МИНИСТЕРСТВО ВЫСШЕГО ОБРАЗОВАНИЯ И НАУКИ РФ

филиал федерального государственного бюджетного образовательного учреждения высшего образования «Национальный исследовательский университет «МЭИ» в г. Смоленске

Кафедра вычислительной техники

Отчет по лабораторной работе №6

по курсу: «Операционные системы»

Тема: «РАБОТА С ФУНКЦИЯМИ TOOLHELP»

Группа: ИВТ1-18

Студент: Федченков М. В.

Преподаватель: Гаврилов А. И.

Вариант: № 15

Смоленск, 2019 г.

1. **Цель работы**

Целью работы является получение навыков работы с функциями библиотеки ToolHelp API для получения системной информации. В данной работе рассматриваются следующие вопросы:

* Получение списка всех процессов в системе;
* Получение списка всех модулей в системе;
* Получение списка всех потоков выбранного процесса;
* Получение карты памяти выбранного процесса.

1. **Задание на лабораторную работу**

Программа, формирующая список трех процессов, выполняющихся на данном компьютере и имеющих самое большое количество потоков (в порядке убывания количества потоков) и идентификаторы всех потоков для каждого процесса.

1. **Анализ задания на лабораторную работу**

Написать программу, которая будет выводить сначала список всех процессов в системе, потом по нажатию кнопки, будет выдавать список из трех процессов по порядке убыванию по количеству потоков. По нажатию на название процесса, в другую таблицу будут выводиться потоки данного процесса с их идентификаторами.

1. **Модульная структура программы**

pe32:tProcessEntry32;

te32:tThreadEntry32;

me32:tModuleEntry32;

hProcess:THandle;

hSnapshot: THandle;

