

# Основы программирования

ФИСТ 1 курс

Власенко

Олег

Федосович

Лекция 4

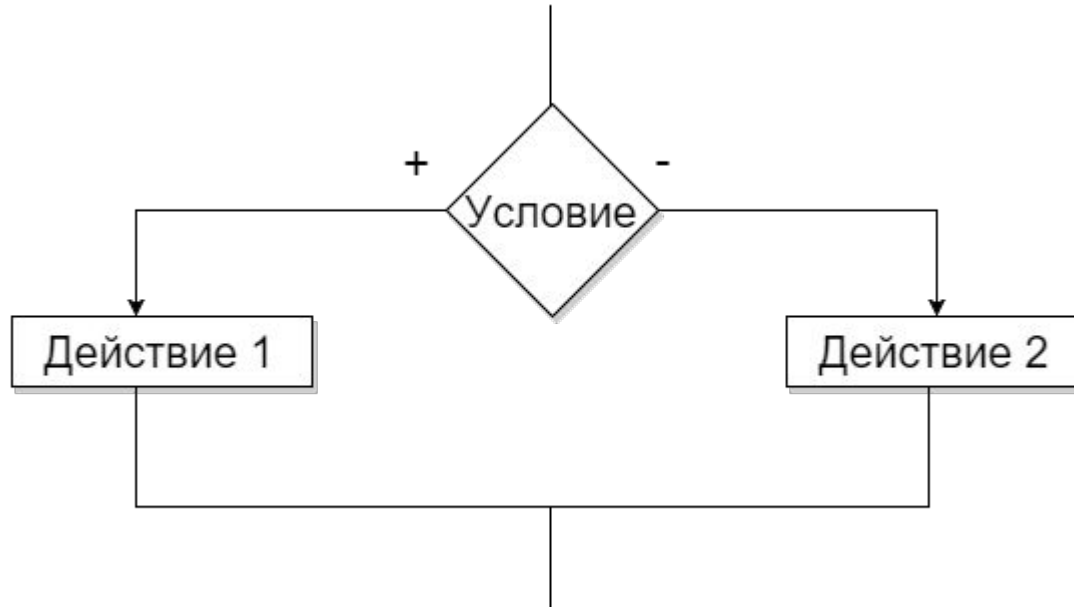
If и Switch.

Вывод текста в графике.

Обработка нажатий клавиш.

