

Основы программирования

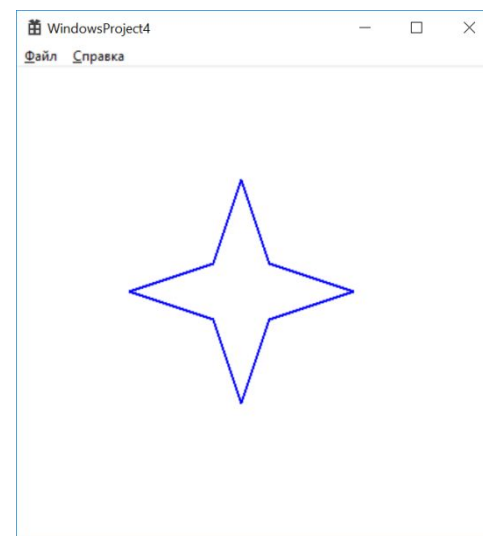
Лабораторная работа №15

Вложенные
циклы

Власенко Олег Федосович

*Задача 2 (ЛР14) – пример кода через Polyline

```
440 void Star1(HDC hdc, int cx, int cy, int size) {
441
442     POINT p[9] = {
443         cx, cy - size,
444         cx + size / 4, cy - size / 4,
445         cx + size, cy,
446         cx + size / 4, cy + size / 4,
447         cx, cy + size,
448         cx - size / 4, cy + size / 4,
449         cx - size, cy,
450         cx - size / 4, cy - size / 4,
451         cx, cy - size,
452     };
453
454     Polyline(hdc, p, 9);
455 }
```



Звезда с разными шириной и высотой

```
void Star(HDC hdc, int cx, int cy, int sizeX, int sizeY) {
```

```
    POINT p[9] = {  
        cx,          cy - sizeY,  
        cx + sizeX / 4,  cy - sizeY / 4,  
        cx + sizeX,   cy,  
        cx + sizeX / 4,  cy + sizeY / 4,  
        cx,          cy + sizeY,  
        cx - sizeX / 4,  cy + sizeY / 4,  
        cx - sizeX,   cy,  
        cx - sizeX / 4,  cy - sizeY / 4,  
        cx,          cy - sizeY  
    };
```

```
    Polyline(hdc, p, 9);
```

```
}
```

```
378 |  
379 |  
380 |  
381 |  
382 |  
383 |  
384 |  
385 |  
386 |  
    case WM_PAINT:  
    {  
        PAINTSTRUCT ps;  
        HDC hdc = BeginPaint(hWnd, &ps);  
  
        Star(hdc, 400, 300, 200, 100);  
  
        EndPaint(hWnd, &ps);  
    }
```



Звезда с разными шириной и высотой

378
379
380
381
382
383
384
385
386
387
388

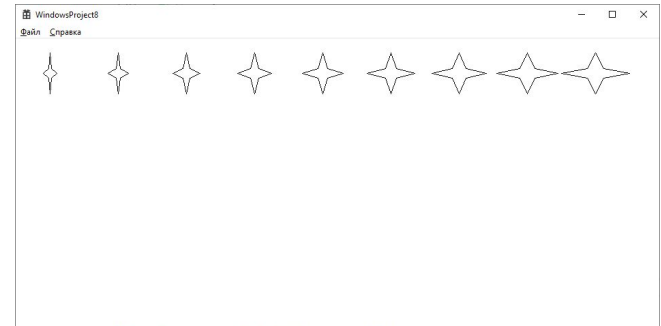


```
case WM_PAINT:  
{  
    PAINTSTRUCT ps;  
    HDC hdc = BeginPaint(hWnd, &ps);  
  
    Star(hdc, 100, 100, 100, 200);  
    Star(hdc, 100, 300, 100, 100);  
    Star(hdc, 400, 300, 200, 100);  
  
    EndPaint(hWnd, &ps);  
}
```



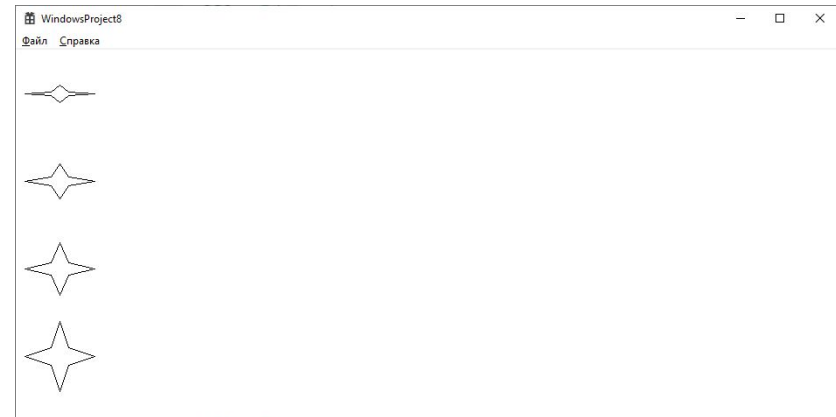
Звёзды в ряд! (горизонтальный)

```
case WM_PAINT:  
{  
    PAINTSTRUCT ps;  
    HDC hdc = BeginPaint(hWnd, &ps);  
  
    int y = 50;  
    int sizeY = 30;  
  
    int x = 50;  
    int sizeX = 10;  
  
    while (x <= 900) {  
  
        Star(hdc, x, y, sizeX, sizeY);  
        x += 100;  
  
        sizeX += 5;  
    }  
  
    EndPaint(hWnd, &ps);  
}
```



