МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯРФ

филиалфедеральногогосударственногобюджетногообразовательногоучреждениявысшегообразования

«Национальныйисследовательскийуниверситет«МЭИ»вг.Смоленске

Кафедра вычислительной техники

Отчет

полабораторнойработе№2

Тема:«Исследование виртуальной памяти»

покурсу:«Операционные системы»

Студент: Воронцов А.А.

Группа: ИВТ-1-18

Преподаватель: Гаврилов А.И.

Смоленск,2019 г.

1. **Цель работы**

Целью работы является изучение основных функций ядраKernel32.dll для работы с виртуальной памятью. Рассматриваемые в данной лабораторной работе функции позволяют:

* получить информацию о состоянии системной памяти и виртуального адресного пространства любого процесса;
* напрямую резервировать регион адресного пространства;
* передавать зарезервированному региону физическую память;
* освобождать регионы адресного пространства;
* изменять атрибуты защиты страниц виртуальной памяти.

1. **Задание на лабораторную работу**
2. Создать приложение, работающее с виртуальной памятью.
3. Самостоятельно запрограммировать событие OnClick для кнопки Button6 («VirtualProtect»).
4. Добавить в приложение возможность вводить атрибут защиты страниц при программировании кнопки «VirtualProtect».
5. Добавить в приложение возможность вводить количество выделяемых с помощью функции VirtualAlloc байт (или страниц).
6. Определить, какие регионы в виртуальном адресном пространстве свободны (найти хотя бы несколько регионов).
7. **Анализ задания на лабораторную работу**

Для выполнения задания для самостоятельной работы требуется создать кнопку Button6 и ComboBox1. В ComboBox1 пользователь сможет выбирать атрибут защиты страниц.

При добавлении в приложение возможности вводить количество выделяемых с помощью функции VirtualAllocбайт следует изменить событие TForm1.Button4Click(Sender: TObject).Количество байт будет браться из Edin2(при отсутствии ввода значение по умолчанию– 4096).

Определить свободные регионы в виртуальном адресном пространстве можно с помощью цикла и функцииVirtualQuery.

1. **Модульная структура программы**
2. procedure Button1Click(Sender: TObject);
3. procedure FormActivate(Sender: TObject);
4. procedure Button2Click(Sender: TObject);
5. procedure Button3Click(Sender: TObject);
6. procedure Button4Click(Sender: TObject);
7. procedure Button5Click(Sender: TObject);
8. procedure Button7Click(Sender: TObject);
9. procedure Button6Click(Sender: TObject);
10. procedure Button8Click(Sender: TObject);
11. **Тестирование программы**

