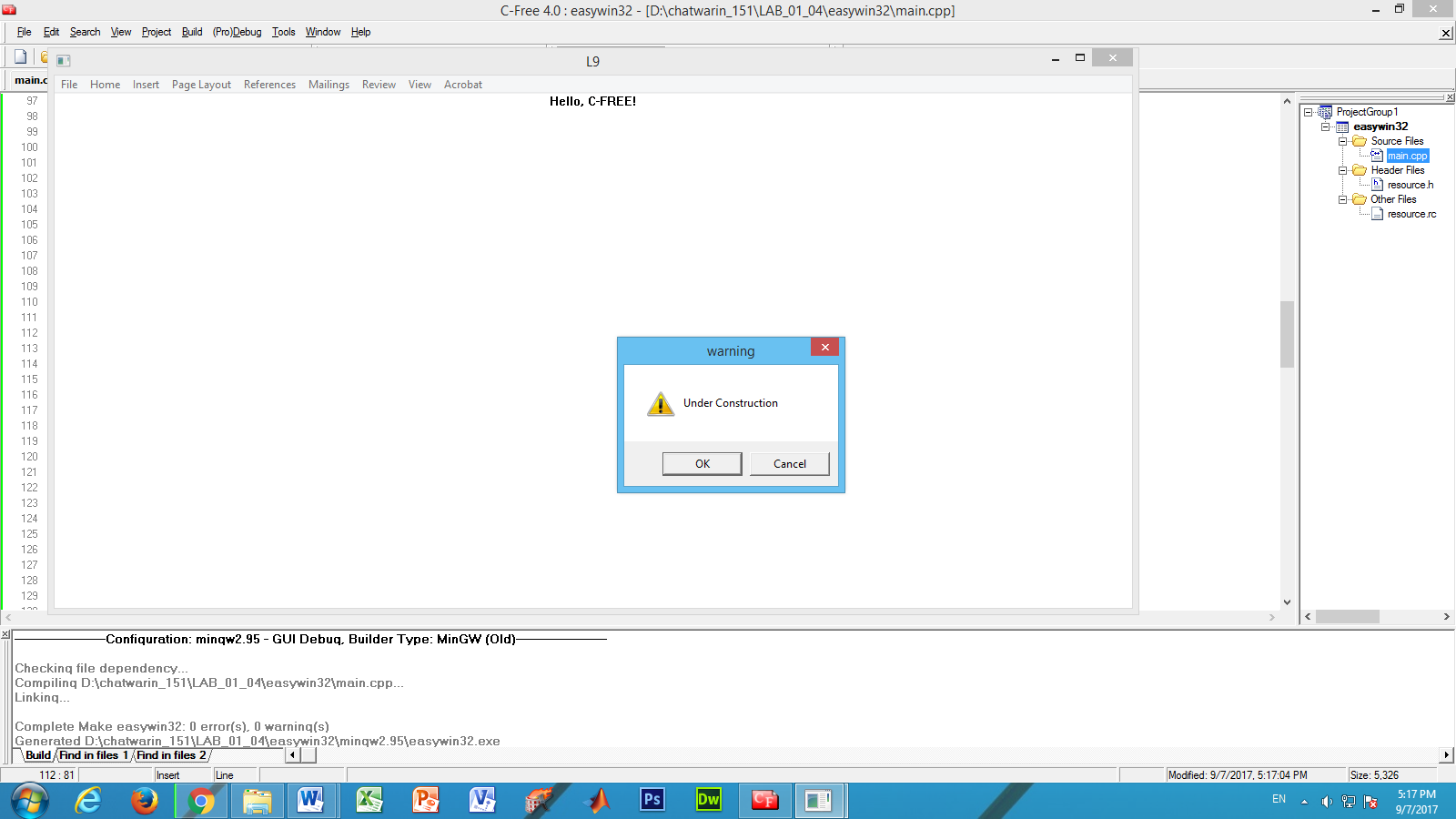








10. ทำการแก้ไข โปรแกรมให้แสดง Message Box ข้อความว่า “Under Construction” เมื่อมีการเลือก Menu ใดๆ ยกเว้น Exit



11. สร้าง Project ใหม่

12. เขียนโปรแกรมวาดภาพด้วยเมาส์เหมือนใบงานที่ 3 แต่เพิ่มความสามารถในการเลือกสีปากกาจาก Menu

13. ทดสอบการทำงาน บันทึกผล

14. สร้าง Project ใหม่

15. สร้างตัวแปร x1, y1, x2, y2

16. เขียนโปรแกรมวาดรูปสี่เหลี่ยม ซึ่งกำหนดค่าตำแหน่งโดยตัวแปรในข้อ 15 (WM\_PAINT)

17. เขียนโปรแกรมกำหนดค่าตัวแปรในข้อ 15 ในขณะทำงานโดยการใช้ WM\_MOUSEMOVE, WM\_LBUTTONDOWN, WM\_CHR, หรือการใช้ Menu เมื่อกำหนดค่าแล้วให้เรียกใช้ฟังก์ชัน InvalidateRect(hWnd, NULL, TRUE);

18. ทดสอบการทำงาน บันทึกผล

# สรุปผลการทดลอง

Ans. จากการทดลองพบว่า เราสามารถเขียนโปรแกรมใช้แถบเมนูตามที่เราต้องการได้ โดยใช้ฟังก์ชั่นคำสั่งพิเศษ WM\_COMMAND ได้รู้ถึงการกำหนดตัวแปรที่เราต้องการเรียกใช้ได้เองในส่วนของ header file ได้รู้การสร้างปุ่มเครื่องมือในแถบเมนูได้ด้วยการกำหนดในส่วนของไฟล์สกุลไฟล์ .rc โดยกำหนดในส่วนของคำสั่ง IDR\_MYMENU MENU

# คำถามท้ายการทดลอง

1. เขียนโปรแกรมบันทึกข้อมูล x1, y1, x2, y2 ลงในไฟล์ชื่อ position.txt
2. เขียนโปรแกรมอ่านข้อมูลจากไฟล์ position.txt ใส่ลงในตัวแปร x1, y1, x2, y2 เมื่อโปรแกรมทำงาน
3. เขียนโปรแกรมวาดภาพ โดยมีความสามารถในการบันทึกข้อมูล และอ่านข้อมูลมาแสดงผล