Звёзды в ряд! (вертикальный)

```
case WM_PAINT:  
{  
    PAINTSTRUCT ps;  
    HDC hdc = BeginPaint(hWnd, &ps);  
  
    int y = 50;  
    int sizeY = 10;  
  
    int x = 50;  
    int sizeX = 40;  
    while (y < 400) {  
  
        Star(hdc, x, y, sizeX, sizeY);  
        y += 100;  
  
        sizeY += 10;  
    }  
  
    EndPaint(hWnd, &ps);  
}
```



Звёзды в ряд! (диагональный)

```
case WM_PAINT:  
{  
    PAINTSTRUCT ps;  
    HDC hdc = BeginPaint(hWnd, &ps);  
  
    int y = 50;  
    int sizeY = 20;  
  
    int x = 50;  
    int sizeX = 10;  
    while (y < 400) {  
        Star(hdc, x, y, sizeX, sizeY);  
  
        x += 100;  
        sizeX += 8;  
  
        y += 50;  
        sizeY += 2;  
    }  
  
    EndPaint(hWnd, &ps);  
}
```



Звёзды в ряды!

```
case WM_PAINT:
{
    PAINTSTRUCT ps;
    HDC hdc = BeginPaint(hWnd, &ps);

    int y = 50;
    int sizeY = 10;
    while (y < 400) {

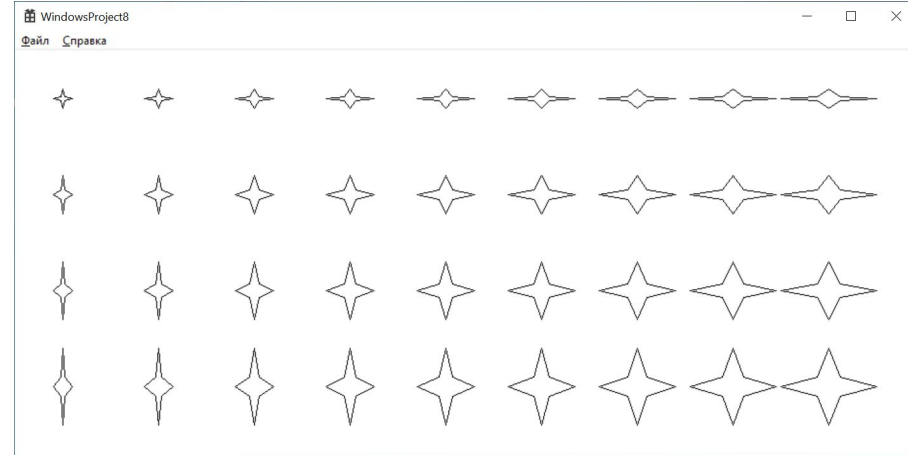
        int x = 50;
        int sizeX = 10;
        while (x < 900) {

            Star(hdc, x, y, sizeX, sizeY);

            x += 100;
            sizeX += 5;
        }

        y += 100;
        sizeY += 10;
    }

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



Звёзды в ряды! (порядок рисования!!!)

```
case WM_PAINT:  
{  
    PAINTSTRUCT ps;  
    HDC hdc = BeginPaint(hWnd, &ps);  
  
    int y = 50;  
    int sizeY = 10;  
    while (y < 400) {  
        int x = 50;  
        int sizeX = 10;  
        while (x < 900) {  
            Star(hdc, x, y, sizeX, sizeY);  
            Sleep(100);  
            x += 100;  
            sizeX += 5;  
        }  
  