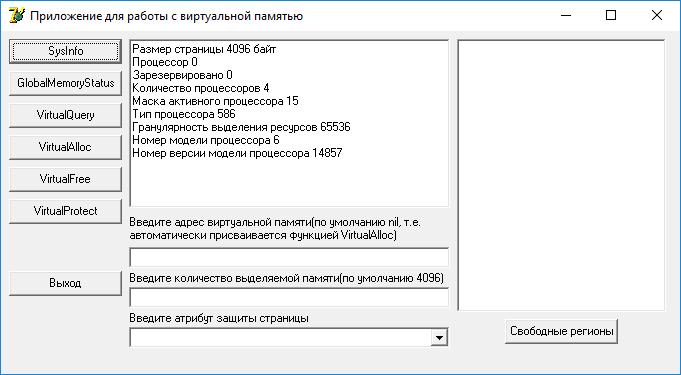


Рисунок 1 – Тестирование SysInfo

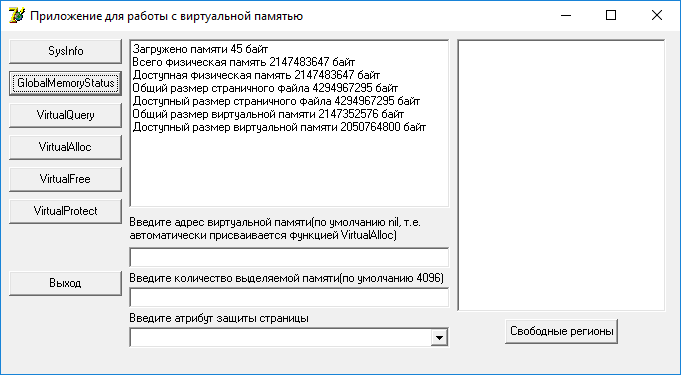


Рисунок 2 – Тестирование GlobalMemoryStatus

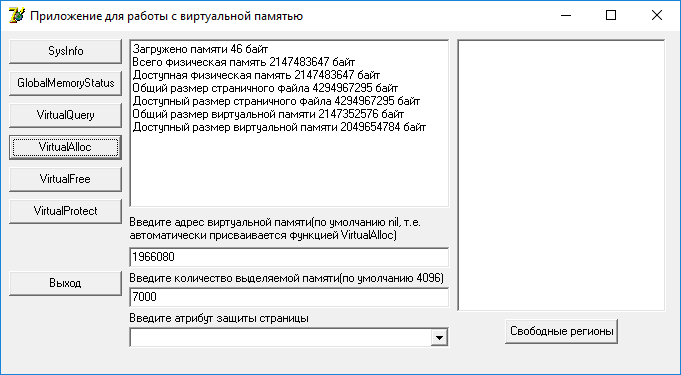


Рисунок 3 – Тестирование VirtualAlloc

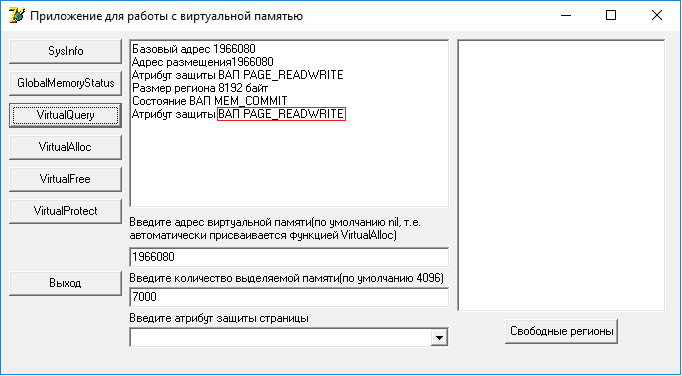


Рисунок 4 – Тестирование VirtualQuery

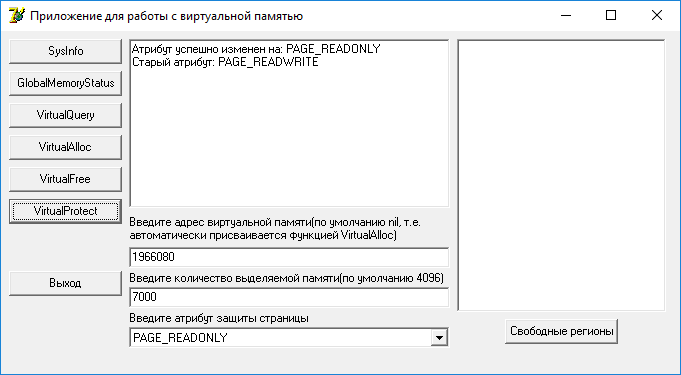


Рисунок 5 – Тестирование VirtualProtect

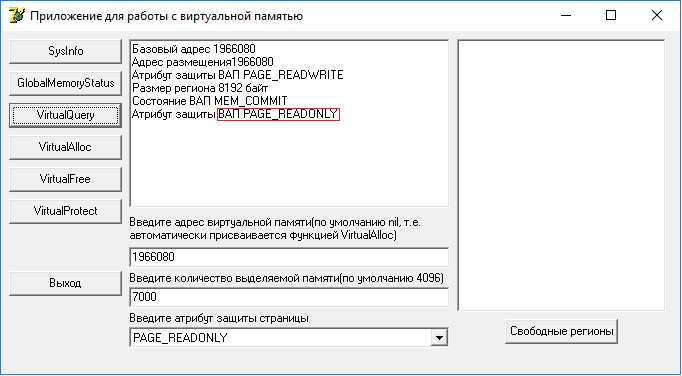


Рисунок 6 – Тестирование VirtualQueryпосле изменения атрибута защиты с помощью VirtualProtect