1. **Алгоритм**

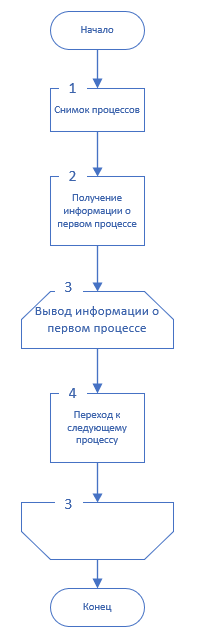


Рисунок 1 – Алгоритм функции TForm1.FormCreate

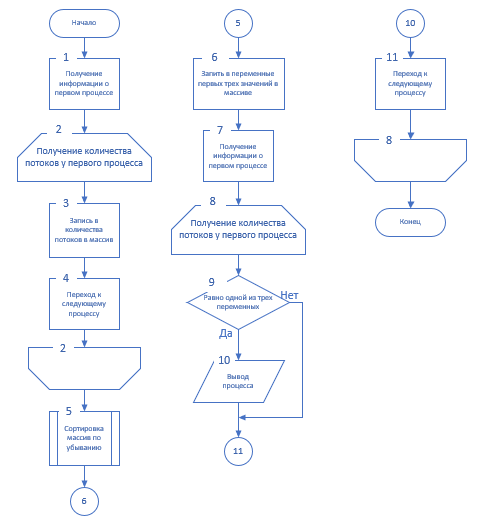


Рисунок 2 – Алгоритм функции TForm1.Button2Click

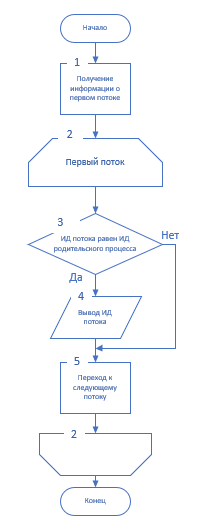


Рисунок 3 – Алгоритм функции TForm1.StringGrid1Click

1. **Результат работы**

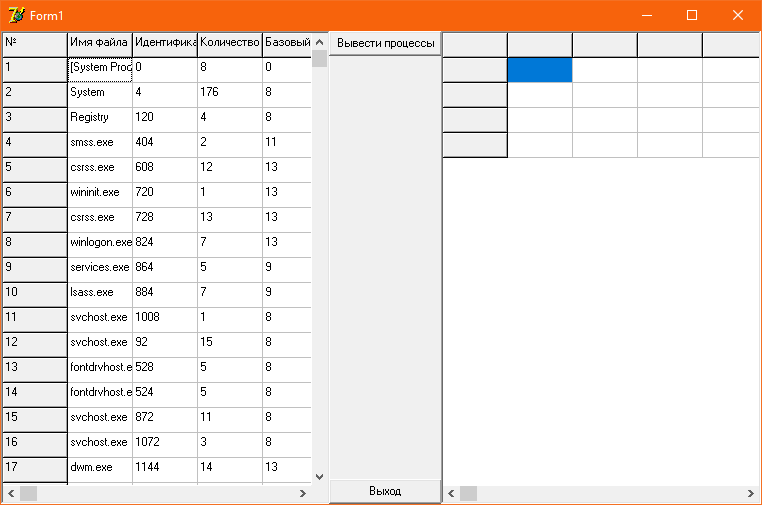


Рисунок 4 – При запуске программы

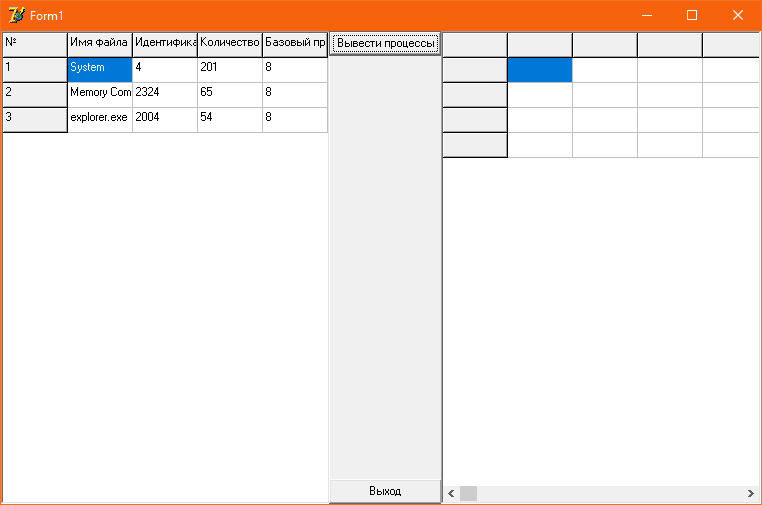


Рисунок 5 – Вывод трех процессов

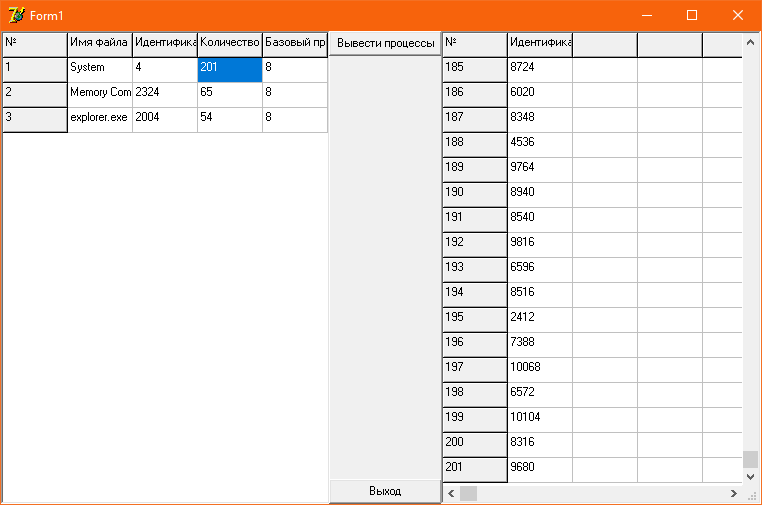


Рисунок 6 – Вывод потоков процесса

1. **Контрольные вопросы**
2. Как получить информацию о всех загруженных процессах?

Для просмотра списка загруженных процессов определены следующие функции:

* + Process32First - получение информации о первом загруженном процессе;
  + Process32Next - получение информации об очередном загруженном процессе;

Подробно эти функции описаны в соответствующем пункте теоретического введения.

1. Как получить карту памяти любого процесса?

Для получения карты памяти любого процесса можно использовать следующий алгоритм:

* С помощью функций Process32First и Process32Next получить список всех процессов в системе, запоминая их идентификаторы;
* Выбрать из списка интересующий вас процесс;
* Зная идентификатор процесса получить дескриптор процесса с помощью функции Open Process;
* Зная дескриптор процесса, можно получить карту памяти, используя функцию VirtualQueryEx.

1. Как получить список потоков выбранного процесса?

* С помощью функций Process32First и Process32Next получить список всех процессов в системе, запоминая их идентификаторы.
* Выбрать из списка интересующий вас процесс и запомнить его идентификатор.
* С помощью функций Thread32First и Thread32Next получить список всех потоков в системе.
* Отобрать потоки, советующие идентификатору выбранного процесса.

1. Как получить список всех блоков выбранной кучи?

Это можно сделать с помощью функций Heap32First и Heap32Next.

1. Как получить список модулей, принадлежащих конкретному процессу?

* С помощью функций Process32First и Process32Next получить список всех процессов в системе, запоминая их идентификаторы.
* Выбрать из списка интересующий вас процесс и запомнить его идентификатор.
* С помощью функций Module32First и Module32Next получить список всех модулей.
* Отобрать модули, советующие идентификатору выбранного процесса.

1. Как получить список всех модулей системы?

Это можно сделать с помощью функций Module32First и Module32Next.

