**Metody numeryczne**

Lista 7

Katarzyna Korsak 229707

Piątek 730

**Zadanie 1**

* 1. **Rozwiązanie zadania**

Na rozwiązanie zadania 1 składa się klasa *Log*, która przyjmuje atrybuty czas końcowy *tend*, liczba próbek *n* i dowolną podstawę logarytmu *a*. Klasa *LogFX* odpowiada za rysowanie wykresu funkcji logarytmu z x o dowolnej podstawie.

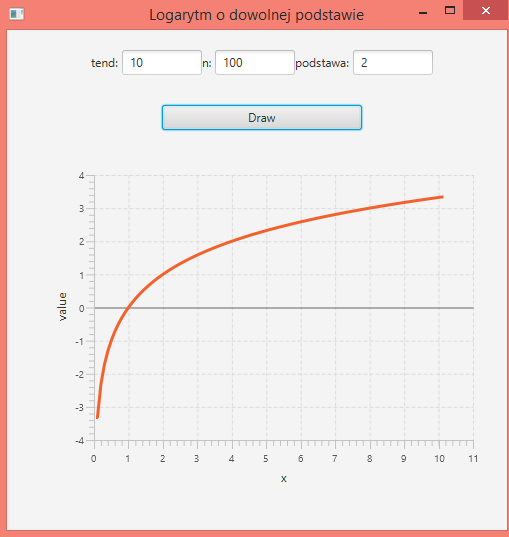
**Klasa Log**

**import** javafx.scene.chart.XYChart;  
  
**public class** Log {  
  
 **private double tend**;  
 **private double a**;  
 **private int n**;  
  
 **public** Log(**double** tend, **double** a, **int** n){  
 **this**.**tend**=tend;  
 **this**.**a**=a;  
 **this**.**n**=n;  
 }  
  
 **public** XYChart.Series getValues(){  
 XYChart.Series values = **new** XYChart.Series();  
  
 **double** dt=**tend**/(**n**-1);  
 **double** x=0.1;  
  
 **for**(**int** i=0;i<**n**;i++){  
 values.getData().add(**new** XYChart.Data(x,Math.*log*(x)/Math.*log*(**a**)));  
 x+=dt;  
 }  
 **return** values;  
 }  
  
  
}

**Klasa LogFX**

**import** javafx.application.Application;  
**import** javafx.geometry.Insets;  
**import** javafx.geometry.Pos;  
**import** javafx.scene.Scene;  
**import** javafx.scene.chart.LineChart;  
**import** javafx.scene.chart.NumberAxis;  
**import** javafx.scene.chart.XYChart;  
**import** javafx.scene.layout.HBox;  
**import** javafx.scene.layout.VBox;  
**import** javafx.stage.Stage;  
**import** javafx.scene.control.Button;  
**import** javafx.scene.control.TextField;  
**import** javafx.scene.control.Label;  
  
**import** java.awt.\*;  
  
**public class** LogFX **extends** Application {  
  
 **private** TextField **tendField**;  
 **private** TextField **aField**;  
 **private** TextField **nField**;  
  
 Button **btnDraw**;  
  
 LineChart<Number,Number> **figure**;  
  
  
 @Override  
 **public void** start(Stage primaryStage) **throws** Exception {  
  
 primaryStage.setTitle(**"Logarytm o dowolnej podstawie"**);  
  
  
 VBox layout = **new** VBox();  
 layout.setSpacing(30);  
 layout.setPadding(**new** Insets(20,20,30,30));  
  
 Label t = **new** Label(**"tend: "**);  
 Label n = **new** Label(**"n: "**);  
 Label a = **new** Label(**"podstawa: "**);  
  
 **tendField** = **new** TextField();  
 **tendField**.setText(**"10"**);  
 **tendField**.setPrefWidth(80);  
 **aField** = **new** TextField();  
 **aField**.setText(**"2"**);  
 **aField**.setPrefWidth(80);  
 **nField** = **new** TextField();  
 **nField**.setText(**"100"**);  
 **nField**.setPrefWidth(80);  
  