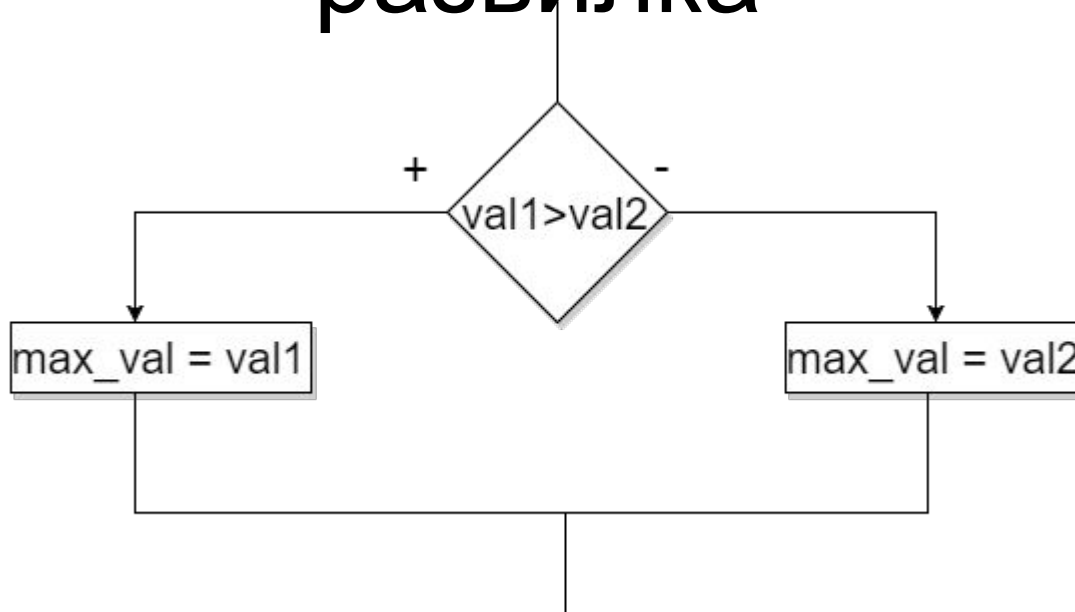
**Делаем простую игру.**

# Развилка (if)



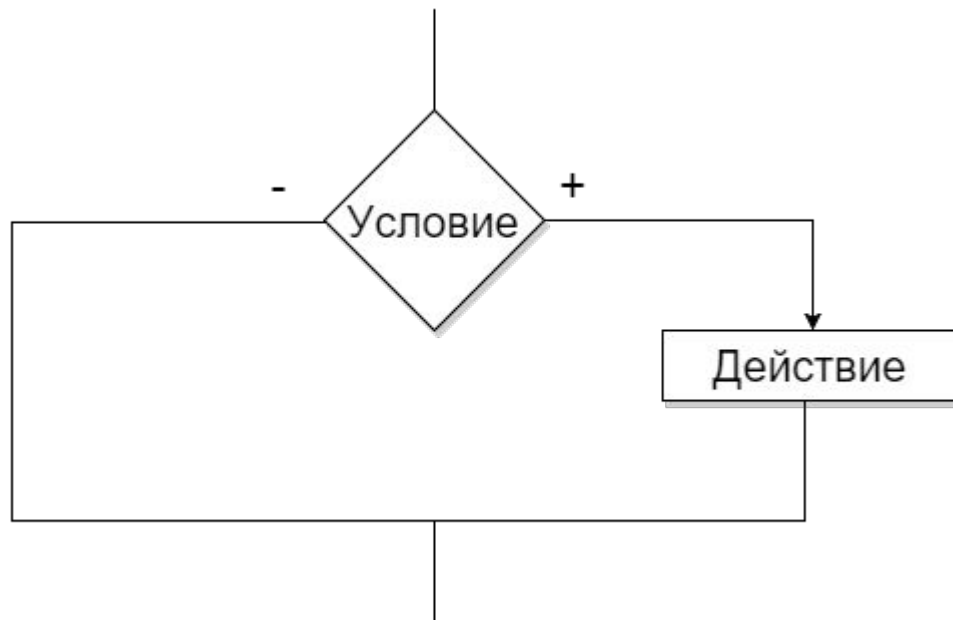
```
if (Условие)  
    Действие1;  
else  
    Действие2;
```

# Найти максимум - полная развилка



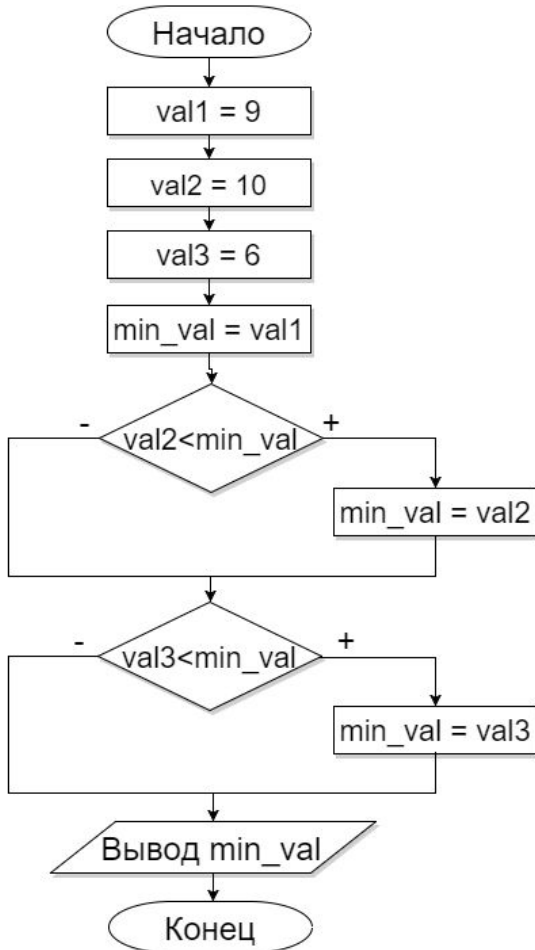
```
if (val1 > val2) {  
    max_val = val1;  
} else {  
    max_val = val2;  
}
```

