МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное
учреждение высшего образования
«Юго-Западный государственный университет»
(ЮЗГУ)

Кафедра вычислительной техники

УТВЕРЖДАЮ
Проректор по учебной работе
_____ О.Г. Локтионова
«___» _____ 2017 г.

ПОСТРОЕНИЕ СЛОЖНЫХ ИЗОБРАЖЕНИЙ В PROCESSING.

Методические указания по выполнению лабораторной работы
по дисциплине «Инженерная и компьютерная графика»
для студентов специальности 09.03.01 «Информатика
и вычислительная техника»

Курск 2017

УДК 621.37(075)

Составители: М.В. Бобырь, С.А. Кулабухов

Рецензент
Доктор технических наук, профессор *И.В. Зотов*

1. Построение сложных изображений в Processing: методические указания по выполнению лабораторной работы по дисциплине «Инженерная и компьютерная графика» / Юго-Зап. гос. ун-т; сост.: М.В. Бобырь, С.А. Кулабухов. – Курск, 2017. – 13 с.: ил. 6, табл.1. – Библиогр.: с.13.

Рассмотрены базовые понятия компьютерной графики на основе программирования в Processing. В учебно-методической работе содержатся задания для выполнения практических работ.

Методические указания соответствуют требованиям программы дисциплины «Компьютерная графика».

Предназначены для студентов специальности 09.03.01 «Информатика и вычислительная техника» дневной и заочной форм обучения.

Текст печатается в авторской редакции

Подписано в печать 10.11.17 Формат 60х84 1/16.
Усл.печ. л. 0,7 Уч.-изд. л. 0,6 Тираж 50 экз. Заказ 1838 Бесплатно.
Юго-Западный государственный университет.
305040, г. Курск, ул. 50 лет Октября, 94.

ПОСТРОЕНИЕ СЛОЖНЫХ ИЗОБРАЖЕНИЙ В PROCESSING.

1. Цель работы

Понять основы компьютерной графики и сложные графические объекты в среде разработки Processing.

2. Общие сведения о Processing

Processing — открытый язык программирования, основанный на Java. Представляет собой лёгкий и быстрый инструментарий для людей, которые хотят программировать изображения, анимацию и интерфейсы.

Используется студентами, художниками, дизайнерами, исследователями и любителями, для изучения, прототипирования и производства. Он создан для изучения основ компьютерного программирования в визуальном контексте и служит альбомным программным обеспечением (имеется в виду то, что каждый *.pde файл визуальной оболочки Processing’а представляет собой отдельное изображение или анимацию, и т. д.) и профессиональным производственным инструментом.

Интерфейс Processing, невероятно минималистичен. Задумали Processing как, своего рода, блокнот для набросков, в котором все начинается практически с чистого листа. Хотя Processing значительно упрощает процесс программирования, он никогда не был предназначен для упрощения творческого процесса. Язык лишен различных графических фильтров и

эффектов, которые можно найти, например, в редакторе Adobe Photoshop. Все это в соответствии с его концепцией.

Processing является полностью автономным исполняемым приложением. Вы просто запускаете его двойным щелчком и начинаете писать код. Большинство других сред программирования требуют «поколдовать» с системными настройками и предпочтениями для того, чтобы начать работу. Кроме того, многие из этих сред разработки темпераментны и легко могут перестать работать, если файлы будут случайно перемещены или пересохранены. Кроме того, сама по себе среды программирования (даже не использующиеся в ней языки программирования) могут быть крайне сложными для освоения. Среда разработки Processing, напротив, проста и интуитивно понятна при использовании и не добавляет сложности к написанию кода.

Processing не требует каких-то специальных базовых знаний. Одной из самых интересных частей Processing является его поддержка 3D. Processing использует OpenGL, являющийся индустриальным стандартом в 3Dпрограммном обеспечении, которое использует аппаратные средства компьютера для ускорения, обеспечивая очень надежную производительность. Хотя 3D обычно требует сложного программирования и математики, Processing значительно упрощает процесс.

3. Дополнительные возможности Processing.