 HBox hbox = **new** HBox(t,**tendField**,n,**nField**,a,**aField**);hbox.setAlignment(Pos.***CENTER***);  
  
 layout.getChildren().add(hbox);  
  
 **btnDraw** = **new** Button(**"Draw"**);  
 **btnDraw**.setPrefWidth(200);  
 **btnDraw**.setOnAction(e->{  
 **double** x = Double.*parseDouble*(**tendField**.getText());  
 **int** nn = Integer.*parseInt*(**nField**.getText());  
 **double** aa = Double.*parseDouble*(**aField**.getText());  
 Log log = **new** Log(x,aa,nn);  
  
 **figure**.setCreateSymbols(**false**);  
 **figure**.getData().clear();  
 XYChart.Series data = log.getValues();  
 data.setName(**"Test"**);  
 **figure**.setLegendVisible(**false**);  
 **figure**.getData().add(data);  
  
 });  
  
 HBox hbox2 = **new** HBox(**btnDraw**);  
 hbox2.setAlignment(Pos.***CENTER***);  
 layout.getChildren().add(hbox2);  
  
 NumberAxis x = **new** NumberAxis();  
 x.setLabel(**"x"**);  
 NumberAxis y = **new** NumberAxis();  
 y.setLabel(**"value"**);  
 **figure** = **new** LineChart<Number, Number>(x,y);  
  
 layout.getChildren().add(**figure**);  
  
  
 Scene scene = **new** Scene(layout,500,500);  
  
 primaryStage.setScene(scene);  
 primaryStage.show();  
 }  
  
 **public static void** main(String[] args) {  
 *launch*(args);  
 }  
}

* 1. **Wyniki**



Wykres 1. Wykres funkcji logarytmicznej o podstawie a=2 z x

* 1. **Wnioski**

Z ulepszeń powyższej funkcji powinno wprowadzić się blokowanie funkcji wpisywanie w pole podstawy liczby ujemnej lub też wyrzucanie okienka z odpowiednim komunikatem. Można by też wprowadzić opcję porównywania kolejnych wykresów dla różnych podstaw, np. poprzez usunięcie metody *figure.getData().clear();* z ciała *btnDraw*.

**Zadanie 2**

**2.1 Rozwiązanie zadania**

Na rozwiązanie zadania 2 składają się klasy *Wahadlo* i *WahadloFX.* Pierwsza z nich za atrybuty przyjmuje parametry wejściowe funkcji obliczanej metodą Eulera-Richardsona oraz czas początkowy i końcowy. W metodzie *GetValues()* tworzone są trzy serie wartości odpowiednio dla wykresów *thety* od *czasu*, *omegi* od *czasu* i *thety* od *omegi*. Równania wyliczające wartości kątowe i prędkościowe oraz ich odpowiedniki połówkowe zaczerpnięte zostały z zadania 2 z listy 6. Pierwsze dwa wykresy tworzone są przy pomocy obiektów *LineChart*, trzeci zaś przy pomocy *ScatterChart*.