# Усеченная развилка



```
if (Условие) {  
    Действие;  
}
```

# Минимум из 3 чисел



```
void main() {
```

```
int val1 = 9;
```

```
int val2 = 10;
```

```
int val3 = 6;
```

```
int min_val = val1; // берем за минимальный val1
```

```
if (val2 < min_val) { // если второе меньше  
    min_val = val2; // то теперь минимальное val2  
}
```

```
if (val3 < min_val) { // если третье меньше  
    min_val = val3; // то теперь минимальное val3  
}
```

```
printf("min_val = %i", min_val);
```

```
}
```

# Оператор ветвления Switch

```
switch (wParam)
```

```
{
```

```
    case VK_DOWN:
```

```
        moveDown();
```

```
        InvalidateRect(hWnd, NULL, TRUE);
```

```
        break;
```

```
    case VK_LEFT:
```

```
        moveToLeft();
```

```
        InvalidateRect(hWnd, NULL, TRUE);
```

```
        break;
```

```
    case VK_UP:
```

```
        moveUp();
```

```
        InvalidateRect(hWnd, NULL, TRUE);
```

```
        break;
```

```
    case VK_RIGHT:
```

```
        moveToRight();
```

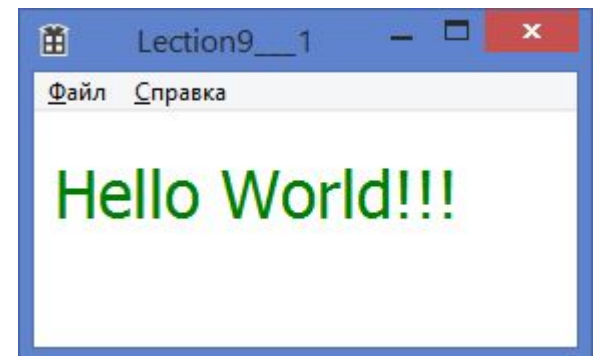
```
        InvalidateRect(hWnd, NULL, TRUE);
```

```
        break;
```

```
}
```

# Шрифты и вывод текста

```
HFONT hFont;  
hFont = CreateFont(40,  
    0, 0, 0, 0, 0, 0, 0,  
    DEFAULT_CHARSET,  
    0, 0, 0, 0,  
    L"Tahoma"  
    );  
SelectObject(hdc, hFont);  
SetTextColor(hdc, RGB(0, 128, 0));  
TCHAR text[] = _T("Hello World!!!");  
TextOut(hdc, 10, 20, text, _tcslen(text));  
DeleteObject(hFont);
```



