```
#include <iostream>
#include <stdio.h>

extern "C" {void myWrite(char* x, char* y); }
extern "C" {void myRead(char* x); }

int main()
{
       char *x;
       int *y;

       int i,j;

       char FileName[24] = "\\\\.\\PhysicalDrive0";   int n = 512;  //     unc-путь
\\?\Диск:[\Каталог][\Файл]
                                                                            //сетевой файл
«\\\\имя_удаленного_компа\\путь_к_ файлу\\ имя_файла».

//     char FileName[14]= "e:\\List.txt";  int n=20;
       char Filetext[20] = "qwertyuiopasdfghjk";

//     myWrite(FileName, Filetext);
       myRead(FileName);

       _asm
       {
              mov x, eax
              mov y, ebx

       }

       printf("%d ", y);
       printf("\n");

/*     for (j = 0; j < n;j++) {    */
              for (i = 0; i < n; i++)
              {
                     printf("%x ", *x);
                     x++;
              }

              printf("\n");

       /*     for (i = 0; i < 512; i++)
              {
                     printf("%x ", *x);
                     x++;
              }

              printf("\n");
*/

                                         /*     }  */
/*
         for (i = 0; i < 512; i++)
         {
                printf("%x ", *y);
                y++;
         }

         printf("\n");
         for (i = 0; i < 512; i++)
         {
                printf("%x ", *y);
                y++;
         }
         */
       return 0;
}
```

```
public myWrite
.686
.MODEL FLAT, C

option casemap: none

includelib Kernel32.lib

include e:\masm32\include\windows.inc
include e:\masm32\include\kernel32.inc

.STACK
.DATA

handleFile DWord 0
numBytes DWord 8
nByteWritten DWord 0
soob DWord 0

.CODE
myWrite Proc

push ebp
mov ebp, esp

xor eax, eax
xor ebx, ebx

mov eax, [ebp + 8]
mov ebx, [ebp + 12]

mov soob, ebx

push eax

invoke CreateFile, eax, GENERIC_WRITE, 0, NULL, OPEN_ALWAYS, FILE_ATTRIBUTE_NORMAL, 0

mov handleFile, eax

;invoke SetFilePointer, handleFile, 0, 0, FILE_END

invoke WriteFile, handleFile, soob, numBytes, ADDR nByteWritten, 0

invoke CloseHandle, handleFile

pop eax

mov  eax,soob
mov ebx,nByteWritten
```

```
pop ebp

ret

myWrite endp
end


public myRead
.686
.MODEL FLAT, C

option casemap: none

includelib Kernel32.lib

include e:\masm32\include\windows.inc
include e:\masm32\include\kernel32.inc
;CreateFile PROTO:DWORD,:DWORD, :DWORD, :DWORD, :DWORD,:DWORD,:DWORD
;ReadFile PROTO:DWORD,:DWORD, :DWORD, :DWORD, :DWORD
;CloseHandle PROTO:DWORD

.STACK
.DATA
handleFile DWord 0
numBytes DWord 512
nByteReaden DWord 0
;сюда читаем:
soob Dword 512 dup(?)
NumPos Dword 4097

.CODE
myRead Proc

push ebp
mov ebp, esp

xor eax, eax
xor ebx, ebx

mov eax, [ebp + 8]

push 0
push 128    ;FILE_ATTRIBUTE_NORMAL
push 3      ;OPEN_EXISTING
push 0      ;указатель на структуру SecurityAttributes
push 1      ;запрещает модификацию данных на этом устройстве всем другим объектам ядра
push 80000000h     ;GENERIC_READ
push eax
call CreateFile
;invoke CreateFile, eax, GENERIC_READ, 2, NULL, OPEN_EXISTING, FILE_ATTRIBUTE_NORMAL, 0
;invoke GetLastError
mov handleFile, eax

push NULL
push OFFSET  nByteReaden
push numBytes
push OFFSET soob
push handleFile
```

```
call  ReadFile
;invoke ReadFile, handleFile, ADDR soob, numBytes, ADDR nByteReaden, 0
push eax
invoke GetLastError
pop  eax

   push handleFile
call CloseHandle
;invoke CloseHandle, handleFile


mov ebx,nByteReaden
mov    eax, offset soob
pop ebp

ret 0

myRead endp
end
```