**Приложение**

**А. Текст программы**

unit Unit1;

interface

uses

Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms, Dialogs, Grids, StdCtrls, Buttons,tlhelp32;

type

TForm1 = class(TForm)

StringGrid1: TStringGrid;

Button1: TButton;

Button2: TButton;

StringGrid2: TStringGrid;

procedure FormCreate(Sender: TObject);

procedure Button1Click(Sender: TObject);

procedure Button2Click(Sender: TObject);

procedure StringGrid1Click(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

Form1: TForm1;

pe32:tProcessEntry32;

te32:tThreadEntry32;

me32:tModuleEntry32;

hProcess:THandle;

hSnapshot: THandle;

implementation

{$R \*.dfm}

procedure TForm1.FormCreate(Sender: TObject);

VAR

x,code:Integer;

s:String;

begin

StringGrid1.Cells[0,0]:='№';

StringGrid1.Cells[1,0]:='Имя файла';

StringGrid1.Cells[2,0]:='Идентификатор процесса';

StringGrid1.Cells[3,0]:='Количество потоков';

StringGrid1.Cells[4,0]:='Базовый приоритет';

pe32.dwSize:=SizeOf(tProcessEntry32);

x:=1;

hProcess:=CreateToolhelp32SnapShot(TH32CS\_SNAPPROCESS,0);

te32.dwSize:=SizeOf(THREADENTRY32);

hSnapshot:=CreateToolhelp32SnapShot(TH32CS\_SNAPTHREAD,0);

Process32First(hProcess,pe32);

{ str(pe32.th32ProcessID,StringGrid1.Cells[2,x]);

val(StringGrid1.Cells[2,x],pe32.th32ProcessID,code);}

str(pe32.th32ProcessID,s);

val(s,pe32.th32ProcessID,code);

Repeat

StringGrid1.Cells[0,x]:=IntToStr(x);

StringGrid1.Cells[1,x]:=pe32.szExeFile;

StringGrid1.Cells[2,x]:=IntToStr(pe32.th32ProcessID);

StringGrid1.Cells[3,x]:=IntToStr(pe32.cntThreads);

StringGrid1.Cells[4,x]:=IntToStr(pe32.pcpriClassBase);

x:=x+1;

StringGrid1.RowCount:=x;

until Process32Next(hProcess,pe32)=False;

end;

procedure TForm1.Button1Click(Sender: TObject);

begin

Form1.Close;

end;

procedure BubbleSort(var Vetor: array of Integer);

var

i, temp: Integer;

changed: Boolean;

begin

changed := True;

while changed do

begin

changed := False;

for i := Low(Vetor) to High(Vetor) - 1 do

begin

if (Vetor[i] < Vetor[i + 1]) then

begin

temp := Vetor[i + 1];

Vetor[i + 1] := Vetor[i];

Vetor[i] := temp;

changed := True;

end;

end;

end;

end;

procedure TForm1.Button2Click(Sender: TObject);

var ThreadCount: array [1..150] of integer;

x:integer;

check:bool;

k1,k2,k3:integer;

n,i,j:integer;

s:string;

begin

//Запись в массив количества потоков в процессах

Process32First(hProcess,pe32);

x:=1;

Repeat

ThreadCount[x]:=pe32.cntThreads;

x:=x+1;

until Process32Next(hProcess,pe32)=False;

//Сортировка по убыванию

BubbleSort(ThreadCount);

//

k1:=0; k2:=0; k3:=0;

x:=1;

check:=true;

while check do

begin

if ThreadCount[x]<1000 then

begin

k1:=ThreadCount[x];

k2:=ThreadCount[x+1];

k3:=ThreadCount[x+2];

check:=false;

end

else x:=x+1;

end;

//

{Edit1.Text:=IntToStr(k1) + ' ' + IntToStr(k2) + ' '+ IntToStr(k3); }

//

x:=1;

Process32First(hProcess,pe32);

Repeat

if ((pe32.cntThreads=k1) or (pe32.cntThreads=k2) or (pe32.cntThreads=k3)) then

begin

StringGrid1.Cells[0,x]:=IntToStr(x);

StringGrid1.Cells[1,x]:=pe32.szExeFile;

StringGrid1.Cells[2,x]:=IntToStr(pe32.th32ProcessID);

StringGrid1.Cells[3,x]:=IntToStr(pe32.cntThreads);

StringGrid1.Cells[4,x]:=IntToStr(pe32.pcpriClassBase);

x:=x+1;

StringGrid1.RowCount:=x;

end;

until Process32Next(hProcess,pe32)=False;

s:=StringGrid1.Cells[3,3];

StringGrid1.Cells[3,3]:=StringGrid1.Cells[3,2];

StringGrid1.Cells[3,2]:=s;

end;

procedure TForm1.StringGrid1Click(Sender: TObject);

var Id:integer;

x:integer;

begin

StringGrid2.Cells[0,0]:='№';

StringGrid2.Cells[1,0]:='Идентификатор';

x:=1;

Id:=StrToInt64(StringGrid1.Cells[2, StringGrid1.Row]);

Thread32First(hSnapshot ,te32);

Repeat

begin

if Id=te32.th32OwnerProcessID then

begin

StringGrid2.Cells[0,x]:=IntToStr(x);

StringGrid2.Cells[1,x]:=IntToStr(te32.th32ThreadID);

x:=x+1;

StringGrid2.RowCount:=x;

end;

end;

until Thread32Next(hSnapshot,te32)=false;

end;

end.