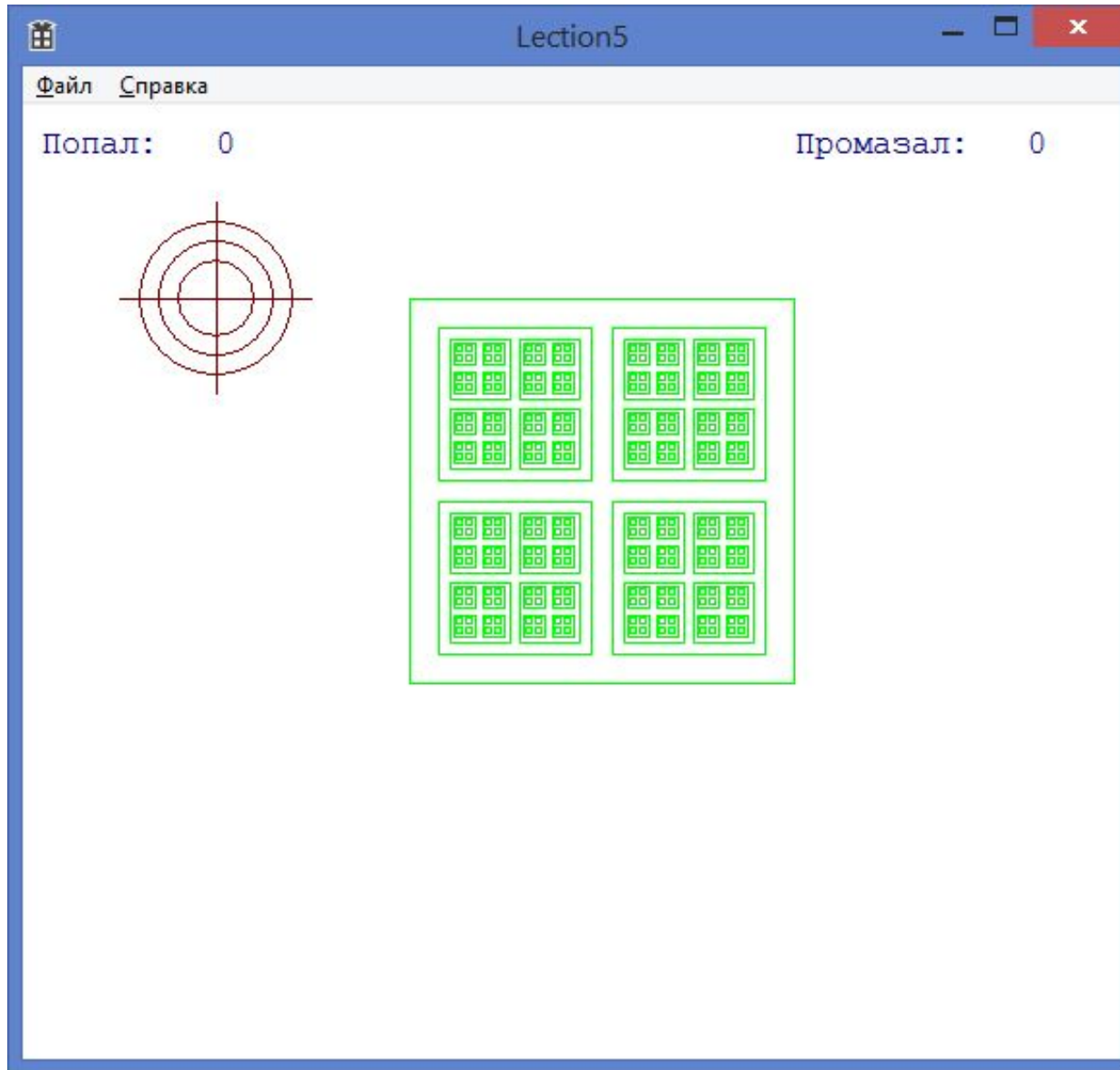
# Шрифты и вывод текста

```
HFONT hFont;  
hFont = CreateFont(80,  
0, 0, 0, 0, 0, 0, 0,  
DEFAULT_CHARSET,  
0, 0, 0, 0,  
L"Courier New"  
);  
SelectObject(hdc, hFont);  
SetTextColor(hdc, RGB(0, 128, 128));  
TCHAR text[] = _T("Hello World!!!");  
TextOut(hdc, 10, 20, text, _tcslen(text));  
DeleteObject(hFont);
```





# Создаем игру – со стрельбой по мишени



# Что нужно добавить в stdafx.h

```
// stdafx.h: включаемый файл для стандартных системных  
включаемых файлов  
// или включаемых файлов для конкретного проекта,  
которые часто используются, но  
// не часто изменяются  
//
```

```
#pragma once
```

```
#define _CRT_SECURE_NO_WARNINGS
```

```
#include "targetver.h"
```

# Что нужно добавить в сpp файл

```
// Lektion5.cpp: определяет точку входа для приложения.
```

```
//
```

```
#include "stdafx.h"
```

```
#include "Lektion5.h"
```

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

```
#define MAX_LOADSTRING 100
```

```
// Глобальные переменные:
```

```
HINSTANCE hInst; // текущий экземпляр
```

```
WCHAR szTitle[MAX_LOADSTRING]; // Текст строки заголовка
```

```
WCHAR szWindowClass[MAX_LOADSTRING]; //имя класса
```

```
главного...
```

# ПИСЕМ КОЛИЧЕСТВО ПОПАДАНИИ И ПРОМАХОВ (1)

```
case WM_PAINT:
```

```
{  
    PAINTSTRUCT ps;  
    HDC hdc = BeginPaint(hWnd, &ps);  
    // TODO: Добавьте сюда любой код прорисовки, использующий
```