        y += 100;  
        sizeY += 10;  
    }  
  
    EndPaint(hWnd, &ps);  
}
```



Звёзды в ряды! В отдельной функции

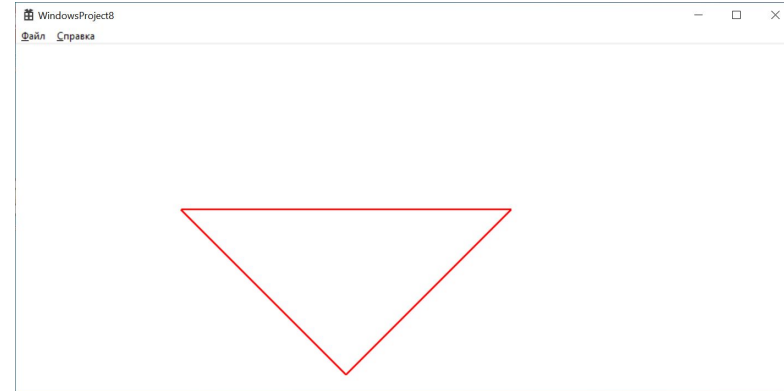
```
void BlockStars(HDC hdc) {  
    int y = 50;  
    int sizeY = 10;  
    while (y < 400) {  
        int x = 50;  
        int sizeX = 10;  
        while (x < 900) {  
            Star(hdc, x, y, sizeX, sizeY);  
  
            x += 100;  
            sizeX += 5;  
        }  
  
        y += 100;  
        sizeY += 10;  
    }  
}
```



```
379  
380  
381  
382  
383  
384  
385  
386  
387  
case WM_PAINT:  
{  
    PAINTSTRUCT ps;  
    HDC hdc = BeginPaint(hWnd, &ps);  
  
    BlockStars(hdc);  
  
    EndPaint(hWnd, &ps);  
}
```

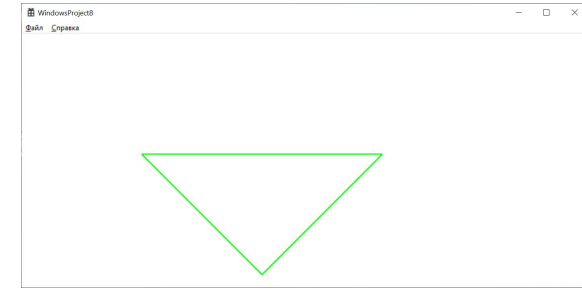
Треугольник с разными шириной, высотой и ЦВЕТОМ

```
void Triangle(HDC hdc, int cx, int cy, int sizeX, int sizeY, COLORREF color) {  
  
    POINT p[] = {  
        cx - sizeX, cy - sizeY,  
        cx + sizeX, cy - sizeY,  
        cx,        cy + sizeY,  
        cx - sizeX, cy - sizeY  
    };  
  
    HPEN hPen;  
    hPen = CreatePen(PS_SOLID, 2, color);  
    SelectObject(hdc, hPen);  
    Polyline(hdc, p, 4);  
    DeleteObject(hPen);  
  
}  
  
...  
case WM_PAINT:  
{  
    PAINTSTRUCT ps;  
    HDC hdc = BeginPaint(hWnd, &ps);  
  
    Triangle(hdc, 400, 300, 200, 100, RGB(255, 0, 0));  
  
