# Разработка виртуальной системы исполнения программ

**Цель работы**

Разработка стекового виртуального процессора для исполнения кода и системы визуализации, основанной на оконной системе Windows

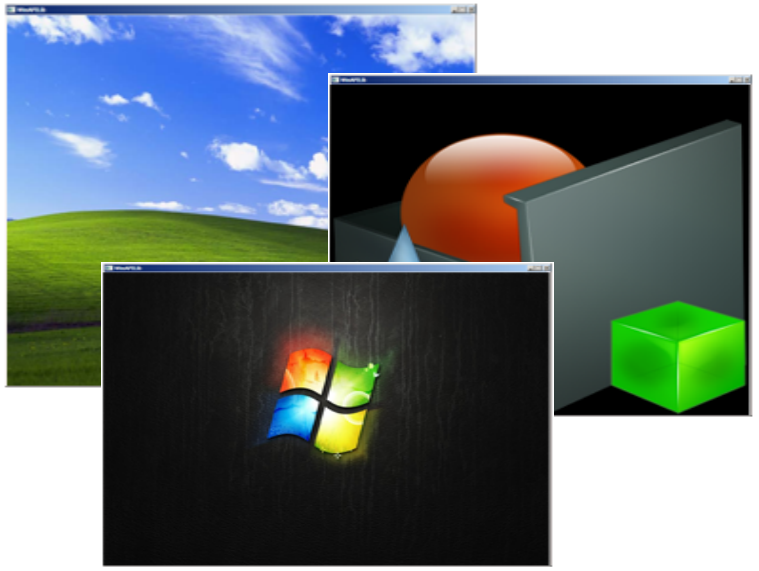
Автор: Делла Пиетра Раффаэль, 8 “Б” класс

Научный руководитель: Дединский Илья Рудольфович

# Задачи работы

- Разработка двух модулей:
  - Система исполнения
    - Обработчик входных данных
    - Виртуальный процессор
    - Хранение данных
  - Система визуализации
    - Оконная система на базе Win32
    - Система слоев
    - Эмулятор подмножества языка Logo

```
link “logo.lib”
start
1:
sel abx
addr 1
cmpr 5
je L2
fw 100
rt 90
…
```



# Архитектура программы

**Система исполнения**

- Обработка входного файла
- Сбор данных для запуска виртуального процессора

**Виртуальный процессор**

- Исполнение кода
- Хранение данных

**Оконная система**

- Работа с окнами
- Взаимодействие с Win32

**Эмулятор Logo**

- Рисование “черепашки”
- Интерфейс для написания скриптов

# Этапы работы системы исполнения

**Первый проход**

- Преобразование команд в байт-код
- Частичная обработка параметров
- Сохранение обработанного файла

| Input data | Parsed data |
|---|---|
| var %data | 405 %data |
| input_ %data | 317 %data |
| sel adx | 300 R12 |
| set 2 | 301 2 |
| mulr %data | 303 %data |
| push adx | 100 R12 |
| call _MyFunc | 309 _MyFunc |

**Второй проход**

- Полный разбор параметров
- Регистрация переменных, функций и меток
- Распределение команд и параметров в вектора

| Parsed data | Virtual processor |
|---|---|
| 405 %data | new var %data |
| 317 %data | input -> %data |
| 300 R12 | selected reg adx |
| 301 2 | adx = 2 |
| 303 %data | adx *= %data |
| 100 R12 | push adx in stack |
| 309 _MyFunc | calling _MyFunc |

**Виртуальный процессор**

- Потоковое исполнение команд
- Работа с регистрами и стеком

# Хранение данных

**Стек**

- Набор команд аналогичен командам x86
- Рекурсивный механизм исключений

**Регистры**

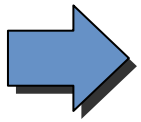
- Быстрый доступ к памяти
- 1, 2, 4 и 8 байт
- Реализован через std::map, хранящий смещения от начала массива регистров