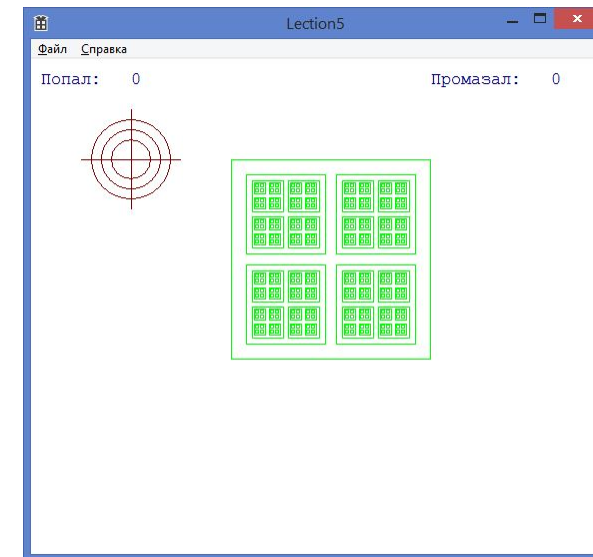
```
HDC...
```

```
    HFONT hFont = CreateFont(20,  
                             0, 0, 0, 0, 0, 0,  
                             DEFAULT_CHARSET,  
                             0, 0, 0, 0,  
                             L"Courier New"  
    );
```

```
    SelectObject(hdc, hFont);  
    SetTextColor(hdc, RGB(0, 0, 128));
```

```
    TCHAR string1[] = _T("Попал:");  
    TextOut(hdc, 10, 10, string1, _tcslen(string1));
```

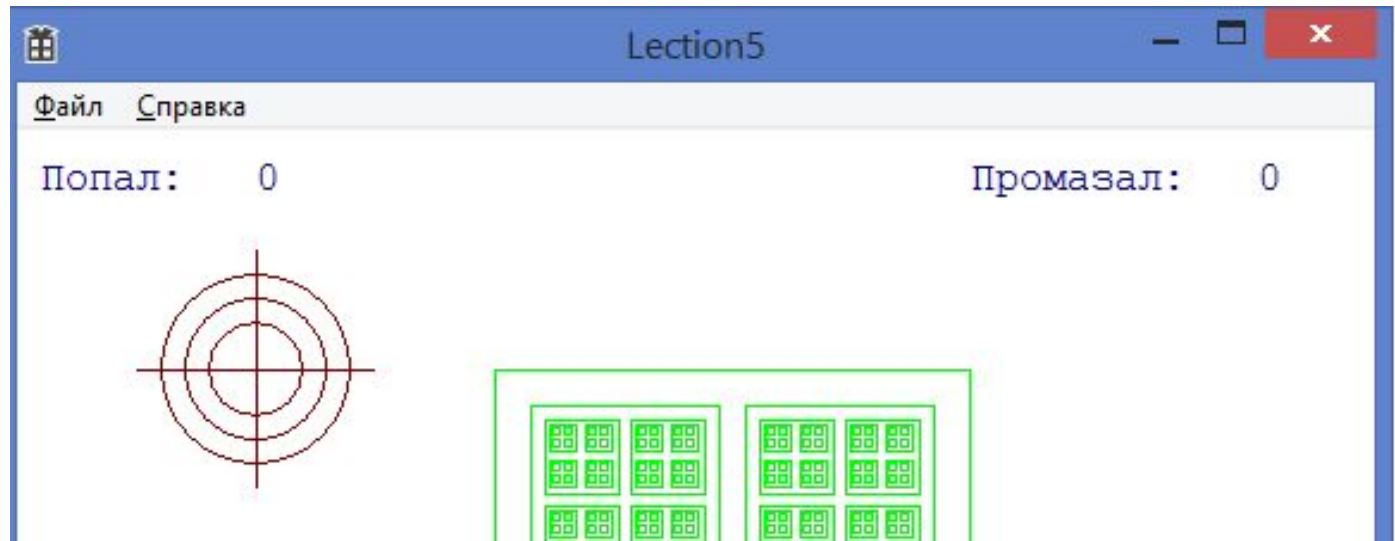
```
    TCHAR string2[] = _T("Промазал:");  
    TextOut(hdc, 400, 10, string2, _tcslen(string2));
```



# ПИСЕМ КОЛИЧЕСТВО ПОПАДАНИИ И ПРОМАХОВ (2)

```
char sHit[5]; // локальная переменная sHit
TCHAR tsHit[5];
sprintf(sHit, "%d", hit); // использование глобальной переменной hit
OemToChar(sHit, tsHit);
TextOut(hdc, 100, 10, tsHit, _tcslen(tsHit));
```

```
char sMissed[5];
TCHAR tsMissed[5];
sprintf(sMissed, "%d", missed);
OemToChar(sMissed, tsMissed);
TextOut(hdc, 520, 10, tsMissed, _tcslen(tsMissed));
```



