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 №5

по курсу: «Операционные системы»

Тема: «СОЗДАНИЕ ПРОЦЕССОВ И ПОТОКОВ»

Группа: ИВТ1-18

Студент: Федченков М. В.

Преподаватель: Гаврилов А. И.

Вариант: № 15

Смоленск, 2019 г.

1. **Цель работы**

Целью работы является изучение основных принципов организации многозадачных операционных систем. Все многозадачные операционные системы используют концепцию процесса и потока. В данной работе рассматриваются следующие вопросы:

Чередование выполнения нескольких процессов с целью повышения степени использования процессора;

Разделение ресурсов между процессами;

Организация обмена данными между процессами и потоками;

Изменение класса приоритета процесса и уровня приоритета потока.

1. **Задание на лабораторную работу**

Создание нескольких приложений. Одно из них рисует окружности, другое эллипсы, третье прямоугольники. Четвертое приложение запускает три предыдущих и изменяет приоритет любого процесса.

1. **Анализ задания на лабораторную работу**

Написать несколько приложений по отдельности: первое будет рисовать окружность, второе эллипс, треть прямоугольник. Далее написать четвертое приложения, которое будет запускать все три и изменяет приоритет любого.

1. **Модульная структура программы**

TForm1 = class(TForm)

Button1: TButton;

Button2: TButton;

Button3: TButton;

OpenDialog1: TOpenDialog;

Button4: TButton;

Button5: TButton;

procedure Button1Click(Sender: TObject);

procedure Button2Click(Sender: TObject);

procedure Button3Click(Sender: TObject);

procedure Button4Click(Sender: TObject);

procedure Button5Click(Sender: TObject);