**Переменные (ОЗУ)**

- Реализованы через таблицу имен
- Для ускорения доступа при исполнении производится прямая адресация к памяти

| name | pointer |
|---|---|
| %jump | 0x4C2928 |
| %len | 0x6E3194 |
| … | … |

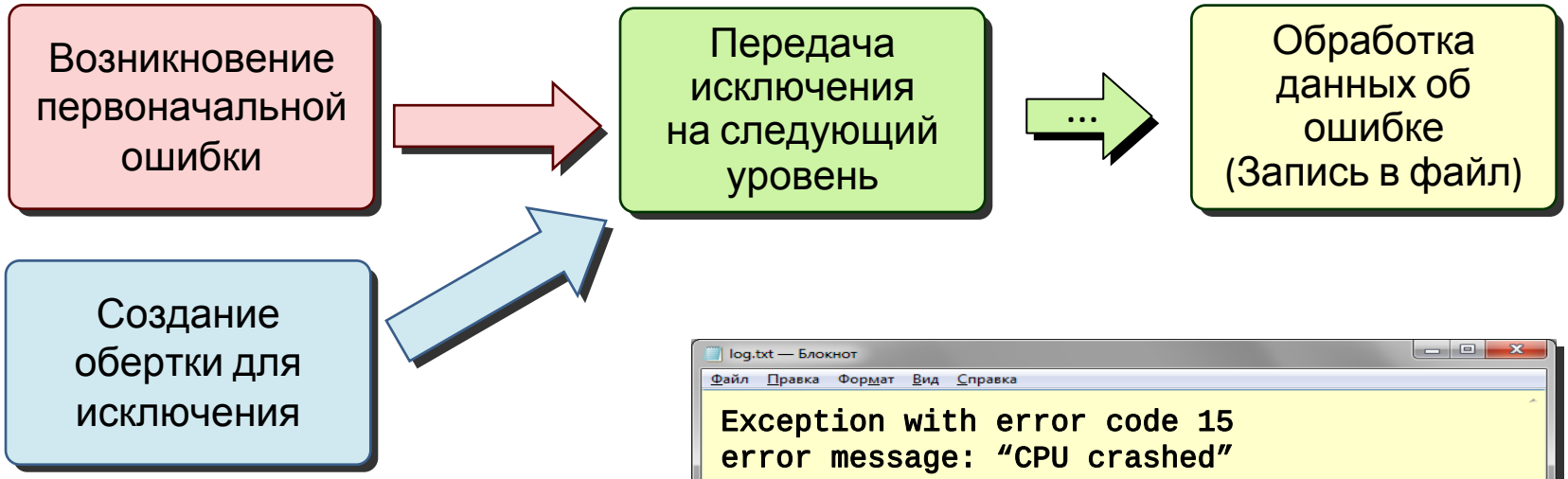
```
…
push %len
…
```



```
…
push 0x6E3194
…
```

# Система обработки ошибок

- При исполнении кода могут возникнуть ошибки, например сбой в стеке




Пример ошибки деления

```
60
61    push 5    ; stack contains 5
62    push 0    ; stack contains 0
63    divs      ; 5 /= 0      Error
64
```

log.txt — Блокнот

Файл Правка Формат Вид Справка

```
Exception with error code 15
error message: “CPU crashed”
occurred in file:
“F:\Projects\...\Compiler.h” on line 60
caused by:
Exception with error code 17
error message: “divs: Divide by 0 error”
occurred in file:
“F:\Projects\...\Compiler.h” on line 586
```

# Оконная система

- Выделение отдельного потока для каждого окна
- Облегченная работа с графической оболочкой