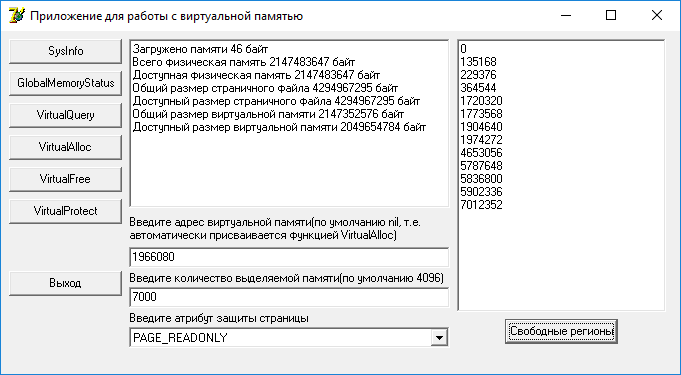


Рисунок 7 – Тестирование VirtualFreeи поиска свободных регионов

**Вывод:** Программа работает корректно и выдает ожидаемый результат.

1. **Контрольные вопросы**
2. Чему в настоящий момент равна гранулярность резервирования регионов адресного пространства для всех платформ Windows?

В настоящий момент гранулярность резервирования регионов адресного пространства равна 64Кб для всех платформ Windows.

1. Каким образом при выделении виртуальной памяти указать, что операционная система может выделить память в любой свободной области виртуального адресного пространства?

functionVirtualAlloc(lpvAddress: Pointer; dwSize, flAllocationType, flProtect: DWORD): Pointer; stdcall – резервирование региона в адресном пространстве. lpvAddress– адрес памяти, указывающий, где именно система должна зарезервировать адресное пространство. Если в качестве этого параметра указать nil, то операционная система сама выберет адрес резервируемого региона в адресном пространстве данного процесса.

1. Сколько байт виртуальной памяти будет выделено, если в функции VirtualAlloc в поле «Размер резервируемого региона в байтах» указать число 7000 байт?

Будет выделено: 8192 байта.

1. Объясните алгоритм преобразования виртуального адреса в физический?

Виртуальный адрес делится на номер виртуальной страницы и сдвиг. Номер страницы служит в качестве индекса в таблице страниц. По записи в таблице страниц находится номер физического блока, данный номер присоединяется к старшим разрядам смещения, заменяя, номер виртуальной страницы и формируется физический адрес.

1. Можно ли изменить атрибуты защиты сразу нескольких страниц виртуальной памяти?

Да, если страницы смежные, и они были зарезервированы одним вызовом VirtualAlloc.

1. **Заключение**

В результате выполнения лабораторной работы были изучены основные функции ядра Kernel.dll для работы с виртуальной памятью.

1. **Приложение А. Текст программы**

unit Unit1;

interface

uses

Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,

Dialogs, StdCtrls;

type

TForm1 = class(TForm)

Button1: TButton;

Button2: TButton;

Button3: TButton;

Button4: TButton;

Button5: TButton;

Button6: TButton;

Button7: TButton;

Label1: TLabel;

Edit1: TEdit;

memo1: TMemo;

Label2: TLabel;

Label3: TLabel;

Edit2: TEdit;

ComboBox1: TComboBox;

Memo2: TMemo;

Button8: TButton;

procedure Button1Click(Sender: TObject);

procedure FormActivate(Sender: TObject);

procedure Button2Click(Sender: TObject);

procedure Button3Click(Sender: TObject);

procedure Button4Click(Sender: TObject);

procedure Button5Click(Sender: TObject);

procedure Button7Click(Sender: TObject);

procedure Button6Click(Sender: TObject);

procedure Button8Click(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

Form1: TForm1;

p:pointer;

implementation

{$R \*.dfm}

procedure TForm1.Button1Click(Sender: TObject);

VAR

//SysInf:TSystemInfo;

SysInf:SYSTEM\_INFO;

begin

GetSystemInfo(SysInf);