# Отрисовываем Цель и Прицел

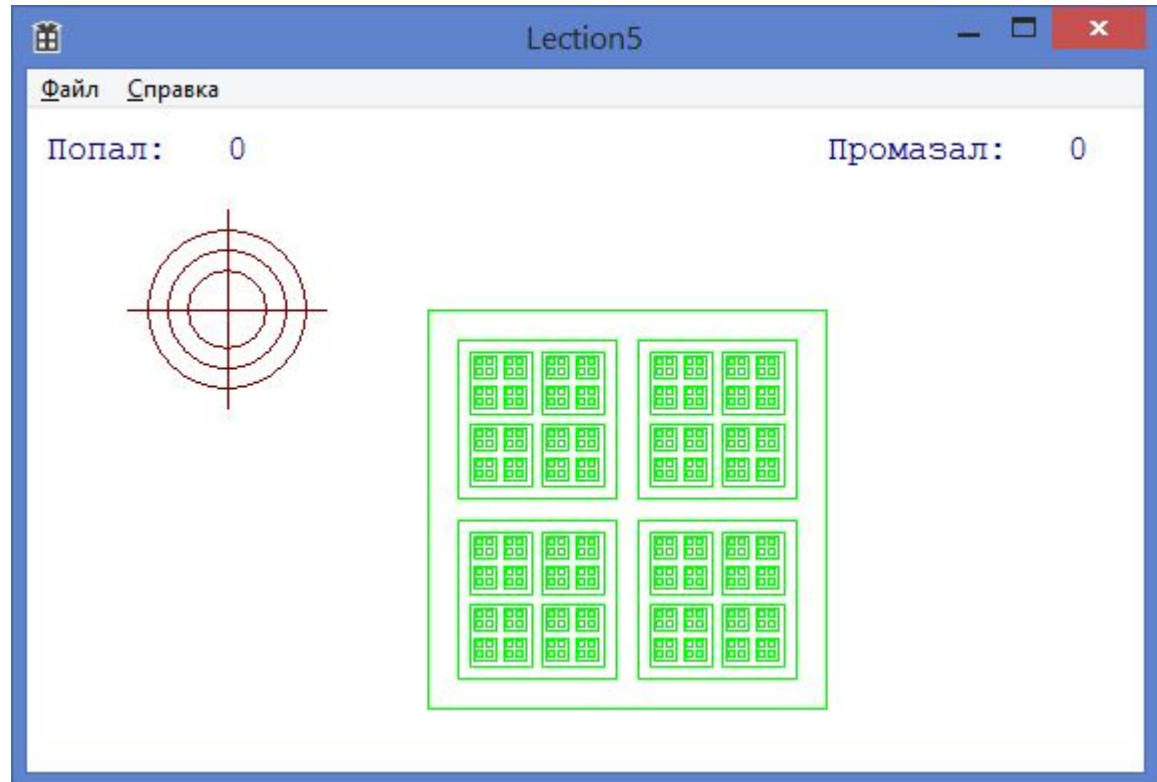
```
DrawGoal(hdc, goalCx, goalCy, goalSize);
```

```
DrawAim(hdc, aimX, aimY);
```

```
EndPaint(hWnd, &ps);
```

```
}
```

```
break;
```



# Управление перемещением прицела

```
case WM_KEYDOWN:
    switch (wParam)
    {
    case VK_DOWN:
        moveDown();
        InvalidateRect(hWnd, NULL, TRUE);
        break;
    case VK_LEFT:
        moveToLeft();
        InvalidateRect(hWnd, NULL, TRUE);
        break;
    case VK_UP:
        moveUp();
        InvalidateRect(hWnd, NULL, TRUE);
        break;
    case VK_RIGHT:
        moveToRight();
        InvalidateRect(hWnd, NULL, TRUE);
        break;
    }
    break;
```

# Управление огнем

```
case WM_KEYDOWN:
```

```
    switch (wParam)
```

```
    {
```

```
        case VK_DOWN:
```

```
            ...
```

```
            break;
```

```
        case VK_RETURN:
```

```
            if (insideGoal(aimX, aimY)) {
```

```
                hit++;
```

```
            }
```

```
            else {
```

```
                missed++;
```

```
            }
```

```
            InvalidateRect(hWnd, NULL, TRUE);
```

```
            break;
```

```
    }
```

```
    break;
```



# Глобальные переменные

// Параметры цели (мишени)