**Главный поток**

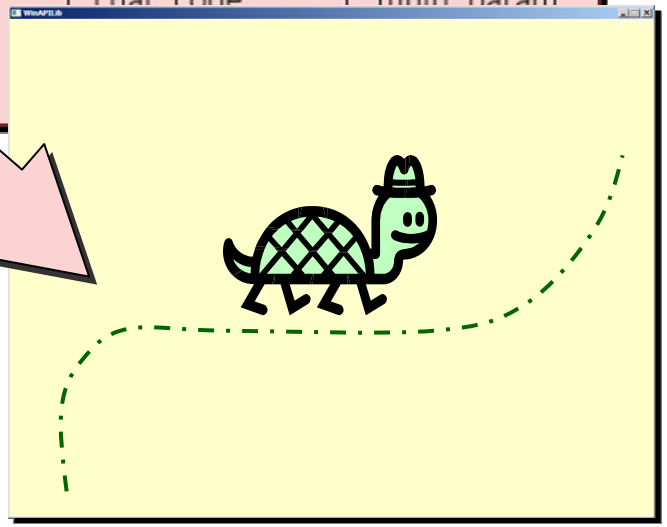
- Работа с классом _Window
- Упрощенное управление графической оболочкой

**Вспомогательный поток**

- Создание окна и обработка сообщений
- Непосредственное взаимодействие с Win32

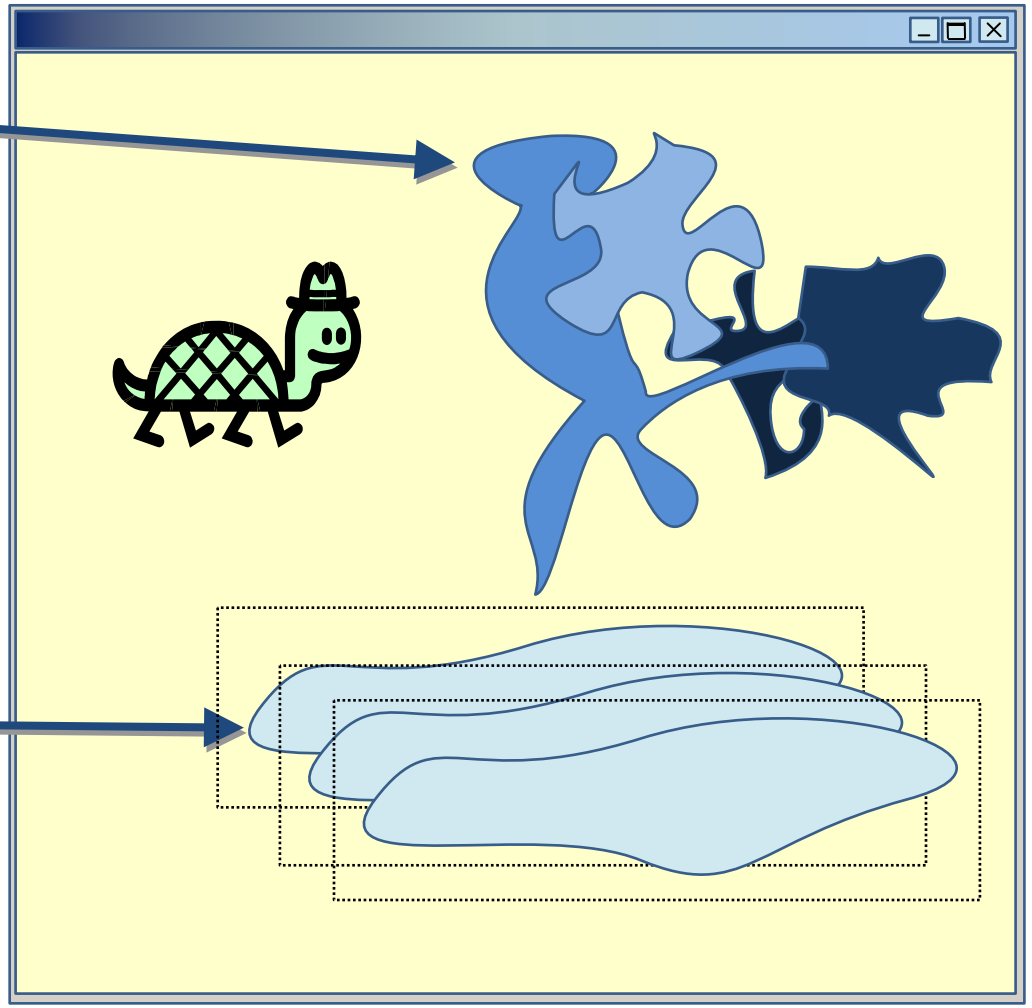
Работа вспомогательного потока

| Message | wParam | lParam |
|---|---|---|
| WM_MOUSEMOVE | MK_LBUTTON | (y << 16) \| x |
| WM_MOVE | not used | (y << 16) \| x |
| WM_PAINT | not used | not used |
| WM_ACTIVATE | WA_ACTIVE | wnd_handle |
| WM_CHAR | char_code | multi_param |
| WM_TIMER | | |
| ... | | |



# Система слоев

- Хранение различных HDC для рисования с возможностью прозрачности
- Хранение указателей на _Layer в std::vector и их отображение при перерисовке окна
- Движение слоев относительно окна



# Эмулятор языка Logo

- Небольшой проект для отработки взаимодействия модулей

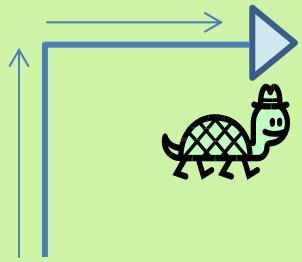
Реализация основных функций Logo (fw, bk, rt, lt …) и работа с ними через виртуальный процессор