    EndPaint(hWnd, &ps);  
}
```

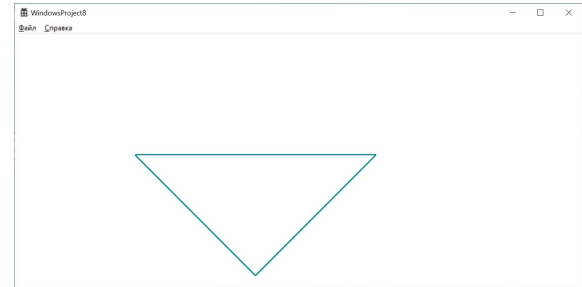


Треугольник с разными шириной, высотой и цветом

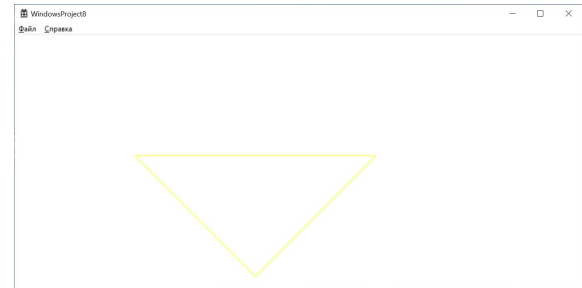
```
378 case WM_PAINT:  
379 {  
380     PAINTSTRUCT ps;  
381     HDC hdc = BeginPaint(hWnd, &ps);  
382  
383     Triangle(hdc, 400, 300, 200, 100, RGB(0, 255, 0));  
384  
385     EndPaint(hWnd, &ps);  
386 }
```



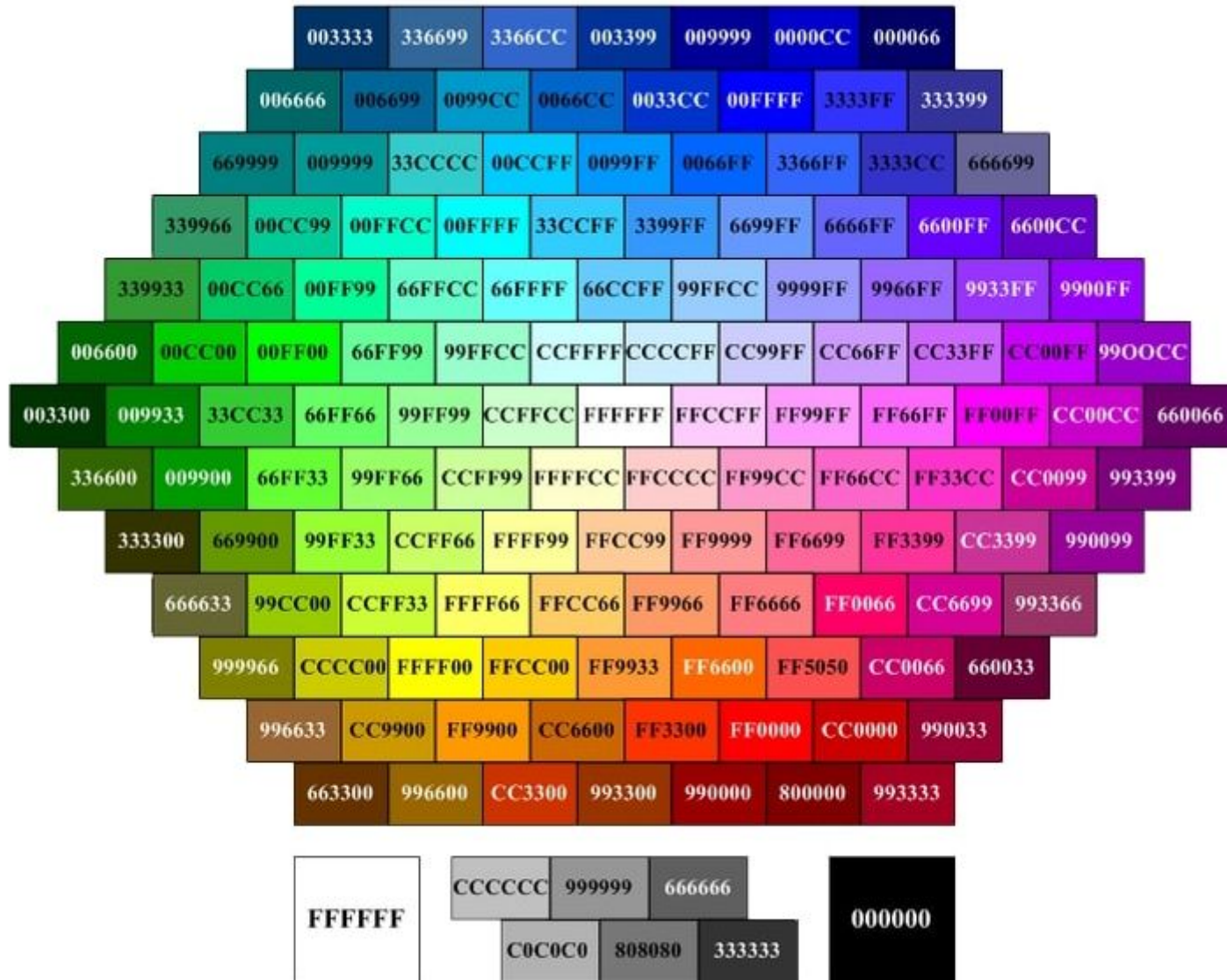
```
378 case WM_PAINT:  
379 {  
380     PAINTSTRUCT ps;  
381     HDC hdc = BeginPaint(hWnd, &ps);  
382  
383     Triangle(hdc, 400, 300, 200, 100, RGB(0, 128, 128));  
384  
385     EndPaint(hWnd, &ps);  
386 }
```



```
378 case WM_PAINT:  
379 {  
380     PAINTSTRUCT ps;  
381     HDC hdc = BeginPaint(hWnd, &ps);  
382  
383     Triangle(hdc, 400, 300, 200, 100, RGB(255, 255, 128));  
384  
385     EndPaint(hWnd, &ps);  
386 }
```



Некоторые цвета



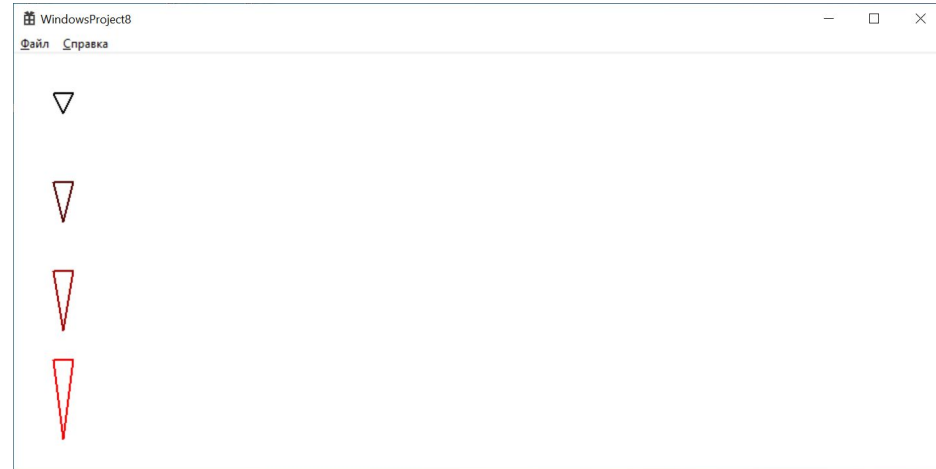
Треугольники в ряд! (горизонтальный)