Memo1.Clear;

memo1.Lines.Add('Размерстраницы '+IntToStr(SysInf.dwPageSize)+' байт');

memo1.Lines.Add('Процессор '+IntToStr(SysInf.wProcessorArchitecture));

memo1.Lines.Add('Зарезервировано '+IntToStr(SysInf.wReserved));

memo1.Lines.Add('Количествопроцессоров '+IntToStr(SysInf.dwNumberOfProcessors));

memo1.Lines.Add('Маскаактивногопроцессора '+IntToStr(SysInf.dwActiveProcessorMask));

memo1.Lines.Add('Типпроцессора '+IntToStr(SysInf.dwProcessorType));

memo1.Lines.Add('Гранулярностьвыделенияресурсов '+IntToStr(SysInf.dwAllocationGranularity));

memo1.Lines.Add('Номермоделипроцессора '+IntToStr(SysInf.wProcessorLevel));

memo1.Lines.Add('Номерверсиимоделипроцессора '+IntToStr(SysInf.wProcessorRevision));

end;

procedure TForm1.FormActivate(Sender: TObject);

begin

Memo1.Clear;

Memo2.Clear;

end;

procedure TForm1.Button2Click(Sender: TObject);

VAR //ms:MEMORYSTATUS;

ms:TMEMORYSTATUS;

begin

ms.dwLength:=sizeof(MEMORYSTATUS);

GlobalMemoryStatus(ms);

Memo1.Clear;

Memo1.Lines.Add('Загруженопамяти '+IntToStr(ms.dwMemoryLoad)+' байт');

Memo1.Lines.Add('Всегофизическаяпамять '+IntToStr(ms.dwTotalPhys)+' байт');

Memo1.Lines.Add('Доступнаяфизическаяпамять '+IntToStr(ms.dwAvailPhys)+' байт');

Memo1.Lines.Add('Общийразмерстраничногофайла '+IntToStr(ms.dwTotalPageFile)+' байт');

Memo1.Lines.Add('Доступныйразмерстраничногофайла '+IntToStr(ms.dwAvailPageFile)+' байт');

Memo1.Lines.Add('Общийразмервиртуальнойпамяти '+IntToStr(ms.dwTotalVirtual)+' байт');

Memo1.Lines.Add('Доступныйразмервиртуальнойпамяти '+IntToStr(ms.dwAvailVirtual)+' байт');

end;

procedure TForm1.Button3Click(Sender: TObject);

VAR //mbi:MEMORY\_BASIC\_INFORMATION;

mbi:TMemoryBasicInformation;

dwRez:DWORD;

begin

if Edit1.GetTextLen<=0

thenShowMessage('Введите виртуальный адрес')

else

begin

dwRez:=VirtualQuery(pointer(StrToInt(Edit1.Text)),mbi,sizeof(MEMORY\_BASIC\_INFORMATION));

Memo1.Clear;

Memo1.Lines.Add('Базовыйадрес '+IntToStr(integer(mbi.BaseAddress)));

Memo1.Lines.Add('Адресразмещения'+IntToStr(integer(mbi.BaseAddress)));

case mbi.AllocationProtect of

0: Memo1.Lines.Add('ВАПзарезервировано');

PAGE\_NOACCESS:Memo1.Lines.Add('АтрибутзащитыВАП PAGE\_NOACCESS');

PAGE\_READONLY:Memo1.Lines.Add('АтрибутзащитыВАП PAGE\_READONLY');

PAGE\_READWRITE:Memo1.Lines.Add('АтрибутзащитыВАП PAGE\_READWRITE');

PAGE\_EXECUTE:Memo1.Lines.Add('АтрибутзащитыВАП PAGE\_EXECUTE');

PAGE\_EXECUTE\_READ:Memo1.Lines.Add('АтрибутзащитыВАП PAGE\_EXECUTE\_READ');

PAGE\_EXECUTE\_READWRITE:Memo1.Lines.Add('АтрибутзащитыВАП PAGE\_EXECUTE\_READWRITE');

PAGE\_WRITECOPY:Memo1.Lines.Add('АтрибутзащитыВАП PAGE\_WRITECOPY');

PAGE\_EXECUTE\_WRITECOPY:Memo1.Lines.Add('АтрибутзащитыВАП PAGE\_EXECUTE\_WRITECOPY');

end;

Memo1.Lines.Add('Размеррегиона '+IntToStr(mbi.RegionSize)+' байт');

case mbi.State of

MEM\_COMMIT: Memo1.Lines.Add('СостояниеВАП MEM\_COMMIT');

MEM\_RESERVE: Memo1.Lines.Add('СостояниеВАП MEM\_RESERVE');

MEM\_FREE: Memo1.Lines.Add('СостояниеВАП MEM\_FREE');

end;

case mbi.Protect of

0: Memo1.Lines.Add('зарезервировано');

