```
// Vahe Karamian - Windows 32 Multithreaded Programming
// Filename: launchTerminateNotePad.cpp

// Launching the Windows Notepad

#include <windows.h>
#include <stdio.h>
#include <stdlib.h>
#include <process.h>

int main( void )
{
  STARTUPINFO         si;
  PROCESS_INFORMATION pi;

  // Fill out the STARTUPINFO structure with default values
  ZeroMemory( &si, sizeof(si) );
  si.cb = sizeof( si );

  // Start Notepad
  CreateProcess( NULL,            // lpApplicationName
                 "notepad.exe",   // lpCommandLine
                 NULL,            // lpProcessAttributes
                 NULL,            // lpThreadAttributes
                 FALSE,           // bInheritHandles
                 0,               // dwCreationFlags
                 NULL,            // lpEnvironment
                 NULL,            // lpCurrentDirectory
                 &si,             // lpStartupInfo
                 &pi );           // lpProcessInformation

  // Print some information
  printf( "Notepad created as process id 0x%08x and thread id 0x%08x.\n",
          pi.dwProcessId,
          pi.dwThreadId );

  // This will wait for five seconds
  Sleep( 5000 );

  // Forcibly terminate the notepad
  TerminateProcess( pi.hProcess, 0x00000007 );

  // Get notepad's exit code
  DWORD dwNotepadExitCode;
  if( GetExitCodeProcess( pi.hProcess, &dwNotepadExitCode ) )
    printf( "Notepad's exit code is 0x%08x.\n", dwNotepadExitCode );
  else
    printf( "Unable to get notepad's exit code.\n" );

  // Close the newly created process and thread handles, this will not
  // terminate notepad, it merely closes this process's handle to
  // notepad's process and thread
  CloseHandle( pi.hProcess );
  CloseHandle( pi.hThread );

  return( 0 );
}
```