**int goalCx = 300;**

**int goalCy = 200;**

**int goalSize = 100;**

// Параметры прицела

**int aimX = 100;**

**int aimY = 100;**

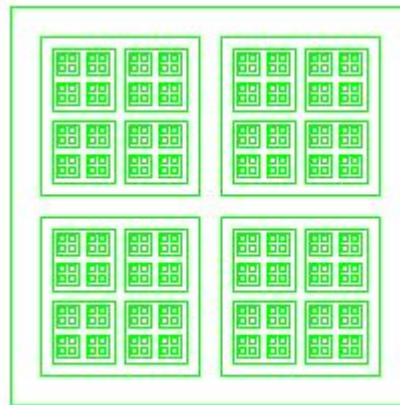
// Счетчики выстрелов

**int hit = 0; // попал**

**int missed = 0; // промазал**

# Рисуем цель

```
void DrawGoal(HDC hdc, int x, int y, int size) {  
    HPEN hPen = CreatePen(PS_SOLID, 1, RGB(0, 255, 0));  
    SelectObject(hdc, hPen);  
  
    RecursiveRectagle(hdc, x, y, size);  
  
}
```



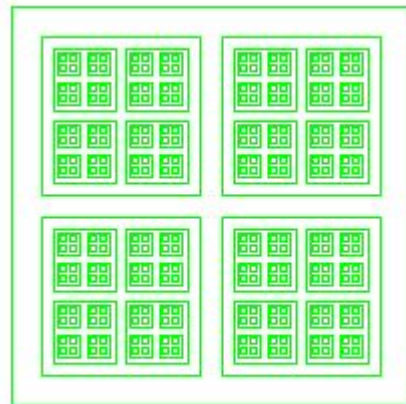
# Рисуем цель (2)

```
void RecursiveRectagle(HDC hdc, int cx, int cy, int size) {  
    Rectangle(hdc, cx - size, cy - size, cx + size, cy + size);
```

```
    if (size < 5) {  
        return;  
    }
```

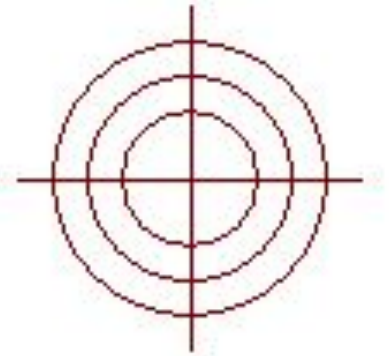
```
    RecursiveRectagle(hdc, cx - size * 0.45, cy - size * 0.45, size / 2.5);  
    RecursiveRectagle(hdc, cx + size * 0.45, cy - size * 0.45, size / 2.5);  
    RecursiveRectagle(hdc, cx - size * 0.45, cy + size * 0.45, size / 2.5);  
    RecursiveRectagle(hdc, cx + size * 0.45, cy + size * 0.45, size / 2.5);
```

```
}
```