PAGE\_NOACCESS:Memo1.Lines.Add('Атрибутзащитыфиз.памяти PAGE\_NOACCESS');

PAGE\_READONLY:Memo1.Lines.Add('АтрибутзащитыВАП PAGE\_READONLY');

PAGE\_READWRITE:Memo1.Lines.Add('АтрибутзащитыВАП PAGE\_READWRITE');

PAGE\_EXECUTE:Memo1.Lines.Add('АтрибутзащитыВАП PAGE\_EXECUTE');

PAGE\_EXECUTE\_READ:Memo1.Lines.Add('АтрибутзащитыВАП PAGE\_EXECUTE\_READ');

PAGE\_EXECUTE\_READWRITE:Memo1.Lines.Add('АтрибутзащитыВАП PAGE\_EXECUTE\_READWRITE');

PAGE\_WRITECOPY:Memo1.Lines.Add('АтрибутзащитыВАП PAGE\_WRITECOPY');

PAGE\_EXECUTE\_WRITECOPY:Memo1.Lines.Add('АтрибутзащитыВАП PAGE\_EXECUTE\_WRITECOPY');

end;

end;

end;

procedure TForm1.Button4Click(Sender: TObject);

VAR ms:MEMORYSTATUS;

begin

If (Edit1.GetTextLen=0) and (Edit2.GetTextLen = 0) then

begin

p:=VirtualAlloc(nil,4096,MEM\_COMMIT,PAGE\_READWRITE)

end

else

begin

if Edit1.GetTextLen = 0 then

begin

p:=VirtualAlloc(nil,StrToInt(Edit2.Text),MEM\_COMMIT,PAGE\_READWRITE);

end

else

begin

if Edit2.GetTextLen = 0 then

begin

p:=VirtualAlloc(pointer(StrToInt64(Edit1.Text)),4096,MEM\_COMMIT,PAGE\_READWRITE);

end

else

begin

p:=VirtualAlloc(pointer(StrToInt64(Edit1.Text)),StrToInt(Edit2.Text),MEM\_COMMIT,PAGE\_READWRITE);

end;

end;

end;

Edit1.Text:=IntToStr(Integer(p));

Memo1.Clear;

ms.dwLength:=sizeof(MEMORYSTATUS);

GlobalMemoryStatus(ms);

Memo1.Clear;

Memo1.Lines.Add('Загруженопамяти '+IntToStr(ms.dwMemoryLoad)+' байт');

Memo1.Lines.Add('Всегофизическаяпамять '+IntToStr(ms.dwTotalPhys)+' байт');

Memo1.Lines.Add('Доступнаяфизическаяпамять '+IntToStr(ms.dwAvailPhys)+' байт');

Memo1.Lines.Add('Общийразмерстраничногофайла '+IntToStr(ms.dwTotalPageFile)+' байт');

Memo1.Lines.Add('Доступныйразмерстраничногофайла '+IntToStr(ms.dwAvailPageFile)+' байт');

Memo1.Lines.Add('Общийразмервиртуальнойпамяти '+IntToStr(ms.dwTotalVirtual)+' байт');

Memo1.Lines.Add('Доступныйразмервиртуальнойпамяти '+IntToStr(ms.dwAvailVirtual)+' байт');

end;

procedure TForm1.Button5Click(Sender: TObject);

VAR

ms:MEMORYSTATUS;

begin

if Edit2.GetTextLen = 0 then

begin

VirtualFree(p,4096,MEM\_DECOMMIT);

VirtualFree(p,4096,MEM\_RELEASE);

end

else

begin

VirtualFree(p,StrToInt(Edit2.Text),MEM\_DECOMMIT);

VirtualFree(p,StrToInt(Edit2.Text),MEM\_RELEASE);

end;

Memo1.Clear;

ms.dwLength:=sizeof(MEMORYSTATUS);

GlobalMemoryStatus(ms);

Memo1.Clear;

Memo1.Lines.Add('Загруженопамяти '+IntToStr(ms.dwMemoryLoad)+' байт');

Memo1.Lines.Add('Всегофизическаяпамять '+IntToStr(ms.dwTotalPhys)+' байт');

Memo1.Lines.Add('Доступнаяфизическаяпамять '+IntToStr(ms.dwAvailPhys)+' байт');