```
void BlockTriangles1(HDC hdc) {  
    int y = 50;  
    int sizeY = 10;  
  
    int r = 0;  
    int b = 0;  
    int g = 0;  
  
    int x = 50;  
    int sizeX = 10;  
  
    while (x < 900) {  
        Triangle(hdc, x, y, sizeX, sizeY, RGB(r, g, b));  
  
        x += 100;  
        sizeX += 5;  
  
        g += 30;  
    }  
}
```



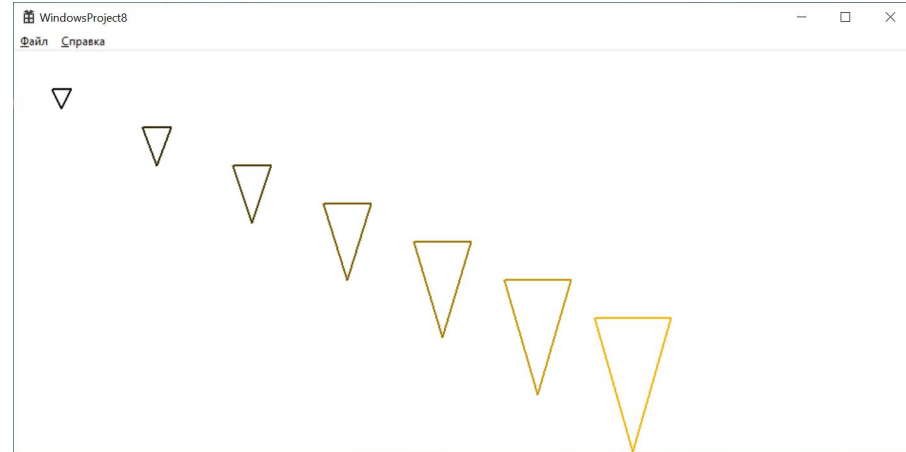
Треугольники в ряд! (вертикальный)

```
257 void BlockTriangles2(HDC hdc) {
258     int y = 50;
259     int sizeY = 10;
260
261     int x = 50;
262     int sizeX = 10;
263
264     int r = 0;
265     int b = 0;
266     int g = 0;
267     while (y < 400) {
268
269         Triangle(hdc, x, y, sizeX, sizeY, RGB(r, g, b));
270
271         y += 100;
272         sizeY += 10;
273
274         r += 80;
275     }
276 }
```



Треугольники в ряд! (диагональный)

```
278 void BlockTriangles3(HDC hdc) {
279     int y = 50;
280     int sizeY = 10;
281
282     int x = 50;
283     int sizeX = 10;
284
285     int r = 0;
286     int b = 0;
287     int g = 0;
288     while (y < 400) {
289
290         Triangle(hdc, x, y, sizeX, sizeY, RGB(r, g, b));
291
292         y += 50;
293         sizeY += 10;
294         r += 40;
295
296         x += 100;
297         sizeX += 5;
298         g += 30;
299     }
300 }
```



Треугольники в несколько рядов!

```
303 void BlockTriangles4(HDC hdc) {
304     int y = 50;
305     int sizeY = 10;
306
307     int r = 0;
308     int b = 0;
309     int g = 0;
310     while (y < 400) {
311
312         int x = 50;
313         int sizeX = 10;
314         g = 0;
315         while (x < 900) {
316             Triangle(hdc, x, y, sizeX, sizeY, RGB(r, g, b));
317
318             x += 100;
319             sizeX += 5;
320             g += 30;
321         }
322
323         y += 100;
324         sizeY += 10;
325         r += 80;
326     }
327 }
```



Трикутники в декількох рядках! (порядок!)