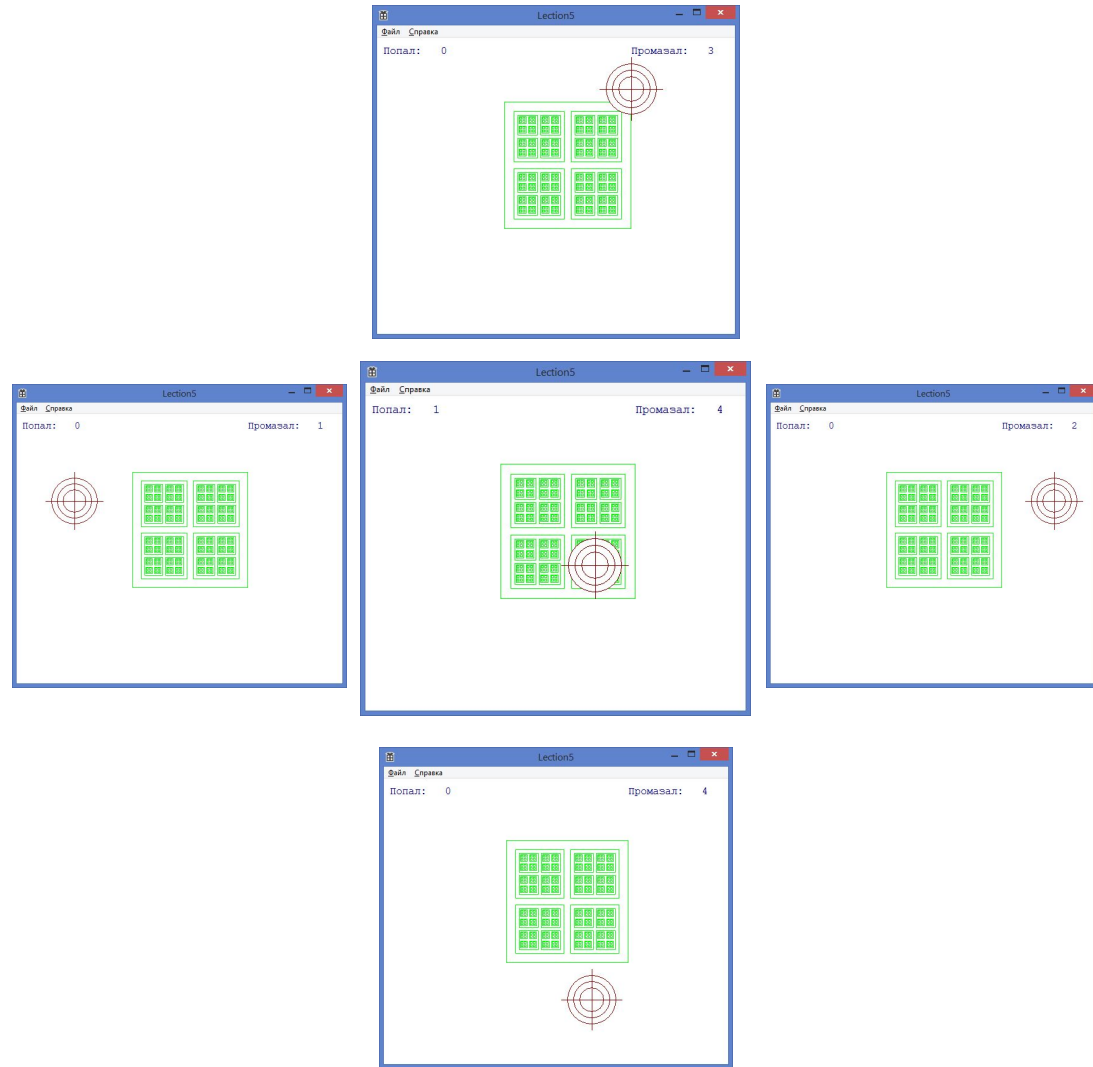
# Рисуем прицел

```
void DrawAim(HDC hdc, int x, int y) {  
    HPEN hPen = CreatePen(PS_SOLID, 1, RGB(128, 0, 0));  
    SelectObject(hdc, hPen);  
  
    Ellipse(hdc, x - 40, y - 40, x + 40, y + 40);  
    Ellipse(hdc, x - 30, y - 30, x + 30, y + 30);  
    Ellipse(hdc, x - 20, y - 20, x + 20, y + 20);  
  
    MoveToEx(hdc, x - 50, y, NULL);  
    LineTo(hdc, x + 50, y);  
  
    MoveToEx(hdc, x, y - 50, NULL);  
    LineTo(hdc, x, y + 50);  
}
```



# Проверка попадания в цель

```
int insideGoal(int x, int y) {  
    if (x < goalCx - goalSize)  
        return 0;  
    if (x > goalCx + goalSize)  
        return 0;  
    if (y < goalCy - goalSize)  
        return 0;  
    if (y > goalCy + goalSize)  
        return 0;  
    return 1;  
}
```



# Собственно перемещение прицела

```
void moveDown() {  
    aimY += 10;  
}
```

```
void moveToLeft() {  
    aimX -= 10;  
}
```

```
void moveUp() {  
    aimY -= 10;  
}
```

```
void moveToRight() {  
    aimX += 10;  
}
```

# Разное

- Локальные и глобальные переменные
- Передача параметров в функции
- Возвращение результата из функции

# Домашнее задание

1. \*\* Собрать игрушку из того, что есть в слайдах
2. \*\*\*\* Написать свою собственную игру – не стреляем по мишени, а собираем грибы в корзину, поливаем огород, и т.п.



# Источники информации

- **Virtual-Key Codes -**

<https://msdn.microsoft.com/ru-ru/library/windows/desktop/dd375731%28v=vs.85%29.aspx>