Переменные в Processing.

```
int x; // Объявить переменную x типа int
x = 7; // Присвоить значение переменной x
```

Можно иначе:

```
int x = 7; // Объявить переменную x типа int и присвоить ей значение
```

При создании программ в Processing можно вместо численных значений подставлять специальные переменные, например:

```
size(300, 200);
smooth();
line(0, 0, width, height); // Линия от (0,0) до (300,200)
line(width, 0, 0, height); // Линия от (300,0) до (0,200)
ellipse(width/2, height/2, 50, 50);
```

Переменные можно использовать при создании различных квадратов, прямоугольников и т.п.

```
size(480, 120);
int x = 25;
int h = 20;
int y = 25;
rect(x, y, 300, h); // Верхний
x = x + 100;
rect(x, y + h, 300, h); // Средний
x = x - 250;
rect(x, y + h*2, 300, h); // Нижний
```

При математическом расчете используется приоритет математических операций:

```
int x = 4 + (4 * 5); // Присвоение значения 24 переменной x
int x = (4 + 4) * 5; // Присвоение значения 40 переменной x
```

Для прибавления или вычитания 1 из переменной используются операторы ++ и -- :

```
x++; // Выражение эквивалентно x = x + 1
y--; // Выражение эквивалентно y = y - 1
x += 10; // Выражение эквивалентно x = x + 10
y -= 15; // Выражение эквивалентно y = y – 15
```

В Processing удобно пользоваться циклами:

Пример листинга программного кода без использования цикла

```
size(480, 120);
smooth();
strokeWeight(8);
line(20, 40, 80, 80);
line(80, 40, 140, 80);
line(140, 40, 200, 80);
line(200, 40, 260, 80);
line(260, 40, 320, 80);
line(320, 40, 380, 80);
line(380, 40, 440, 80);
```

Листинг программного кода с использованием цикла **for**

```
size(480, 120);
smooth();
strokeWeight(8);

for (int i = 20; i < 400; i += 60)
{
    line(i, 40, i + 60, 80);
}
```

Используя цикл **for** можно получить и другие рисунки:

```
size(480, 120);
smooth();
```

```
strokeWeight(2);
for (int i = 20; i < 400; i += 8)
{
	line(i, 40, i + 60, 80);
}
size(480, 120);
smooth();
strokeWeight(2);
for (int i = 20; i < 400; i += 20)
{
	line(i, 0, i + i/2, 80);
}
size(480, 120);
smooth();
strokeWeight(2);
for (int i = 20; i < 400; i += 20)
{
	line(i, 0, i + i/2, 80);
	line(i + i/2, 80, i*1.2, 120);
}
```

Также в Processing можно пользоваться вложенными циклами **for**:

```
size(480, 120);
background(0);
smooth();
noStroke();
for (int y = 0; y <= height; y += 40)
{
	for (int x = 0; x <= width; x += 40)
	{
		fill(255, 140);
		ellipse(x, y, 40, 40);
	}
}
size(480, 120);
```

```
background(0);
smooth();
noStroke();
for (int y = 0; y < height+45; y += 40)
{
        fill(255, 140);
        ellipse(0, y, 40, 40);
}
for (int x = 0; x < width+45; x += 40)
{
        fill(255, 140);
        ellipse(x, 0, 40, 40);
}

size(480, 120);
background(0);
smooth();
fill(255);
stroke(102);
for (int y = 20; y <= height-20; y += 10)
{
        for (int x = 20; x <= width-20; x += 10)
        {
        ellipse(x, y, 4, 4);
        line(x, y, 240, 60);
        }
}
size(480, 120);
background(0);
smooth();
for (int y = 32; y <= height; y += 8) {
        for (int x = 12; x <= width; x += 15) {
                ellipse(x + y, y, 16 - y/10.0, 16 - y/10.0);
        }
}
```

4. Задания для самостоятельного выполнения

Запустите программу Processing. В текстовом редакторе наберите следующее:

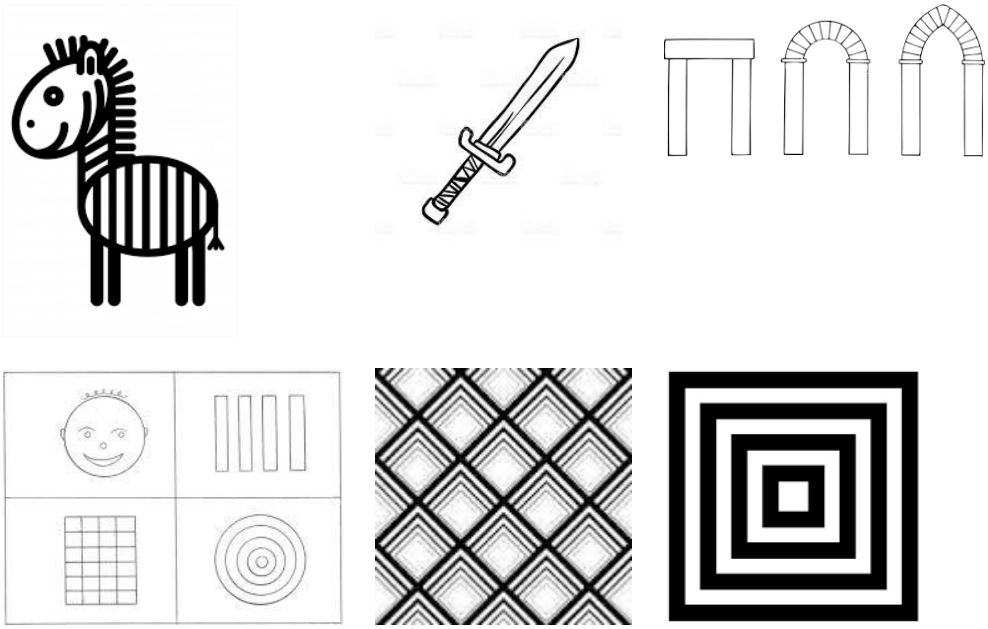
```
size(720, 480);
smooth();
strokeWeight(2);
background(204);
ellipseMode(RADIUS);

// Шея
stroke(102); // Сделать линии серыми
line(266, 257, 266, 162); // Левая
line(276, 257, 276, 162); // Центральная
line(286, 257, 286, 162); // Правая

// Антенны
line(276, 155, 246, 112); // Малая
line(276, 155, 306, 56); // Большая
line(276, 155, 342, 170); // Средняя

// Туловище
noStroke(); // Невидимые линии
fill(102); // Окрашивать серым
ellipse(264, 377, 33, 33); // Шар
fill(0); // Окрашивать черным
rect(219, 257, 90, 120); // Собственно туловище
fill(102); // Окрашивать серым
rect(219, 274, 90, 6); // Серая полоса

// Голова
fill(0); // Окрашивать черным
ellipse(276, 155, 45, 45); // Голова
fill(255); // Окрашивать белым
ellipse(288, 150, 14, 14); // Большой глаз
```

fill(0); // Окрашивать черным
ellipse(288, 150, 3, 3); // Зрачок
fill(153); // Окрашивать в светло-серый
ellipse(263, 148, 5, 5); // Маленький глаз 1
ellipse(296, 130, 4, 4); // Маленький глаз 2
ellipse(305, 162, 3, 3); // Маленький глаз 3

Получится изображение робота.

Задание 1

Нарисуйте следующие картины:



Задание 2

Используя модифицированный код робота ниже построить изображение согласно варианту, представленному в таблице 1.

```
int x = 60; // x-координата
int y = 420; // y-координата
int bodyHeight = 110; // Длина туловища
int neckHeight = 140; // Длина шеи
int radius = 45; // Радиус головы
int ny = y - bodyHeight - neckHeight - radius; // Шея Y
```

```
size(170, 480);
smooth();
strokeWeight(2);
background(204);
ellipseMode(RADIUS);

// Шея
stroke(102);
line(x+2, y-bodyHeight, x+2, ny); line(x+12,
y-bodyHeight, x+12, ny); line(x+22, y-bodyHeight,x+22, ny);

// Антенны
line(x+12, ny, x-18, ny-43);
line(x+12, ny, x+42, ny-99);
line(x+12, ny, x+78, ny+15);

// Туловище
noStroke();
fill(102);
ellipse(x, y-33, 33, 33);
fill(0);
rect(x-45, y-bodyHeight, 90, bodyHeight-33);
fill(102);
rect(x-45, y-bodyHeight+17, 90, 6);

// Голова
fill(0);
ellipse(x+12, ny, radius, radius);
fill(255);
ellipse(x+24, ny-6, 14, 14);
fill(0);
ellipse(x+24, ny-6, 3, 3);
fill(153);
ellipse(x, ny-8, 5, 5);
ellipse(x+30, ny-26, 4, 4);
ellipse(x+41, ny+6, 3, 3);
```

Таблица 1

№ варианта	Y	bodyHeight	neckHeight
1	454	233	134
2	482	84	147
3	411	171	133
4	458	172	117
5	423	200	122
6	408	236	97
7	492	139	37
8	378	95	124
9	381	227	12
10	360	85	145
11	378	109	107
12	397	107	69
13	461	82	109
14	412	85	136
15	415	106	123
16	431	102	58
17	492	217	43
18	373	159	33
19	464	129	44
20	401	92	10
21	350	92	42
22	428	193	104
23	475	215	75
24	403	107	50
25	481	105	23

5. Контрольные вопросы

1. Расскажите о среде разработки Processing?
2. Как устроены операторы if? Каков их синтаксис?
3. Расскажите о циклах их назначении?
4. Как строятся сложные объекты в Processing?
5. Как циклы помогают в создании сложных объектов?

6. Содержание отчёта

Отчёт должен содержать:

1) титульный лист;

2) наименование работы и цель исследований;

3) описание хода выполнения задания;

4) изображение построенной фигуры и код программы для ее построения.

7. Библиографический список

1. Кейси Риз и Бен Фрай «Учимся программировать вместе с Processing» перевод с английского Издательская группа BHV, 2012. - 194 с., ил.