Рисование “черепашки” через оконную систему и систему слоев

Logo.lib

```
extern fw
extern bk
extern rt
extern lt
extern circle
extern square
...
```

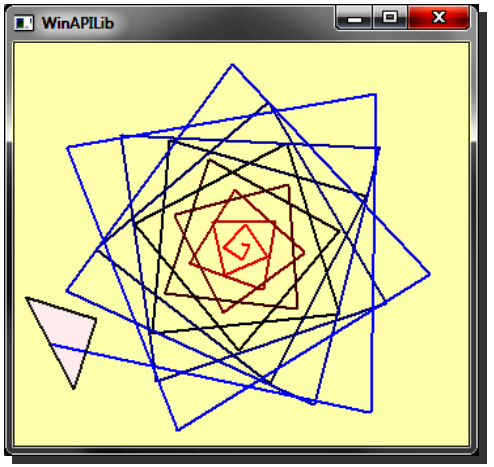
Движение черепашки



**fw 100**

**rt 90**

**fw 100**



# Пример простейшего скрипта

- Синтаксис, схожий с ассемблером

```
link “logo.lib”  ; импорт функций из C++
start            ; исполнение начнется с этой строчки
var %i           ; создана переменная i
1:               ; начало цикла
 sel %i          ; текущий выбранный объект – переменная i
 cmpr 10         ; сравнение выбранного объекта с 10
 jae L2          ; переход на L2, если  i>= 10 (above or equal)
 addr 1          ; i += 1
 fw 100          ; черепашка сдвигается вперед на 100
 rt 105          ; черепашка поворачивает направо на 105 градусов
 jmp L1          ; переход в L1, повтор цикла
2:               ; выход из цикла
eof              ; конец программы
```