```
303 void BlockTriangles4(HDC hdc) {  
304     int r = 0;  
305     int b = 0;  
306     int g = 0;  
307  
308     int x = 50;  
309     int sizeX = 10;  
310  
311     while (x < 900) {  
312         int y = 50;  
313         int sizeY = 10;  
314         r = 0;  
315  
316         while (y < 400) {  
317  
318             Triangle(hdc, x, y, sizeX, sizeY, RGB(r, g, b));  
319             Sleep(100);  
320  
321             y += 100;  
322             sizeY += 10;  
323             r += 80;  
324         }  
325         x += 100;  
326         sizeX += 5;  
327         g += 30;  
328     }  
329 }
```



Корона с разными шириной, высотой и цветом

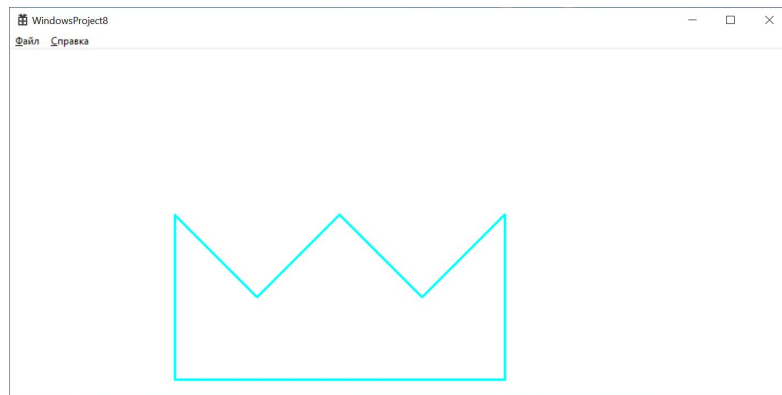
```
void Crown(HDC hdc, int cx, int cy, int sizeX, int sizeY, COLORREF color) {
```

```
    POINT p[] = {  
        cx,      cy - sizeY,  
        cx + sizeX / 2, cy,  
        cx + sizeX, cy - sizeY,  
        cx + sizeX, cy + sizeY,  
        cx - sizeX, cy + sizeY,  
        cx - sizeX, cy - sizeY,  
        cx - sizeX / 2, cy,  
        cx,      cy - sizeY  
    };
```

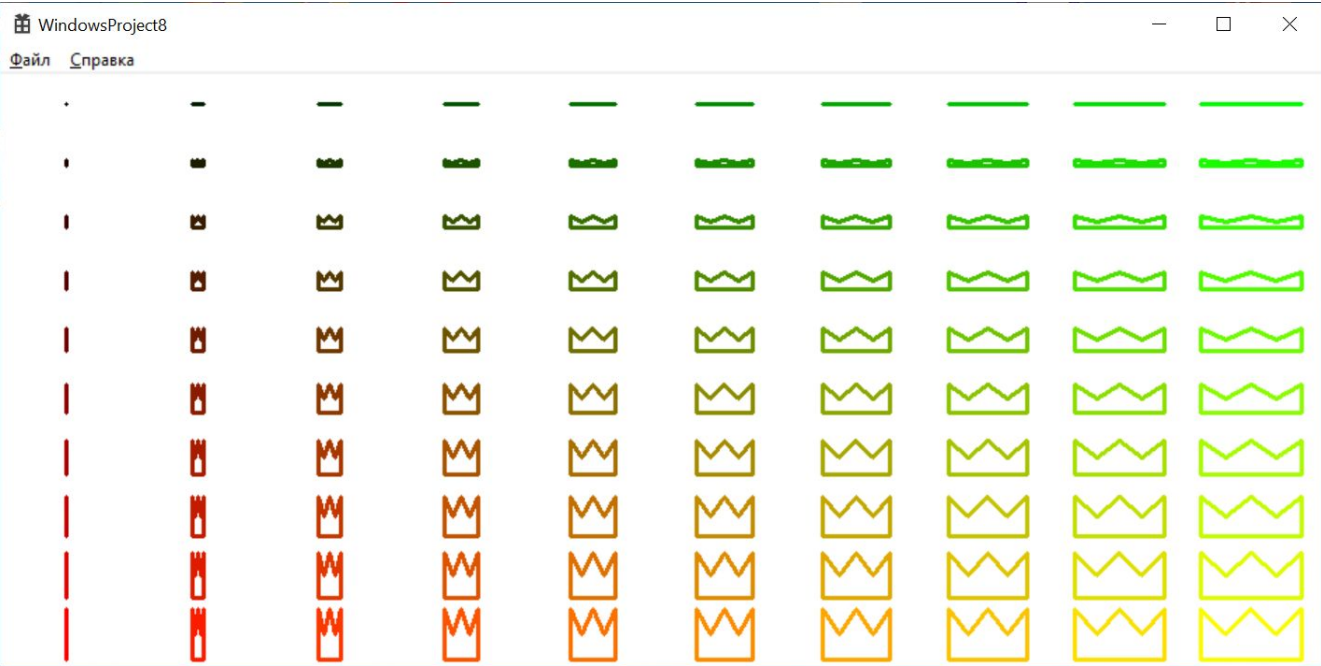
```
    HPEN hPen;  
    hPen = CreatePen(PS_SOLID, 3, color);  
    SelectObject(hdc, hPen);  
    Polyline(hdc, p, 8);  
    DeleteObject(hPen);
```