Memo1.Lines.Add('Общийразмерстраничногофайла '+IntToStr(ms.dwTotalPageFile)+' байт');

Memo1.Lines.Add('Доступныйразмерстраничногофайла '+IntToStr(ms.dwAvailPageFile)+' байт');

Memo1.Lines.Add('Общийразмервиртуальнойпамяти '+IntToStr(ms.dwTotalVirtual)+' байт');

Memo1.Lines.Add('Доступныйразмервиртуальнойпамяти '+IntToStr(ms.dwAvailVirtual)+' байт');

end;

procedure TForm1.Button7Click(Sender: TObject);

begin

Form1.Close;

end;

procedure TForm1.Button6Click(Sender: TObject);

var

lpflOldProtect: DWORD;

dwSize: DWORD;

success: boolean;

begin

if ComboBox1.Text = '' then

ShowMessage('Выберите атрибут защиты страницы')

else

begin

if Edit2.GetTextLen = 0 then

begin

dwSize:= 4096;

end

else

begin

dwSize:=StrToInt(Edit2.Text);

end;

case ComboBox1.ItemIndex of

0: success:=VirtualProtect(pointer(StrToInt64(Edit1.Text)),dwSize,PAGE\_NOACCESS,lpflOldProtect);

1: success:=VirtualProtect(pointer(StrToInt64(Edit1.Text)),dwSize,PAGE\_READONLY,lpflOldProtect);

2: success:=VirtualProtect(pointer(StrToInt64(Edit1.Text)),dwSize,PAGE\_READWRITE,lpflOldProtect);

3: success:=VirtualProtect(pointer(StrToInt64(Edit1.Text)),dwSize,PAGE\_EXECUTE,lpflOldProtect);

4: success:=VirtualProtect(pointer(StrToInt64(Edit1.Text)),dwSize,PAGE\_EXECUTE\_READ,lpflOldProtect);

5: success:=VirtualProtect(pointer(StrToInt64(Edit1.Text)),dwSize,PAGE\_EXECUTE\_READWRITE,lpflOldProtect);

6: success:=VirtualProtect(pointer(StrToInt64(Edit1.Text)),dwSize,PAGE\_WRITECOPY,lpflOldProtect);

7: success:=VirtualProtect(pointer(StrToInt64(Edit1.Text)),dwSize,PAGE\_EXECUTE\_WRITECOPY,lpflOldProtect);

end;

Memo1.Clear;

if success then

begin

Memo1.Lines.Add('Атрибутуспешноизмененна: ' + ComboBox1.Text);

case lpflOldProtect of

PAGE\_NOACCESS:Memo1.Lines.Add('Старыйатрибут: PAGE\_NOACCESS');

PAGE\_READONLY:Memo1.Lines.Add('Старыйатрибут: PAGE\_READONLY');

PAGE\_READWRITE:Memo1.Lines.Add('Старыйатрибут: PAGE\_READWRITE');

PAGE\_EXECUTE:Memo1.Lines.Add('Старыйатрибут: PAGE\_EXECUTE');

PAGE\_EXECUTE\_READ:Memo1.Lines.Add('Старыйатрибут: PAGE\_EXECUTE\_READ');

PAGE\_EXECUTE\_READWRITE:Memo1.Lines.Add('Старыйатрибут: PAGE\_EXECUTE\_READWRITE');

PAGE\_WRITECOPY:Memo1.Lines.Add('Старыйатрибут: PAGE\_WRITECOPY');

PAGE\_EXECUTE\_WRITECOPY:Memo1.Lines.Add('Старыйатрибут: PAGE\_EXECUTE\_WRITECOPY');

end;

end

else

begin

Memo1.Lines.Add('Неудалосьизменитьатрибут');

end;

end;

end;

procedure TForm1.Button8Click(Sender: TObject);

var

mbi:TMemoryBasicInformation;

dwRez, i:DWORD;

begin

memo2.Clear;

i:=00000000;

while i < 11111111 do

begin

dwRez:=VirtualQuery(pointer(i),mbi,sizeof(MEMORY\_BASIC\_INFORMATION));

if mbi.State = MEM\_FREE then Memo2.Lines.Add(IntToStr(i));

Inc(i,mbi.RegionSize);

end;

end;

end.