Лабораторная работа №2 «Двумерные примитивы»

Для работы с примитивами необходимо знать координаты точек на экране. Вся область экрана имеет координаты от -1 до 1 по оси X и по оси У. Точка (0,0) расположена в центре экрана. Левый нижний угол имеет координаты (- 1 -1), правый верхний угол — (l , 1).

Все что имеет координаты, по модулю превосходящие 1, будет невидимо на экране.

Для рисования примитивов используются командные скобки

```
glBegin(<тип примитива>);
glEnd;
```

Между командными скобками указываются вершины, задаваемые командой `glVertex2f(x,y).`

Таким образом, для рисования 5 точек можно использовать следующий код:

```
glBegin(GL_POINTS); // открываем командную скобку
glVertex2f(-1, -1); // левый нижний угол
glVertex2f(-1, 1); // левый верхний угол
glVertex2f(0, 0);
glVertex2f(1, -1);
glVertex2f(1, 1);
glEnd;
```

Этот код необходимо разместить в обработчике onPaint, после очистки буфера кадра, но перед выводом содержимого буфера на экран.

Для задания цвета точек необходимо использовать команду glColor3f(r,g,b) с тремя параметрами, задающими красную, зеленую и синюю цветовую составляющую. Все составляющие изменяются в пределах [0..1].

Для задания размера точек используется команда glPointSize(p), задающая масштабирующий множитель для однопиксельной точки.

Точки на экране выглядят квадратными. Чтобы избавиться от этого эффекта необходимо воспользоваться командой glEnable(GL_POINT_SMOOTH), которая включает режим сглаживания точек. Эту команду необходимо разместить перед командными скобками.

Следующий пример иллюстрирует отрисовку 100 точек случайного цвета со случайными координатами.

```
Randomize;
glBegin(GL_POINTS);
For i:=1 to 100 do begin
  glColor3f(random, random, random);
  glVertex2f(random * 2 - 1, random * 2-1);
end;
glEnd;
```

Для рисования линий необходимо использовать тип примитива GL_LTNES, тогда все вершины, указанные между командными скобками, объединяются в отрезки по 2 штуки.

```
glBegin(GL_LINES);
glVertex2f(-1, 1);
glVertex2f(1, -1);
glVertex2f(-1, -1);
glVertex2f(1, 1);
glEnd;
```

Для увеличения толщины отрезков перед командными скобками укажите ширину линии: glLineWidth (2.5);

Для отрисовки ломаной линии используется примитив CL_LINE_STRIP.

```
glBegin(GL_LINE_STRIP);
glVertex2f(-1, -1);
glVertex2f(-1, 1);
glVertex2f(1, 1);
```

```
glVertex2f(1, -1);
glEnd;
```

Дли рисования замкнутой ломаной линии используется примитив GL_LINE_LOOP.

Линии можно рисовать со штриховкой. Для этого необходимо перед командными скобками разместить команды

```
glLineStipple (1, $F0F0);
glEnable(GL_LINE_STRIPPLE);
```

У функции glLineStipple первый аргумент масштабный множитель, второй аргумент задает шаблон штриховки (побитовым способом). Команда glEnable(GL_LINE_STIPPLE) включает режим штриховки.

Способы штриховки:

```
glLineStipple (1, $0101); // точечный
glLineStipple (1, $00FF); // штрихи
glLineStipple (1, $1C47); // штрих-пунктир
```

Написать программу, демонстрирующую различные виды штриховки.

Примитив треугольник задается с помощью параметра GL_TRIANGLES. Тогда вершины, перечисленные между командными скобками по 3 штуки объединяются в треугольники.

```
glBegin(GL_TRIANGLES);
glVertex2f(-1, -1);
glVertex2f(-1, 1);
glVertex2f(1, 0);
glEnd;
```

По умолчанию многоугольники рисуются с заливкой. Чтобы изменить тип отображения многоугольника, необходимо использовать команду glPolygonMode, у которой первый параметр отвечает за сторону фигуры, второй — задаст тип отображения. В качестве первого параметра обычно используется CL_FRONT_AND_BACK, второй параметр имеет 3 возможных значения — GL_FILL(заливка), GL_LINE(фигуры), GL_POINT(опорные точки).

```
glPolygonMode(GL_FRONT_AND_BACK, GL_LINE);
glBegin(GL_TRIANGLES);
For i := 0 to 5 do begin
glVertex2f(0, 0);
glVertex2f(0.5 * cos (2 * Pi * i / 6), 0.5 * sin (2 * Pi * i / 6));
glVertex2f(0.5 * (2 * Pi * (i + 1) 0.5 * sin (2 * Pi * (i + 1) / 6));
end;
glEnd;
```

Примитив связные треугольники (GL TRIANGLE STRIP) строит треугольники из 1-2-3 вершины, 2-3-4 вершины, 3-4-5 вершины и т.д.

```
glBegin(GL_TRIANGLE_STRIP) ;
glVertex2f(1, 1);
glVertex2f(-1, 1);
glVertex2f(-1, -1);
glVertex2f(1, -1);
glEnd;
```

Еще один пример:

```
glBegin(GL TRIANGLE_STRIP);
glVertex2f(random * 2 - 1, random * 2 - 1);
For i: = 0 to 9 do begin
glColor3f(random, random, random);
glVertex2f(random * 2 - 1, random * 2 - 1);
glVertex2f(random * 2 - 1, random * 2 - 1);
end;
glEnd;
```

Примитив GL_TRIANGLE_FAN позволяет нарисовать связные треугольники с общей точкой. Общей считается точка, которая стоит в списке вершин первой. Например, используя связные треугольники можно нарисовать правильныq шестиугольник.

```
glBegin(GL TRIANGLE FAN);
```

```
glVertex2f(0, 0); // вершина, общая для всех треугольников
For i: = o to 6 do begin
glColor3f(random, random, random);
glVertex2f(0. 5 * cos (2 * Pi * i / 6), 0. 5 * sin (2 * Pi * i / 6) );
end;
```

Используя связные треугольники с общей точкой можно построить круг, поскольку такого примитива нет. Для этого нужно увеличить количество шагов цикла.

```
glBegin(GL_TRIANGLE_FAN);
glVertex2f(0, 0); // вершина, общая для всех треугольников
For i:= 0 to 100 do begin
glColor3f(random, random, random);
glVertex2f(0. 5 * (2 * Pi * i / 100), 0.5 * sin (2 * Pi * i / 100));
end;
```

Константа GL_QUADS задает примитив, когда перечисляемые вершины берутся по четыре и по ним строятся независимые четырехугольники.

```
glBegin(GL_QUADS);
glColor3f(random, random, random);
glVertex2f(-0.6, 0.2);
glVertex2f(-0.7, 0. 7);
glVertex2f(0.1, 0.65);
glVertex2f(0.25, -0.78);
glColor3f(random, random, random);
glVertex2f(0.3, -0.6);
glVertex2f(0.45, 0.7);
glVertex2f(0.8, 0.65);
glVertex2f(0.9, -0.8);
glEnd;
```

Примитив, задаваемый константой GL_QUAD_STRIP, состоит из связанных четырехугольников Первый четырехугольник формируется из вершин номер один, два, три и четыре. Второй четырехугольник в качестве опорных берет вторую, третью, пятую и четвертую вершины. Для рисования выпуклого многоугольника используется примитив GL_POLYGON.

```
glBegin(GL_POLYGON);
For i:=0 to 6
glVertex2f(0.5 *cos (2 * Pi * i / 6), O. 5 * sin (2 * Pi *i/6));
glEnd;
```