/*
 * Simple program demonstrating Win32 anonymous pipes.
 *
 * Figures 3.25 & 3.26
 *
 * @author Gagne, Galvin, Silberschatz
 * Operating System Concepts - Eighth Edition
 * Copyright John Wiley & Sons - 2008.
 */

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

#define BUFFER_SIZE 25

int main(VOID)
{
    HANDLE ReadHandle, WriteHandle;
    STARTUPINFO si;
    PROCESS_INFORMATION pi;
    char message[BUFFER_SIZE] = "Greetings";
    DWORD written;

    /* set up security attributes so that pipe handles are inherited */
    SECURITY_ATTRIBUTES sa = {sizeof(SEcurity_ATTRIBUTES), NULL,TRUE};

    /* allocate memory */
    ZeroMemory(&pi, sizeof(pi));

    /* create the pipe */
    if ( !CreatePipe(&ReadHandle, &WriteHandle, &sa, 0)) {
        fprintf(stderr,"Create Pipe Failed\n");
        return 1;
    }

    /* establish the START_INFO structure for the child process */
    GetStartupInfo(&si);
    si.hStdError = GetStdHandle(STD_ERROR_HANDLE);
    si.hStdOutput = GetStdHandle(STD_OUTPUT_HANDLE);

    /* redirect the standard input to the read end of the pipe */
    si.hStdInput = ReadHandle;
    si.dwFlags = STARTF_USESTDHANDLES;

    /* we do not want the child to inherit the write end of the pipe */
    SetHandleInformation(WriteHandle, HANDLE_FLAG_INHERIT, 0);

    /* create the child process */
    if ( !CreateProcess(NULL, "\child.exe", NULL, NULL, TRUE, 0, NULL, NULL, NULL, NULL)
{  
  fprintf(stderr, "Process Creation Failed\n");  
  return -1;  
}

/* close the unused end of the pipe */
CloseHandle(ReadHandle);

/* the parent now wants to write to the pipe */
if (!WriteFile (WriteHandle, message, BUFFER_SIZE, &written, NULL))
  fprintf(stderr, "Error writing to pipe\n");

/* close the write end of the pipe */
CloseHandle(WriteHandle);

/* wait for the child to exit */
WaitForSingleObject(pi.hProcess, INFINITE);

/* close all handles */
CloseHandle(pi.hProcess);
CloseHandle(pi.hThread);
}