1. **Алгоритм**

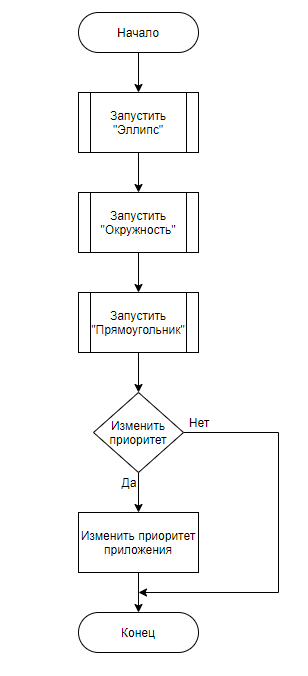


Рисунок 1 – Алгоритм программы

1. **Результат работы**

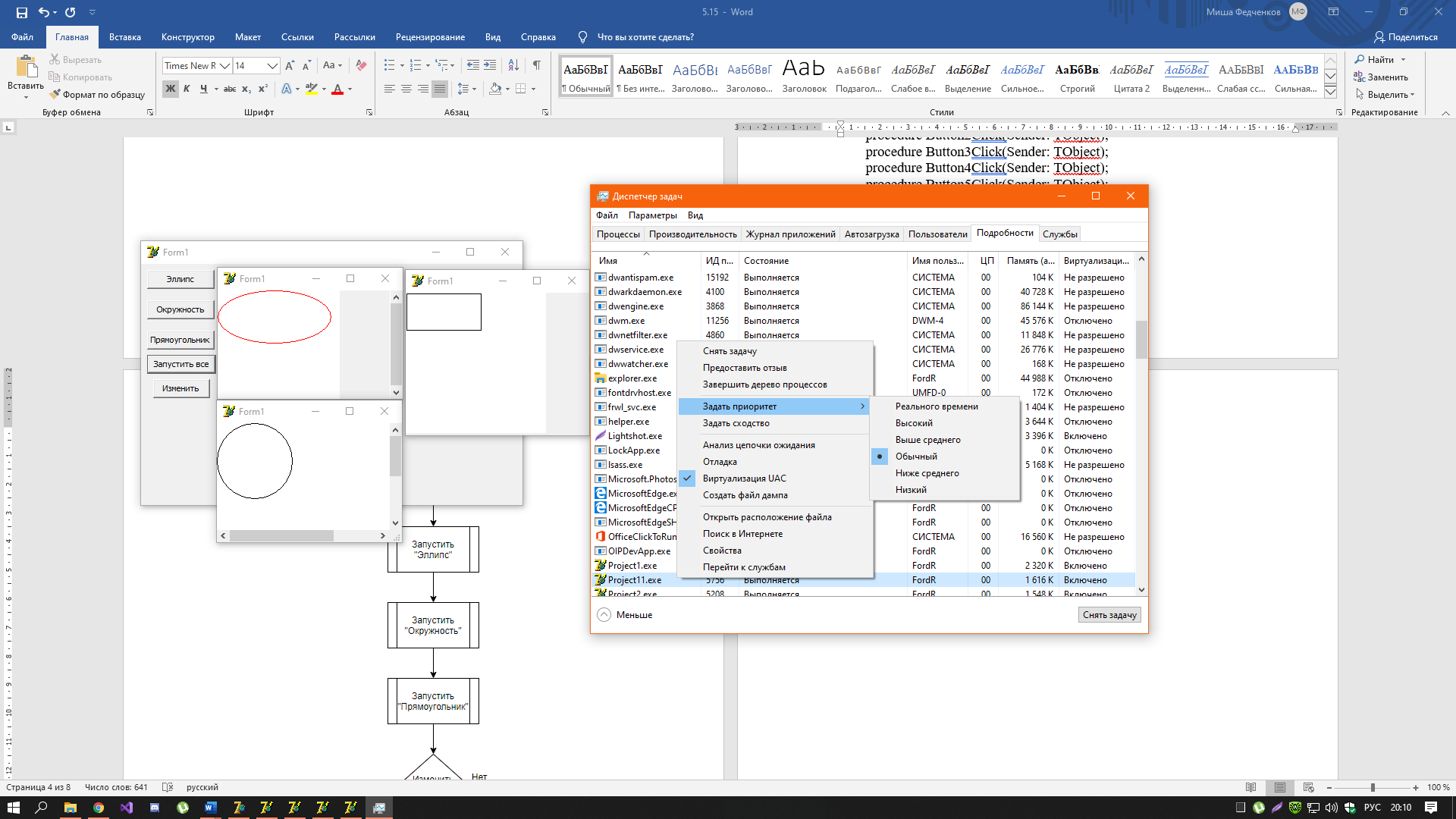


Рисунок 2 – Результат при запуске

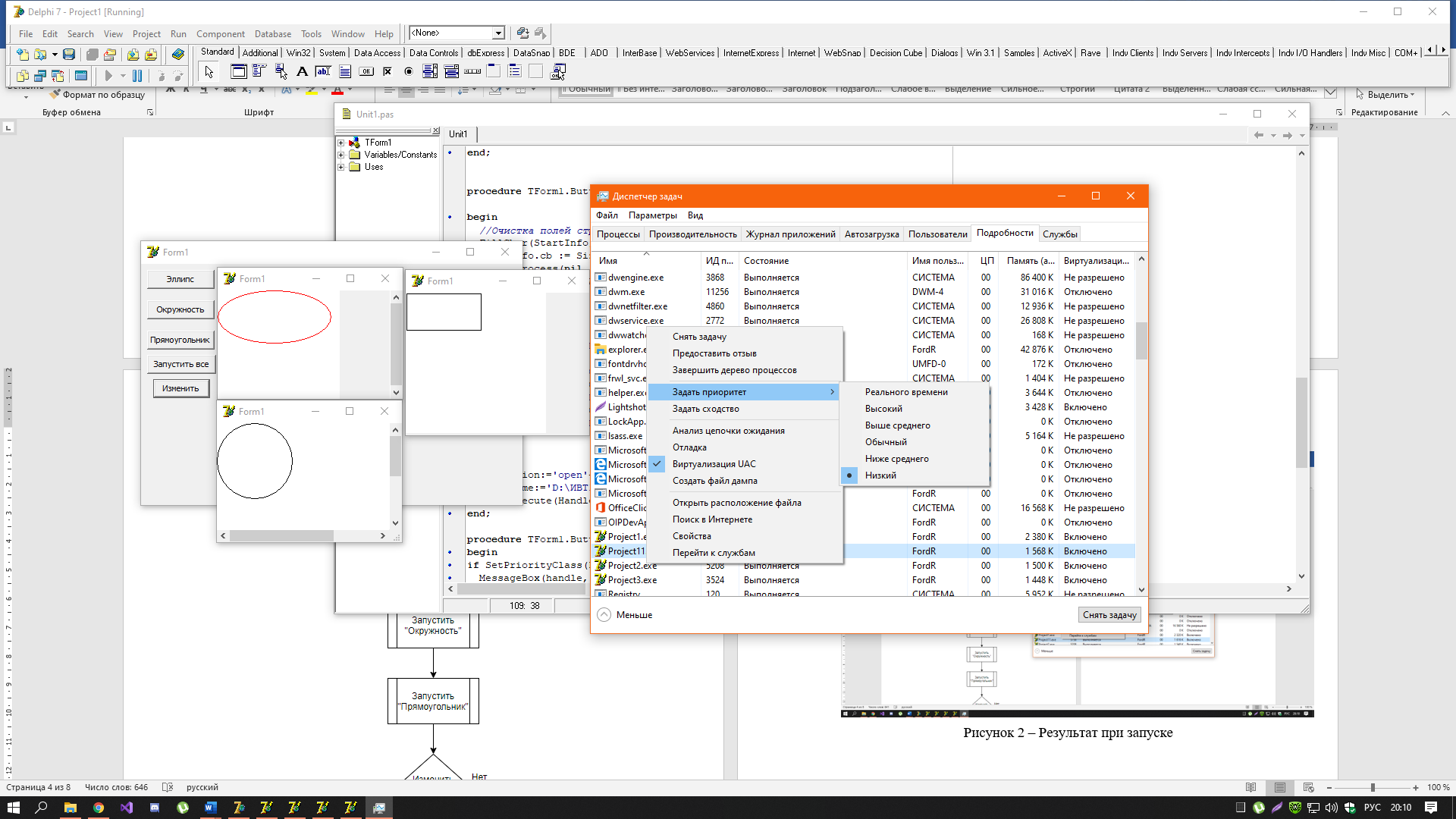


Рисунок 3 – Результат после изменения приоритета

1. **Контрольные вопросы**
2. Сравните возможности функций CreateProcess? WinExec и ShellExecute.

Для создания процесса используется функция CreateProcess. Она позволяет иметь возможность задавать определенные параметры запуска (например, базовый приоритет, наследование дескрипторов, переменные окружения и т.п.).

Самый простой способ запустить какую-то программу из своего приложе­ния — использовать функцию **WinExec.** Эта функция может работать в любых версиях Windows и выполнять любые файлы: приложения Windows, MS-DOS, файлы PIF и т.п.Достоинством функции WinExecявляется ее совместимость с ранними версиями Windows. Для этого она и сохраняется в Win32, хотя для Win32 рекомендуется пользоваться функцией CreateProcess.

Функция ShellExecuteможет не только выполнять заданное приложение, но и открывать документ и печатать его. Под термином «открыть файл документа» понимается выполнение связанного с ним приложения и загрузка в него этого до­кумента. Например, обычно с документами, имеющими расширение .doc, связан Word. В этом случае открыть файл, например, с именем file.docозначает запус­тить Word и передать ему в качестве параметра имя файла file.doc. Кроме описан­ных возможностей функция ShellExecuteпозволяет открыть указанную папку. Это означает, что будет запущена программа «Проводник» с открытой указанной папкой.

1. В чем заключается различие между процедурой ExitProcessи функцией TerminateProcess?

Если функция TerminateProcess успешно выполнена, она возвращает ненулевое значение (true). Если произошла ошибка – возвращается 0 (false).

Главное отличие функции TerminateProcess от функции ExitProcess в том, что TerminateProcess может вызвать любой поток, любого процесса для завершения другого процесса.

1. Почему ExitProcess и ExitThread реализованы как процедуры, TerminateProcess и TerminateThread – как функции?