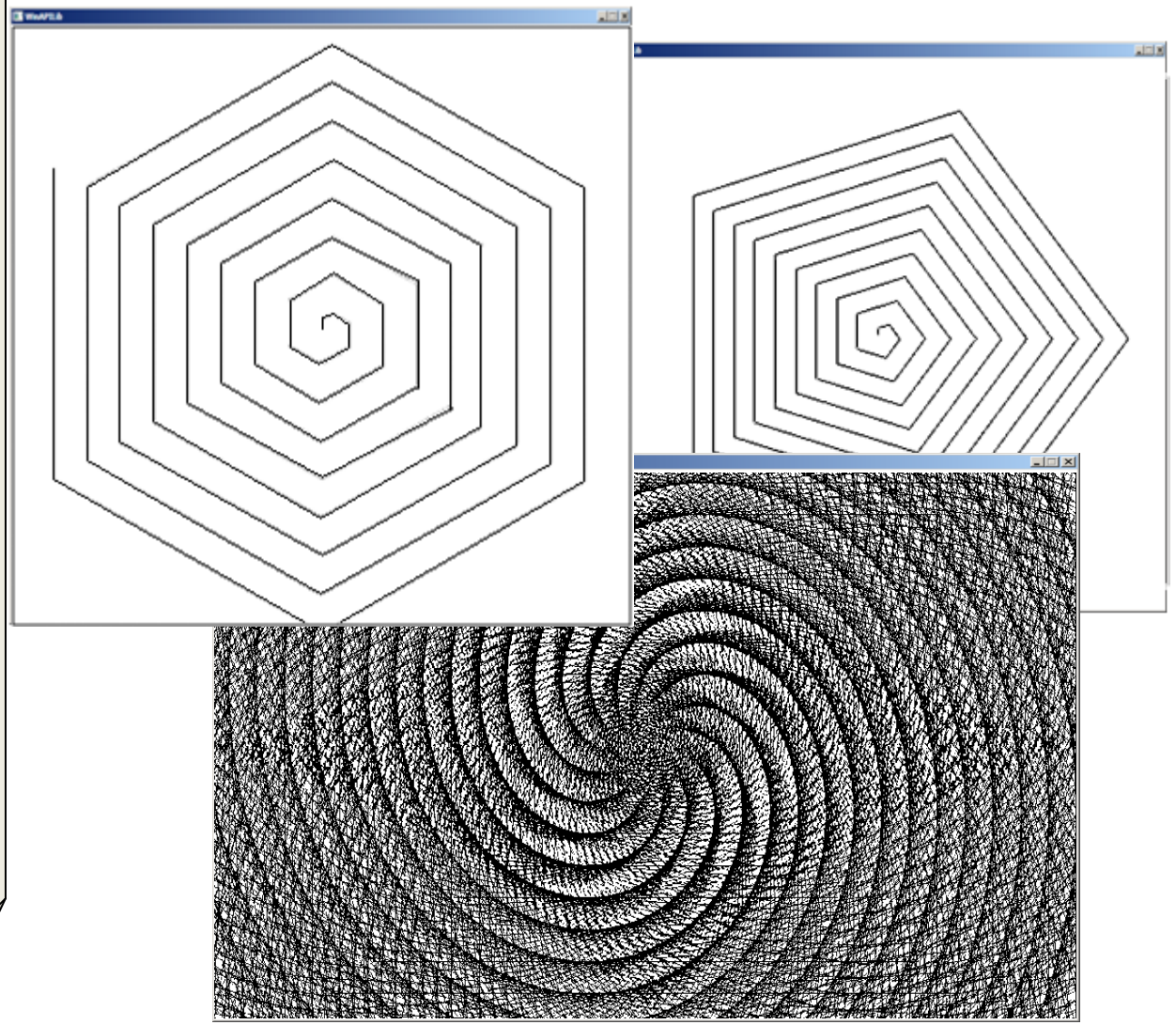
# Примеры работы

```
…
printl “Angle:"
ascii 10
s_input_ aqx
push aqx

printl "Jump:"
ascii 10
s_input_ aqx
push aqx

printl “Max length:"
ascii 10
s_input_ aqx
push aqx

call _Spiral
eof
…
```