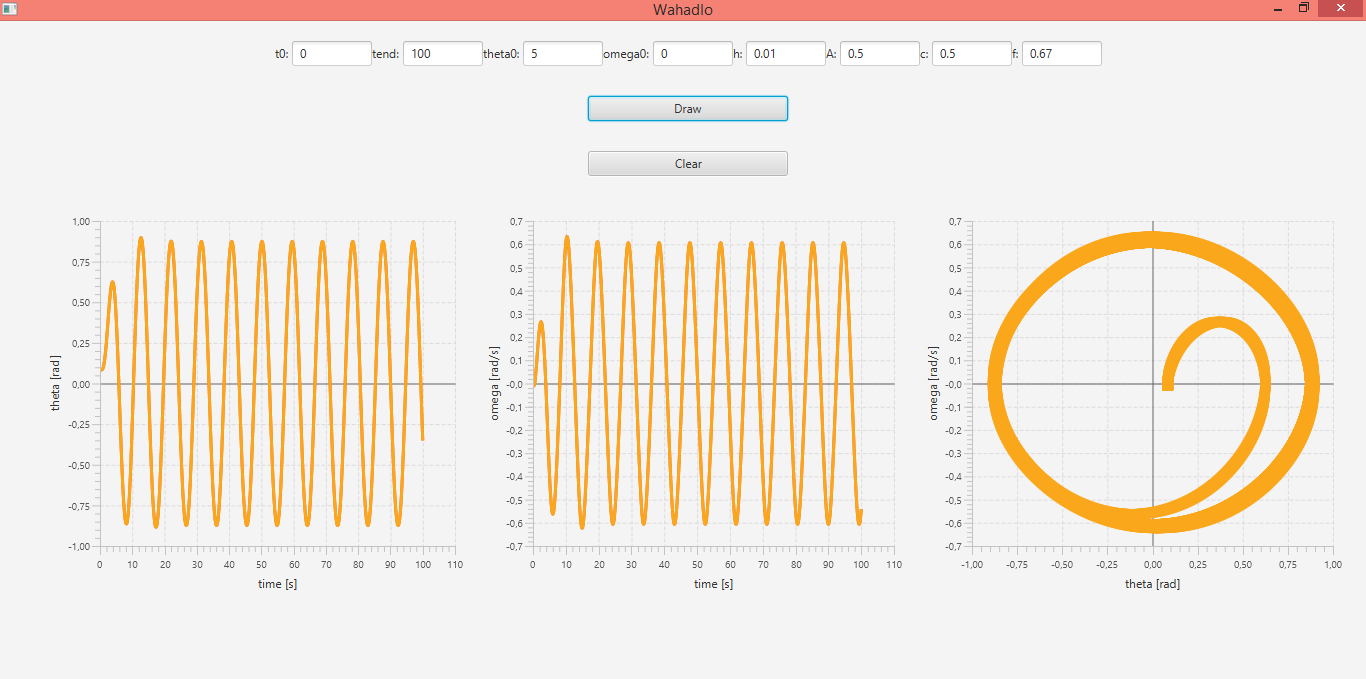
**Klasa Wahadlo**

**import** javafx.scene.chart.XYChart;  
  
**import** java.util.ArrayList;  
  
**public class** Wahadlo {  
  
 **private double A**;  
 **private double f**;  
 **private double c**;  
 **private double theta0**;  
 **private double t0**;  
 **private double h**;  
 **private double tend**;  
 **private double omega0**;  
  
  
 **public** Wahadlo(**double** A, **double** f, **double** c, **double** theta0, **double** t0, **double** h, **double** tend, **double** omega0) {  
 **this**.**A** = A;  
 **this**.**f** = f;  
 **this**.**c** = c;  
 **this**.**theta0** = theta0;  
 **this**.**t0** = t0;  
 **this**.**h** = h;  
 **this**.**tend** = tend;  
 **this**.**omega0** = omega0;  
 }  
  
 **public** ArrayList<XYChart.Series> getValues() {  
 XYChart.Series values = **new** XYChart.Series();  
 XYChart.Series values2 = **new** XYChart.Series();  
 XYChart.Series values3 = **new** XYChart.Series();  
  
 ArrayList<XYChart.Series> daneOut = **new** ArrayList();  
  
 **double** t = **t0**;  
 **double** theta = **theta0**;  
 **double** omega = **omega0**;  
 **double** n = Math.*floor*((**tend** - **t0**) / **h**);  
  
 **for** (**int** i = 1; i < n; i++) {  
 **double** thetaOld = theta;  
  
 **double** tMid = t + **h** / 2;  
 **double** thetaMid = theta + **h** / 2 \* omega;  
 **double** omegaMid = omega - Math.*sin*(thetaOld) \* **h** / 2 - **c** \* omega \* **h** / 2 + **A** \* Math.*sin*(t \* **f**) \* **h** / 2;  
 theta = theta + **h** \* omegaMid;  
  
 omega = omega - Math.*sin*(thetaMid) \* **h** - **c** \* omegaMid \* **h** + **A** \* Math.*sin*(tMid \* **f**) \* **h**;  
  
 t = t + **h**;  
  
 values.getData().add(**new** XYChart.Data(t, theta));  
  
 values2.getData().add(**new** XYChart.Data<>(t, omega));  
  
 values3.getData().add(**new** XYChart.Data<>(theta, omega));  
  
 }  
 daneOut.add(values);  
 daneOut.add(values2);  
 daneOut.add(values3);  
  
 **return** daneOut;  
  
 }  
  
}

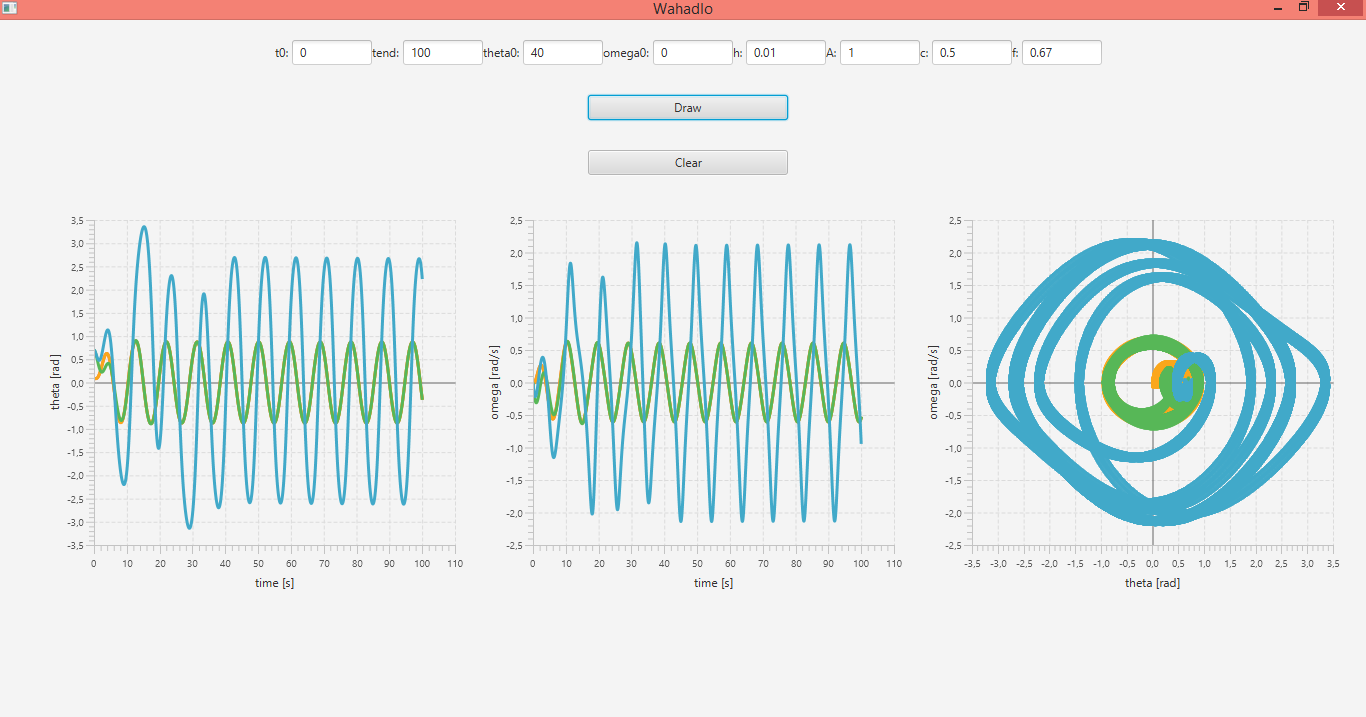
**Klasa WahadloFX**

**import** javafx.application.Application;  
**import** javafx.geometry.Insets;  
**import** javafx.geometry.Pos;  
**import** javafx.scene.Scene;  
**import** javafx.scene.chart.LineChart;  
**import** javafx.scene.chart.NumberAxis;  
**import** javafx.scene.chart.ScatterChart;  
**import** javafx.scene.chart.XYChart;  
**import** javafx.scene.control.Button;  
**import** javafx.scene.control.Label;  
**import** javafx.scene.control.TextField;  
**import** javafx.scene.layout.HBox;  
**import** javafx.scene.layout.VBox;  
**import** javafx.stage.Stage;  
  
**import** java.util.ArrayList;  
  
**public class** WahadloFX **extends** Application {  
  
  
 **private** TextField **t0Field**;  
 **private** TextField **tendField**;  
 **private** TextField **hField**;  
 **private** TextField **aField**;  
 **private** TextField **cField**;  
 **private** TextField **fField**;  
 **private** TextField **omega0Field**;  
 **private** TextField **theta0Field**;  
  
 Button **btnDraw**;  
 Button **btnClear**;  
  
 LineChart<Number, Number> **figure**;  
  
 LineChart<Number, Number> **figure2**;  
  
 ScatterChart<Number, Number> **figure3**;  
  
 @Override  
 **public void** start(Stage primaryStage) **throws** Exception {  
  
 primaryStage.setTitle(**"Wahadlo"**);  
  
 VBox layout = **new** VBox();  
 layout.setSpacing(30);  
 layout.setPadding(**new** Insets(20, 20, 30, 30));  
  
 Label tend = **new** Label(**"tend: "**);  
 Label f = **new** Label(**"f: "**);  
 Label a = **new** Label(**"A: "**);  
 Label h = **new** Label(**"h: "**);  
 Label c = **new** Label(**"c: "**);  
 Label omega0 = **new** Label(**"omega0: "**);  
 Label theta0 = **new** Label(**"theta0: "**);  
 Label t0 = **new** Label(**"t0: "**);  
  
 **tendField** = **new** TextField();  
 **tendField**.setText(**"10"**);  
 **tendField**.setPrefWidth(80);  
 **aField** = **new** TextField();  
 **aField**.setText(**"0.5"**);  
 **aField**.setPrefWidth(80);  
 **fField** = **new** TextField();  
 **fField**.setText(**"0.67"**);  
 **fField**.setPrefWidth(80);  
 **t0Field** = **new** TextField();  
 **t0Field**.setText(**"0"**);  
 **t0Field**.setPrefWidth(80);  
 **hField** = **new** TextField();  
 **hField**.setText(**"0.01"**);  
 **hField**.setPrefWidth(80);  
 **cField** = **new** TextField();  
 **cField**.setText(**"0.5"**);  
 **cField**.setPrefWidth(80);  
 **omega0Field** = **new** TextField();  
 **omega0Field**.setText(**"0"**);  
 **omega0Field**.setPrefWidth(80);  
 **theta0Field** = **new** TextField();  
 **theta0Field**.setText(**"5"**);  
 **theta0Field**.setPrefWidth(80);  
  
 HBox hbox = **new** HBox(t0, **t0Field**, tend, **tendField**, theta0, **theta0Field**, omega0, **omega0Field**, h, **hField**, a, **aField**, c, **cField**, f, **fField**);  
  
 hbox.setAlignment(Pos.***CENTER***);  
  
 layout.getChildren().add(hbox);  
  
 **btnDraw** = **new** Button(**"Draw"**);  
 **btnDraw**.setPrefWidth(200);  
 **btnDraw**.setOnAction(e -> {  
 **double** tstop = Double.*parseDouble*(**tendField**.getText());  
 **double** ff = Double.*parseDouble*(**fField**.getText());  
 **double** aa = Double.*parseDouble*(**aField**.getText());  
 **double** cc = Double.*parseDouble*(**cField**.getText());  
 **double** theta = Double.*parseDouble*(**theta0Field**.getText())\*Math.***PI***/180;  
 **double** tstart = Double.*parseDouble*(**t0Field**.getText());  
 **double** hh = Double.*parseDouble*(**hField**.getText());  
 **double** omega = Double.*parseDouble*(**omega0Field**.getText());  
  
 Wahadlo wahadlo = **new** Wahadlo(aa, ff, cc, theta, tstart, hh, tstop, omega);  
  
 ArrayList<XYChart.Series> data = wahadlo.getValues();  
 ArrayList<ScatterChart.Series> data2 = wahadlo.getValues();  
  
 **figure**.setCreateSymbols(**false**);  
 **figure**.setLegendVisible(**false**);  
 **figure**.getData().add(data.get(0));  
  
 **figure2**.setCreateSymbols(**false**);  
 **figure2**.setLegendVisible(**false**);  
 **figure2**.getData().add(data.get(1));  
  
 **figure3**.setLegendVisible(**false**);  
 **figure3**.getData().add(data2.get(2));  
 });  
  
 **btnClear** = **new** Button(**"Clear"**);  
 **btnClear**.setPrefWidth(200);  
 **btnClear**.setOnAction(e -> {  
  
 **figure**.getData().clear();  
 **figure2**.getData().clear();  
 **figure3**.getData().clear();  
  
 });  
  
 HBox hbox2 = **new** HBox(**btnDraw**);  
 hbox2.setAlignment(Pos.***CENTER***);  
 layout.getChildren().add(hbox2);  
  
 HBox hbox3 = **new** HBox(**btnClear**);  
 hbox3.setAlignment(Pos.***CENTER***);  
 layout.getChildren().add(hbox3);  
  
 NumberAxis x = **new** NumberAxis();  
 x.setLabel(**"time [s]"**);  
 NumberAxis y = **new** NumberAxis();  
 y.setLabel(**"theta [rad]"**);  
  
 **figure** = **new** LineChart<Number, Number>(x, y);  
  
 NumberAxis x2 = **new** NumberAxis();  
 x2.setLabel(**"time [s]"**);  
 NumberAxis y2 = **new** NumberAxis();  
 y2.setLabel(**"omega [rad/s]"**);  
  
 **figure2** = **new** LineChart<Number, Number>(x2, y2);  
  
 NumberAxis x3 = **new** NumberAxis();  
 x3.setLabel(**"theta [rad]"**);  
 NumberAxis y3 = **new** NumberAxis();  
 y3.setLabel(**"omega [rad/s]"**);  
  
 **figure3** = **new** ScatterChart<Number, Number>(x3, y3);  
  
 HBox hbox4 = **new** HBox(**figure**, **figure2**, **figure3**);  
 hbox4.setAlignment(Pos.***CENTER***);  
 layout.getChildren().add(hbox4);  
  
 Scene scene = **new** Scene(layout, 1000, 1000);  
  
 primaryStage.setScene(scene);  
 primaryStage.show();  
  
 }  
  
 **public static void** main(String[] args) {  
 *launch*(args);  
 }  
}

**2.2 Wyniki**



Wykres 2. Wykres zależności położenia kątowego i prędkości od czasu oraz przestrzeni fazowej.



Wykres 3. Wykres porównujący charakterystyki dla różnych wartości (wykres żółty = Wykres 2, wykres zielony - theta0=40, wykres niebieski - A=1).

*Theta0* podawana przez użytkownika jest w stopniach, następnie program przelicza je na radiany widoczne na wykresie 2.

**2.3 Wnioski**

Klasa *WahadloFX* została skonstruowana tak, by przy rysowaniu kolejnego wykresu poprzedni nie był kasowany, w celu porównania wykresów na jednym dla różnych wartości (wykres 3). Do wyczyszczenia wszystkich układów służy przycisk *Clear*. W celu większej rozróżnialności wartości można by zmniejszyć markery wykresów.