ExitProcess и ExitTread завершают текущий поток и процесс соответственно, которые всегда можно завершить. Поскольку TerminateProcessи TerminateThread могут завершать любой поток и процесс соответственно, имеет смысл возвращать результат завершение, так как может возникнуть ошибка.

1. Как установит приоритет текущего процесса равным 31?

Установить класс приоритета процесса Real-timeс помощью функции SetPriorityClass. Затем установить уровень приоритета потока Time-criticalс помощью функции SetThreadPriority.

1. Как установит приоритет текущего процесса равным 1?

Установить класс приоритета процессаIdleс помощью функции SetPriorityClass.Затем установить уровень приоритета потокаIdleс помощью функции SetThreadPriority.

1. Величина кванта потока - 6 квантовых единиц. Компьютер имеет несколько процессоров. Определите величину кванта в миллисекундах.

90 миллисекунд.

1. Величина кванта потока - 18 квантовых единиц. Компьютер имеет один процессор. Определите величину кванта в миллисекундах.

180 миллисекунд.

1. Вы начали длительный пересчет электронной таблицы и переключились на приложение, активно использующее процессор (например, игра типа “action”). Что сделает ОС для эффективного использования процессорного времени – увеличит приоритет активного процесса (игры) или увеличит квант времени активного процесса. Ответ обоснуйте.

Всякий раз, когда поток в активном процессе завершает ожидание на объекте ядра, функция ядра KiUnwaitThread динамически повышает его текущий (не базовый) приоритет на величину текущего значения PsPrioritySeparation. (Какой процесс является активным, определяет подсистема управления окнами.)

PsPrioritySeparation представляет собой индекс в таблице квантов (байтовом массиве), с помощью которой выбираются величины квантов для потоков активных процессов. Однако в данном случае PsPrioritySeparation используется как значение, на которое повышается приоритет.

Это увеличивает отзывчивость интерактивного приложения по окончании ожидания. B результате повышаются шансы на немедленное возобновление его потока — особенно если в фоновом режиме выполняется несколько процессов с тем же базовым приоритетом.

