1. Задача. Рассмотрим создание более сложных компонентов для построения графиков и диаграмм. Класс BarChart позволяет создавать гистограммы.

Гистограмма рисуется на плоскости и основана на значениях двух параметров, откладываемых соответственно на двух осях. Числовая ось задаётся объектом класса NumberAxis, ось из уникальных строковых категорий объектом класса CategoryAxis. Для построения гистограммы необходимо создать две оси любого типа. Созданные объекты-оси передаются в конструктор BarChart. Объект класса BarChart отображает гистограмму набора пар значений. Набор пар – это объект класса XYChart.Series, каждая пара – объект класса XYChart.Data. Первый параметр конструктора XYChart.Data – значение по горизонтальной оси и должно соответствовать первой оси для BarChart. Второй параметр – значение по вертикальной оси, второй для BarChart.

Приведём код создания гистограммы:

private Group createBarChart() {

Group groupChart = new Group();

String austria = "Austria", brazil = "Brazil", france = "France",

italy = "Italy", usa = "USA";

final CategoryAxis xAxis = new CategoryAxis();

final NumberAxis yAxis = new NumberAxis();

final BarChart<String, Number> bc

= new BarChart<String, Number>(xAxis, yAxis);

bc.setTitle("Country weather");

xAxis.setLabel("Country");

yAxis.setLabel("Temperature");

XYChart.Series series1 = new XYChart.Series();

series1.setName("June");

series1.getData().add(new XYChart.Data(austria, 16.4));

series1.getData().add(new XYChart.Data(brazil, 22.6));

series1.getData().add(new XYChart.Data(france, 19.2));

series1.getData().add(new XYChart.Data(italy, 20.5));

series1.getData().add(new XYChart.Data(usa, 22.1));

XYChart.Series series2 = new XYChart.Series();

series2.setName("July");

series2.getData().add(new XYChart.Data(austria, 17.6));

series2.getData().add(new XYChart.Data(brazil, 21.8));

series2.getData().add(new XYChart.Data(france, 21.1));

series2.getData().add(new XYChart.Data(italy, 22.5));

series2.getData().add(new XYChart.Data(usa, 24.6));

XYChart.Series series3 = new XYChart.Series();

series3.setName("August");

series3.getData().add(new XYChart.Data(austria, 17.2));

series3.getData().add(new XYChart.Data(brazil, 22.1));

series3.getData().add(new XYChart.Data(france, 20.4));

series3.getData().add(new XYChart.Data(italy, 22.3));

series3.getData().add(new XYChart.Data(usa, 23.8));

bc.getData().addAll(series1, series2, series3);

groupChart.getChildren().add(bc);

return groupChart;}

1. Задача. Класс LineChart обеспечивает возможность рисования графиков кривых на координатной плоскости.

Для построения графиков также как и для построения гистограмм необходимо создать две оси и набор пар соответствующих значений. По умолчанию график рисуется в виде точек. Чтобы получить сплошную кривую, необходимо использовать метод LineChart – setCreateSymbols(boolean value) с параметром false. Полный код примера с классом для формирования пар значений для LineChart из точек графика математической функции выглядит так:

private Group createLineChart(){

Group groupChart = new Group();

Func f = new Func(-5, 5, 600);

NumberAxis x = new NumberAxis();

NumberAxis y = new NumberAxis();

LineChart<Number, Number> numberLineChart = new LineChart<>(x, y);

numberLineChart.setCreateSymbols(false);

numberLineChart.setLayoutX(400);

numberLineChart.setLayoutY(60);

numberLineChart.setMinHeight(600);

numberLineChart.setMinWidth(600);

numberLineChart.getData().add(f.getSeries());

groupChart.getChildren().add(numberLineChart);

return groupChart;}

class Func {

private double xmin, xmax;

private int step;

public Func(double min, double max, int numStep){

xmin = min;

xmax = max;

step = numStep; }

private double f(double x){

return (3 - 4\*x)/(x\*x+1); }

public String getName(){

return "(3 - 4\*x)/(x\*x+1)"; }

public XYChart.Series getSeries(){

ObservableList<XYChart.Data> datas = FXCollections.*observableArrayList*();

double x0 = xmin;

double h = (xmax- x0) / step;

for (int i = 0; i < step; i++) {

datas.add(new XYChart.Data(x0 + h \* i, f(x0 + h \* i))); }

XYChart.Series series = new XYChart.Series();

series.setName(getName());

series.setData(datas);

return series; }}

1. Задача. Класс PieChart предназначен для визуализации наборов чисел в виде круговой диаграммы.

Круговая диаграмма PieChart состоит из секторов, соответствующих элементам списка ObservableList<PieChart.Data>. Каждый элемент этого списка – объект класса PieChart.Data – содержит пару значений: строка с наименованием категории и число, характеризующее эту категорию. Размер сектора определяется долей числа среди чисел всего списка. Все графики и диаграммы снабжаются легендами с описанием обозначений. Легенда диаграммы PieChart поясняет сектор какого цвета соответствует заданной категории. Метод setLegendSide(Side value) позволяет управлять расположением легенды относительно диаграммы: внизу, вверху, слева или справа. Для этого используются константы, определённые в классе Side.

Код создания диаграммы выглядит так:

private Group createPieChart() {

Group groupChart = new Group();

ObservableList<PieChart.Data> pieChartData

= FXCollections.*observableArrayList*(

new PieChart.Data("Apple", 73.8),

new PieChart.Data("Samsung", 69.4),

new PieChart.Data("Huawei", 56.2),

new PieChart.Data("Xiaomi", 32.8),

new PieChart.Data("OPPO", 30.6),

new PieChart.Data("Others", 106));

PieChart chart = new PieChart(pieChartData);

chart.setTitle("Smartphone sales");

chart.setLegendSide(Side.*LEFT*);

Label caption = new Label("");

caption.setTextFill(Color.*DARKBLUE*);

caption.setStyle("-fx-font: 16 arial;");

groupChart.getChildren().add(caption);

for (PieChart.Data data : chart.getData()) {

data.getNode().addEventHandler(MouseEvent.*MOUSE\_PRESSED*, (MouseEvent e) -> {

caption.setTranslateX(e.getSceneX()-caption.getWidth()/2);

caption.setTranslateY(e.getSceneY()-caption.getHeight());

caption.setText(data.getPieValue()+"");

caption.toFront(); }); }

groupChart.getChildren().add(chart);

return groupChart;}

\*Задача

Создайте однооконное приложение для решения следующей задачи. Задан дневник температуры за март: {-2, -5, -2, -4, 3, -6, -2, -1, 5, 1, 1, 0, -1, 0, 3, -1, 2, 5, 2, 4, 4, 0, 6, 1, 4, 6, -1, 2, 4, 7, 11}. Изобразите график температуры по дням и соотношение между днями с положительной, отрицательной и нулевой температурой в виде круговой диаграммы.

В поле ответа на задание введите количество дней, соответствующее самому большому сектору получившейся диаграммы.

Дополнительно: реализуйте возможность изменения дневника температуры. Это можно сделать разными способами. Например, создать поле ввода температуры для каждого дня месяца. Второй вариант – вводить все числа в одной строке, которую затем преобразовать в массив чисел методами классов String и Integer, в этом случае, не забудьте проверить корректность ввода данных.