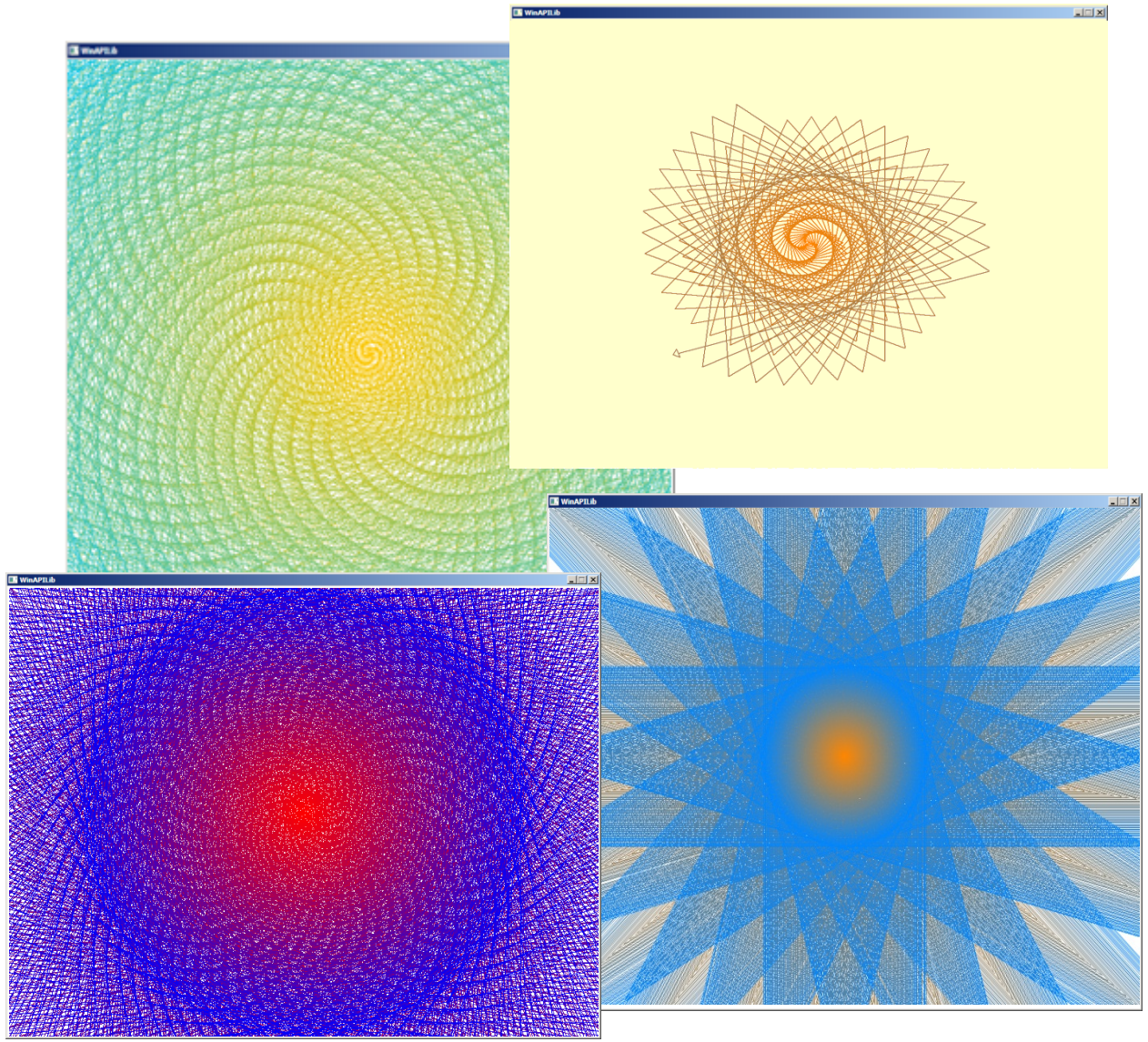
# Примеры работы

```
...

1:
    colorB %currlen
    colorR %red
    sel %red
    subr %jump

    rt %angle
    fw %currlen
    sel %currlen
    addr %jump
    cmpr %maxlen
    sleep 1
  jl L1

...
```



# Результаты работы

- Реализован виртуальный процессор для исполнения кода
- Разработан язык ассемблера для этого процессора
- Разработана библиотека окон над Win32
- Модули готовы для включения в другие проекты, что активно используется
- Объем кода составляет ≈ 6000 строк

# Дополнительные задачи

- Работа с файлами и строками
- Переработка скрипта в более удобный формат для исполнения
- Работа со стеком и с системой регистров
- Регистрация пользовательских функции с и без возможности передачи параметров
- Разработка стека на собственном векторе
- Разработка многопоточной оконной системы, предоставляющая работу с объектами
- Предотвращение ошибок при создании и обработки исключений
- Работа с виртуальными холстами (HDC)