1. Поток ожидал завершения операции ввода-вывода. Текущий приоритет потока – 12. По завершении операции ввода-вывода приоритет потока должен быть увеличен на 4 уровня. Каким будет приоритет потока после завершения операции ввода-вывода?

Приоритет потока будет равняться 15.

Система повышает приоритет только тех потоков, базовый уровень которых находится в пределах 1-15. Именно поэтому данный диапазон называется "областью динамического приоритета" (dynamicpriorityrange). Система не допускает динамического повышения приоритета потока до уровней реального времени (более 15). Поскольку потоки с такими уровнями обслуживают системные функции, это ограничение не дает приложению нарушить работу ОС.

**Приложение**

**А. Текст программы**

unit Unit1;

interface

uses

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

Dialogs, StdCtrls, ShellAPI;

type

TForm1 = class(TForm)

Button1: TButton;

Button2: TButton;

Button3: TButton;

OpenDialog1: TOpenDialog;

Button4: TButton;

Button5: TButton;

procedure Button1Click(Sender: TObject);

procedure Button2Click(Sender: TObject);

procedure Button3Click(Sender: TObject);

procedure Button4Click(Sender: TObject);

procedure Button5Click(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

Form1: TForm1;

StartInfo: TStartUpInfo;

ProcInfo: TProcessInformation;

ErrCode:Integer;

PName:Array [0..79] of char;

POperation:Array[0..50] of Char;

PFilename:Array[0..254] of Char;

implementation

{$R \*.dfm}

procedure TForm1.Button1Click(Sender: TObject);

var

StartInfo: TStartUpInfo;

ProcInfo: TProcessInformation;

begin

//Очистка полей структуры StartInfo

FillChar(StartInfo, Sizeof(StartInfo), #0);

StartInfo.cb := Sizeof(StartInfo);

CreateProcess(nil, 'C:\Users\страж\Desktop\итв1-18.15\5.1\project11.exe', nil,

nil, False, NORMAL\_PRIORITY\_CLASS, nil, nil, StartInfo, ProcInfo)

end;

procedure TForm1.Button2Click(Sender: TObject);

VAR ErrCode:Integer;

PName:Array [0..79] of char;

begin

If OpenDialog1.Execute then

begin

ErrCode:=WinExec(StrPCopy(PName,OpenDialog1.FileName),SW\_RESTORE);

PName:=PName;

Case ErrCode of

0: ShowMessage('Не хватает памяти или ресурсов');

ERROR\_BAD\_FORMAT:

ShowMessage('Ошибочный файл '+PName);

ERROR\_FILE\_NOT\_FOUND:

ShowMessage('Не найден файл '+PName);

ERROR\_PATH\_NOT\_FOUND:

ShowMessage('Не найден каталог '+ExtractFilePath(OpenDialog1.FileName));

end;

end;

end;

procedure TForm1.Button3Click(Sender: TObject);

VAR

POperation:Array[0..50] of Char;

PFilename:Array[0..254] of Char;

begin

POperation:='open'+#0;

PFilename:='C:\Users\страж\Desktop\итв1-18.15\5.3\project3.exe'+#0;

ShellExecute(Handle,POperation,PFilename,nil,nil,SW\_RESTORE);

end;

procedure TForm1.Button4Click(Sender: TObject);

begin

//Очистка полей структуры StartInfo

FillChar(StartInfo, Sizeof(StartInfo), #0);

StartInfo.cb := Sizeof(StartInfo);

CreateProcess(nil, 'C:\Users\страж\Desktop\итв1-18.15\5.1\project11.exe', nil,

nil, False, NORMAL\_PRIORITY\_CLASS, nil, nil, StartInfo, ProcInfo);

ErrCode:=WinExec('C:\Users\страж\Desktop\итв1-18.15\5.2\project2.exe',SW\_RESTORE);

PName:=PName;

Case ErrCode of

0: ShowMessage('Не хватает памяти или ресурсов');

ERROR\_BAD\_FORMAT:

ShowMessage('Ошибочный файл '+PName);

ERROR\_FILE\_NOT\_FOUND:

ShowMessage('Не найден файл '+PName);

ERROR\_PATH\_NOT\_FOUND:

ShowMessage('Не найден каталог '+ExtractFilePath(OpenDialog1.FileName));

end;

POperation:='open'+#0;

PFilename:='C:\Users\страж\Desktop\итв1-18.15\5.3\project3.exe'+#0;

ShellExecute(Handle,POperation,PFilename,nil,nil,SW\_RESTORE);

end;

procedure TForm1.Button5Click(Sender: TObject);

begin

if SetPriorityClass(ProcInfo.hProcess,IDLE\_PRIORITY\_CLASS) then

MessageBox(handle,'Выполнено','Done',0)

else MessageBox(handle,'Ошибка','NotDone',0)

end;

end.