```
}
```

```
454 case WM_PAINT:  
455 {  
456     PAINTSTRUCT ps;  
457     HDC hdc = BeginPaint(hWnd, &ps);  
458  
459     Crown(hdc, 400, 300, 200, 100, RGB(0, 255, 255));  
460     EndPaint(hWnd, &ps);  
461 }
```



Корона с разными шириной, высотой и цветом



Задача 1.1

Вывести короны в горизонтальный ряд (с изменением ширины и с изменением цвета). Какой цвет менять – на ваш выбор (RED, GREEN, BLUE) .

Задача 1.2

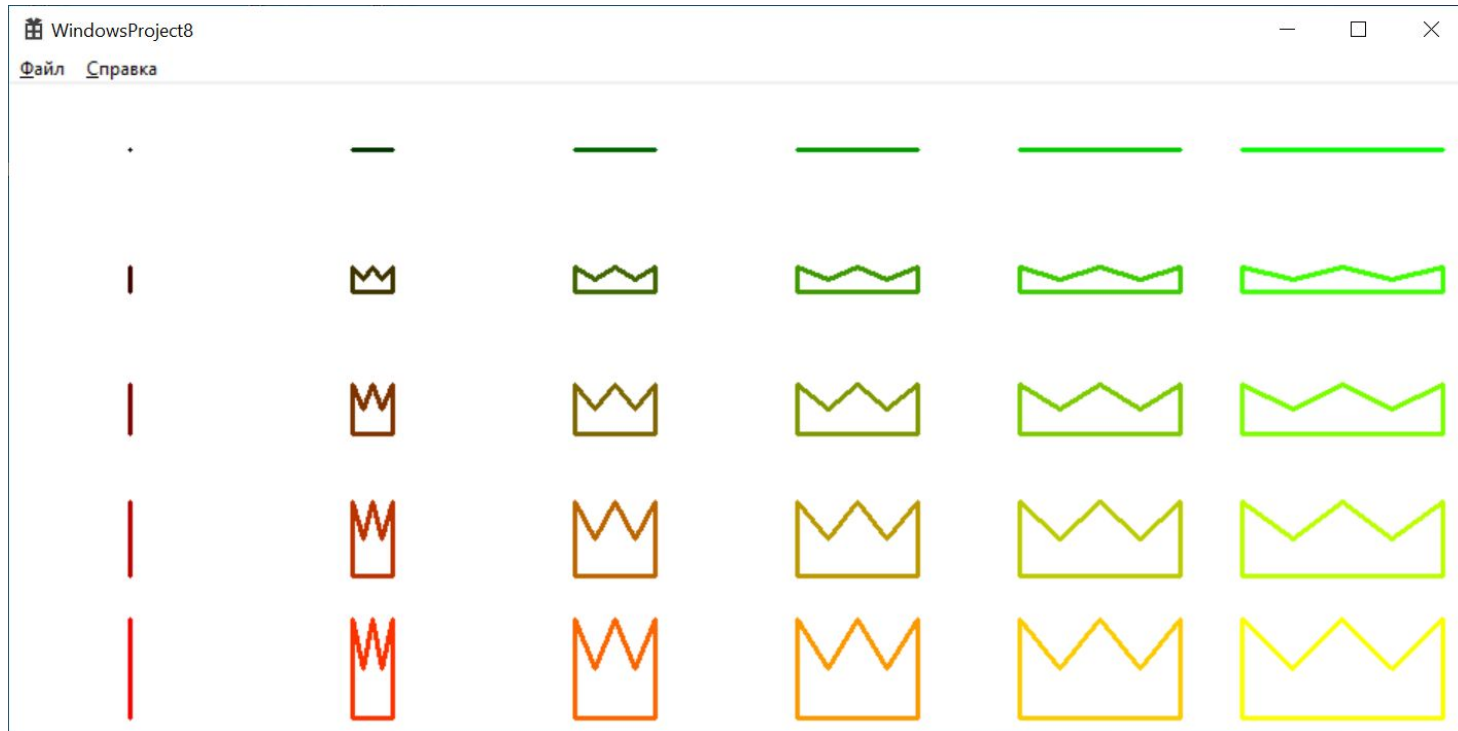
Вывести короны в вертикальный ряд (с изменением высоты и с изменением цвета – отличного от изменения по горизонтали). Какой цвет менять – на ваш выбор (RED, GREEN, BLUE) .

Задача 1.3

Вывести короны в диагональный ряд (с одновременным изменением ширины, высоты и изменением двух цветов).

Задача 1.4

Вывести короны в виде блока (несколько рядов по несколько корон).



Задача 2.0

Создать функцию MyFigure() для отрисовки своей собственной фигуры – с возможностью изменения ширины, высоты и цвета.

Прототип функции должен выглядеть следующим образом:

```
void MyFigure(HDC hdc, int cx, int cy, int sizeX, int sizeY, COLORREF color);
```

Ваша собственная фигура должна быть отрисована при помощи Polyline() или Polygon(). Должна содержать не менее 4 разных точек.

Задача 2.1

Вывести MyFigure() (*созданный в задаче 2.0*) в горизонтальный ряд (с изменением ширины и с изменением цвета). Какой цвет менять – на ваш выбор (RED, GREEN, BLUE) .

Задача 2.2

Вывести MyFigure() (*созданный в задаче 2.0*) в вертикальный ряд (с изменением высоты и с изменением цвета – отличного от изменения по горизонтали). Какой цвет менять – на ваш выбор (RED, GREEN, BLUE) .

Задача 2.3

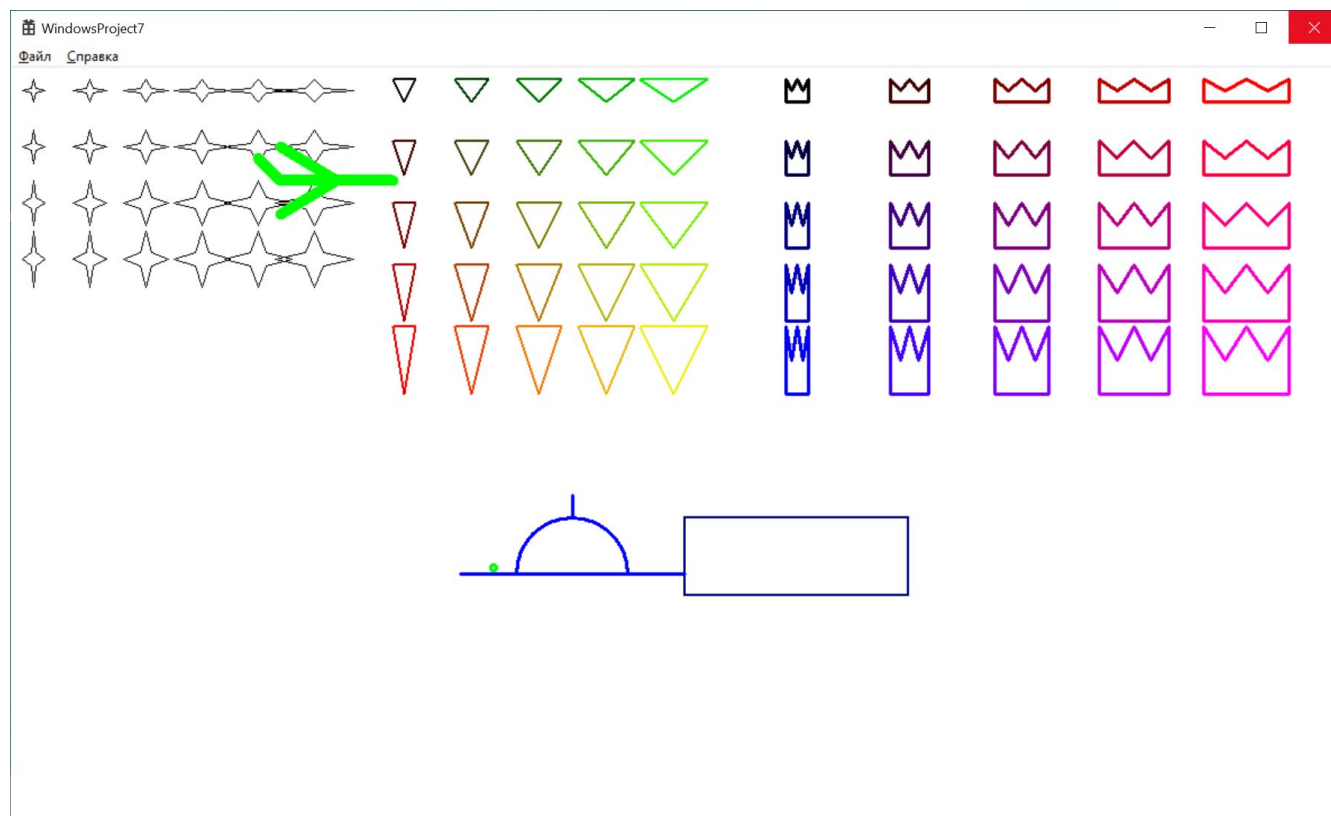
Вывести MyFigure() (*созданный в задаче 2.0*) в диагональный ряд (с одновременным изменением ширины, высоты и изменением двух цветов).

Задача 2.4

Вывести MyFigure() (*созданный в задаче 2.0*) в виде блока (несколько рядов по несколько корон).

Задача 3

В Игре 1 небо заполнить 3-5 блоками разных фигур (звезды, короны, MyFigure() и т.п.).



Домашнее задание

- Доделать задачи 1-3, которые не успели сделать в классе.

Задача 4**. Нарисовать при помощи `MulImage()` узор в виде пирамиды:

