Лабораторная работа №5

Модель хищник-жертва

Ильин Андрей Владимирович

Содержание

# 1 Цель работы

Рассмотреть модель хищник-жертва. Построить вышеуказанную модель средствами OpenModellica и Julia.

# 2 Задачи

Для модели «хищник-жертва»:

1. Построить график зависимости численности хищников от численности жертв, а также графики изменения численности хищников и численности жертв при следующих начальных условиях: , .
2. Найти стационарное состояние системы.

# 3 Среда

* Julia – это открытый свободный высокопроизводительный динамический язык высокого уровня, созданный специально для технических (математических) вычислений. Его синтаксис близок к синтаксису других сред технических вычислений, таких как Matlab и Octave. [1]
* OpenModelica — свободное открытое программное обеспечение для моделирования, симуляции, оптимизации и анализа сложных динамических систем. Основано на языке Modelica. [2]

# 4 Теоретическое введение

Простейшая модель взаимодействия двух видов типа «хищник — жертва» - модель Лотки-Вольтерры. Данная двувидовая модель основывается на следующих предположениях [3]:

1. Численность популяции жертв x и хищников y зависят только от времени (модель не учитывает пространственное распределение популяции на занимаемой территории);
2. В отсутствии взаимодействия численность видов изменяется по модели Мальтуса, при этом число жертв увеличивается, а число хищников падает;
3. Естественная смертность жертвы и естественная рождаемость хищника считаются несущественными;
4. Эффект насыщения численности обеих популяций не учитывается;
5. Скорость роста численности жертв уменьшается пропорционально численности хищников;

В этой модели – число жертв, - число хищников. Коэффициент описывает скорость естественного прироста числа жертв в отсутствие хищников, - естественное вымирание хищников, лишенных пищи в виде жертв. Вероятность взаимодействия жертвы и хищника считается пропорциональной как количеству жертв, так и числу самих хищников (). Каждый акт взаимодействия уменьшает популяцию жертв, но способствует увеличению популяции хищников (члены и в правой части уравнения).

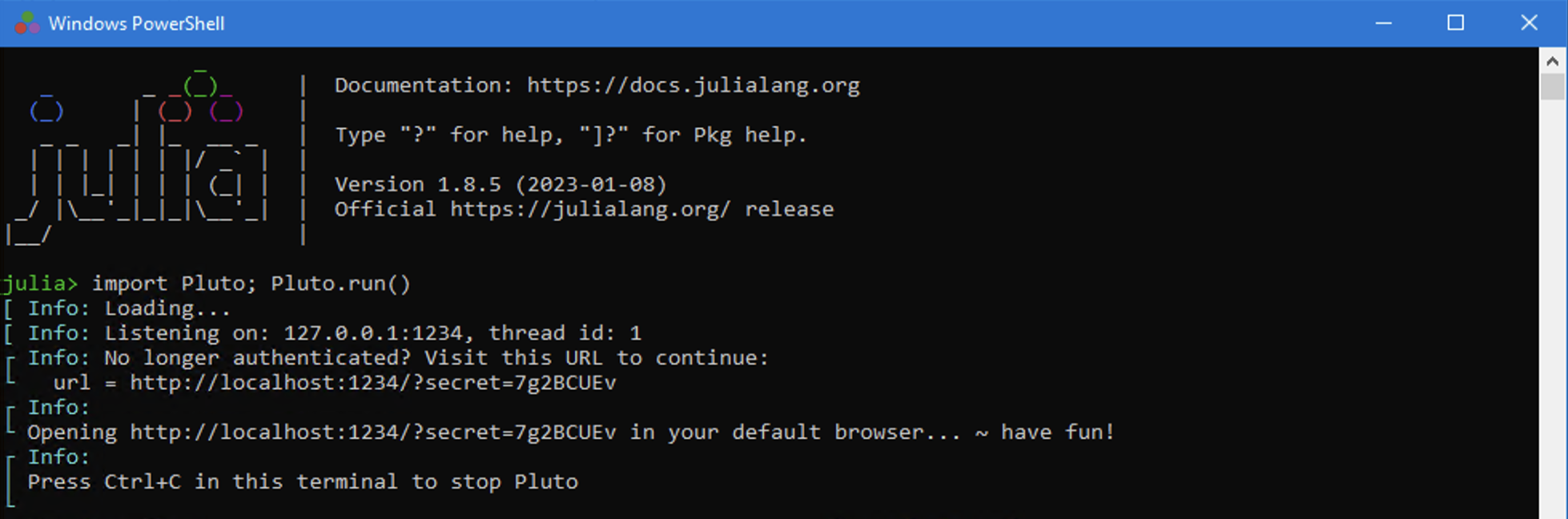
Стационарное состояние данной системы (положение равновесия, не зависящее от времени решение) будет в точке:

Если начальные значения задать в стационарном состоянии

то в любой момент времени численность популяций изменяться не будет. При малом отклонении от положения равновесия численности как хищника, так и жертвы с течением времени не возвращаются к равновесным значениям, а совершают периодические колебания вокруг стационарной точки. Амплитуда колебаний и их период определяется начальными значениями численностей , y(0). Колебания совершаются в противофазе.

# 5 Выполнение лабораторной работы

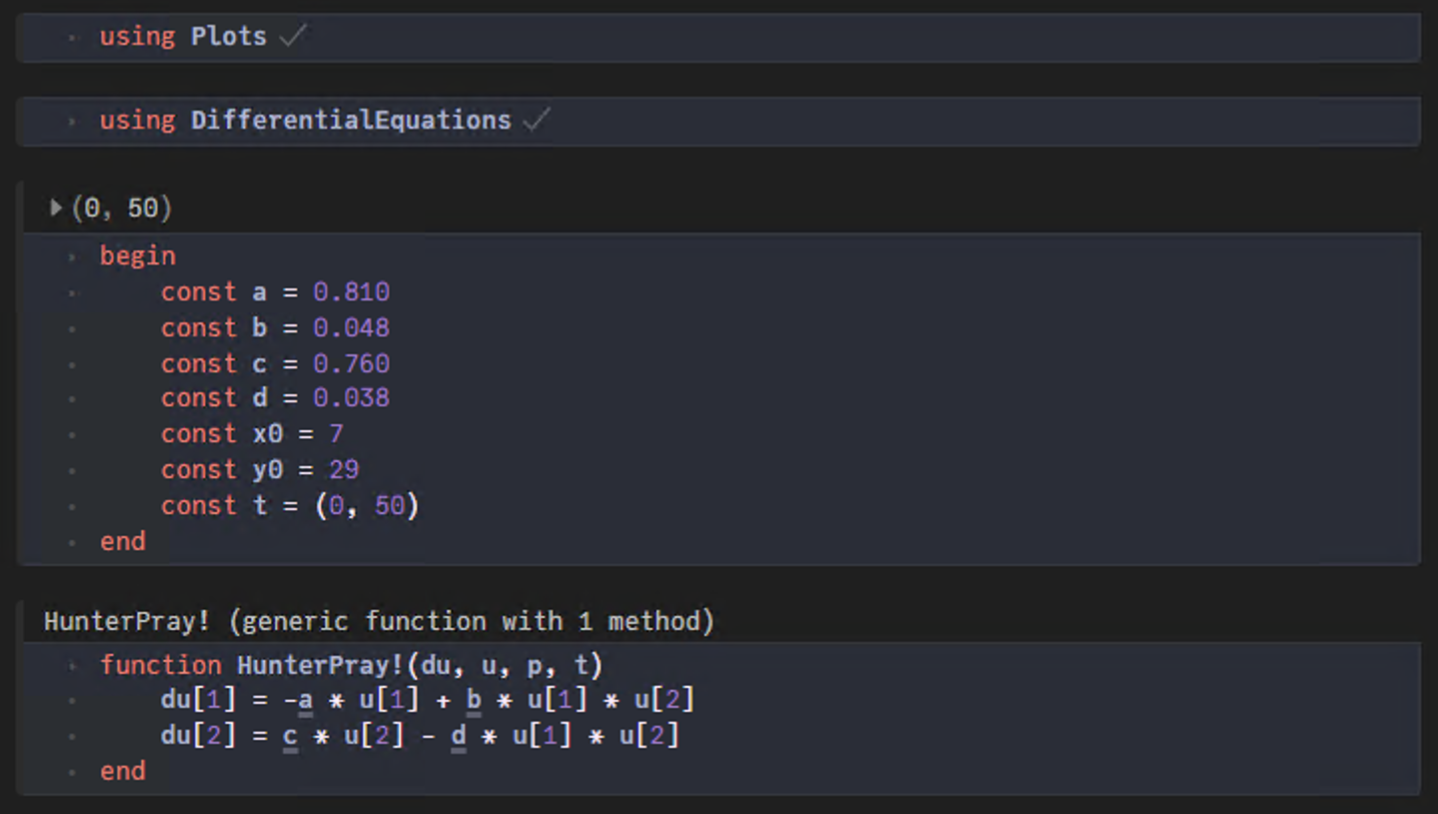
1. Начнем выполнения поставленных задач в Julia. Для этого запустим Pluto [4]. (рис. ??)



Julia. Запуск Pluto

1. Первым делом подкючим пакеты “Plots” [5] и “DifferentialEquations” [6]. Далее объявим начальные данные при помощи констант. Также объявим начальное условие для системы ДУ и промежуток времени, на котором будет проходить моделирование. После этого объявим функцию, представляющую систему. (рис. ??)

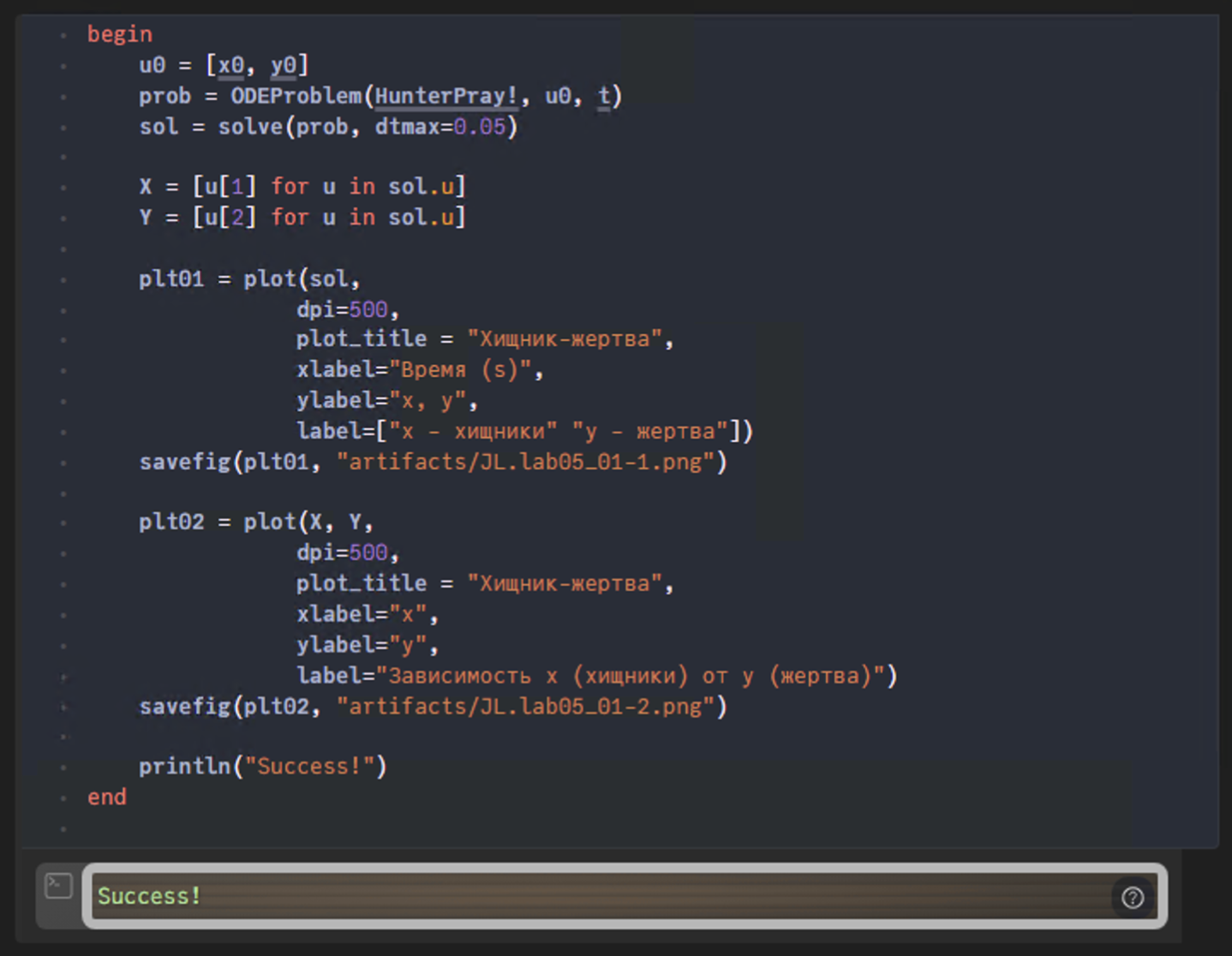
# подключение пакетов  
using Plots  
using DifferentialEquations  
  
# входные данные  
const a = 0.810  
const b = 0.048  
const c = 0.760  
const d = 0.038  
const x0 = 7  
const y0 = 29  
  
const t = (0, 50) # промежуток времени  
  
function HunterPray!(du, u, p, t)  
 du[1] = -a \* u[1] + b \* u[1] \* u[2]  
 du[2] = c \* u[2] - d \* u[1] \* u[2]  
end



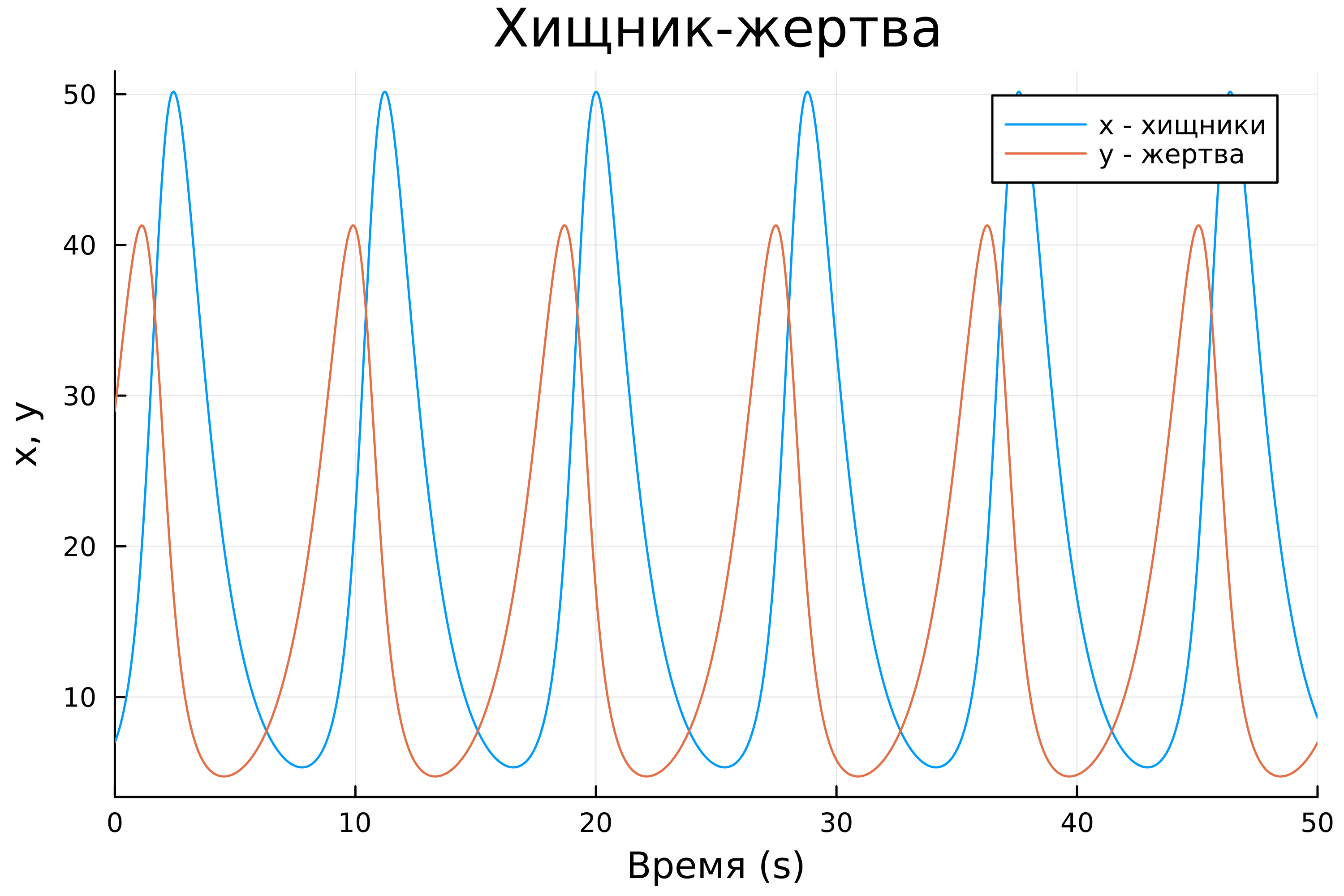
Julia. Скрипт (1). Модель “Хищник-жертва” (при начальных условиях x = 7, y = 29)

1. В следующей ячейке Pluto построим график зависимости от и графики функций , . При помощи ‘DifferentialEquations’ зададим и решим систему ДУ, после чего построим графики функций , $y(t). Так же создадим два списка, в которых будут храниться точки уравнений. Воспользуемся данным списком, чтобы построить график зависимости от . Далее запустим итоговый скрипт. (рис. ??, ??, ??)

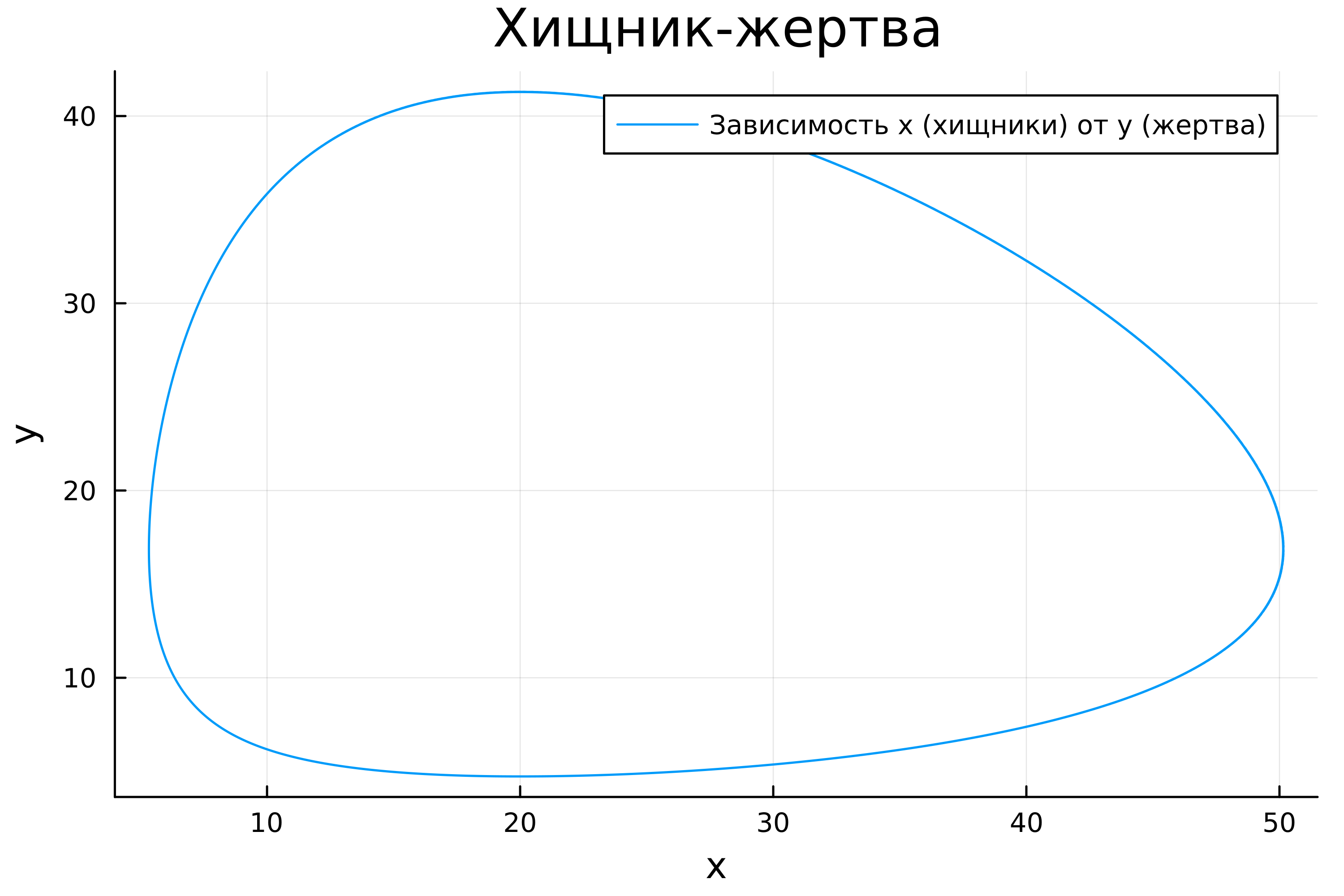
begin  
u0 = [x0, y0] # начальные условия  
prob = ODEProblem(HunterPray!, u0, t)  
sol = solve(prob, dtmax=0.05)  
  
X = [u[1] for u in sol.u]  
Y = [u[2] for u in sol.u]  
  
# используем Plots,  
# чтобы построить графики  
plt01 = plot(sol,  
 dpi=500,  
 plot\_title = "Хищник-жертва",  
 xlabel="Время (s)",  
 ylabel="x, y",  
 label=["x - хищник" "y - жертва"])  
savefig(plt01, "artifacts/JL.lab05\_01-1.png")  
  
plt02 = plot(X, Y,  
 dpi=500,  
 plot\_title = "Хищник-жертва",  
 xlabel="x",  
 ylabel="y",  
 label="Зависимость x (хищник) от y (жертва)")  
savefig(plt02, "artifacts/JL.lab05\_01-2.png")  
  
println("Success!")



Julia. Скрипт (2). Модель “Хищник-жертва” (при начальных условиях x = 7, y = 29)



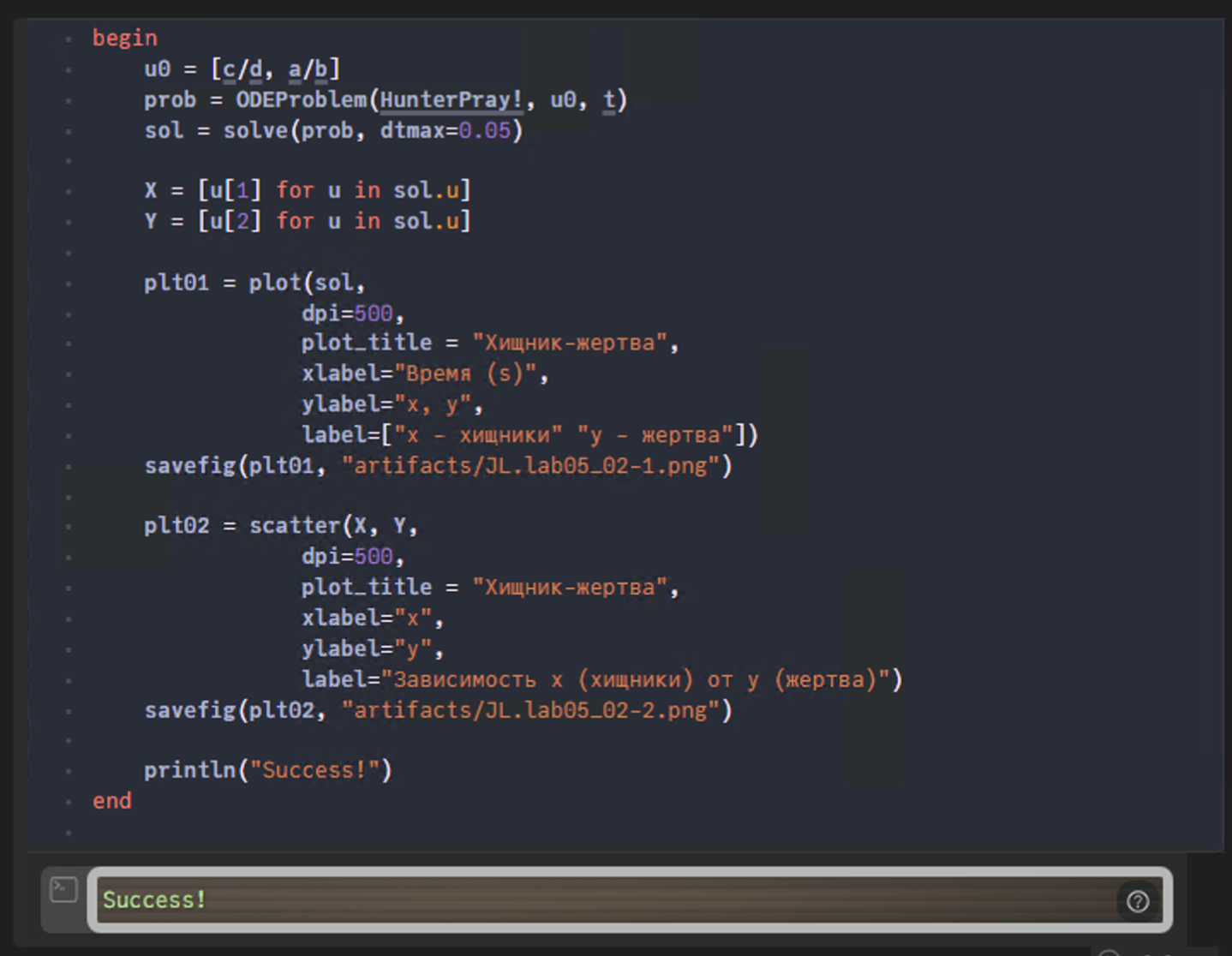
Julia. Модель. Графики функций изменения численности хищников и изменения численности жертв (при начальных условиях x = 7, y = 29)



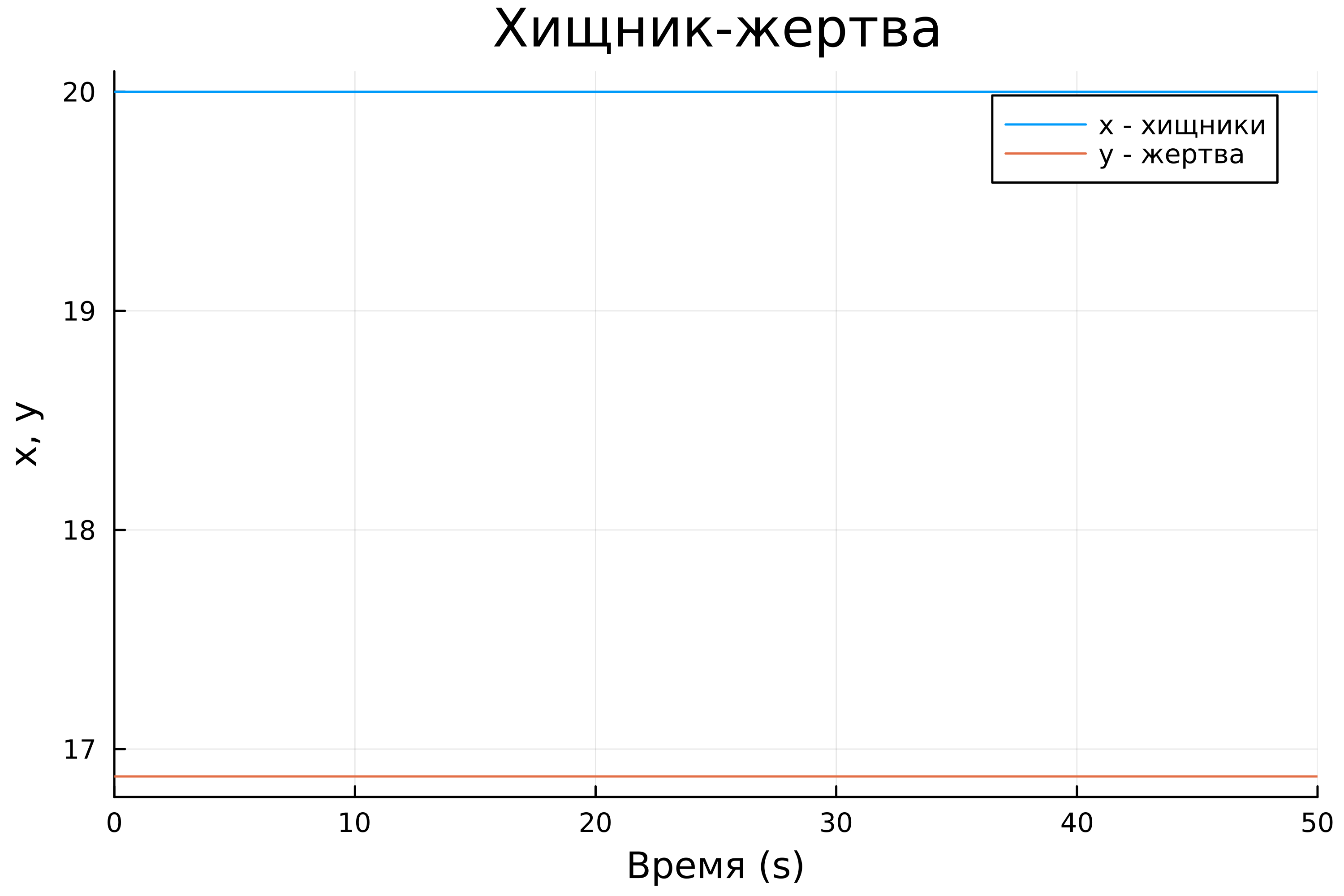
Julia. Модель. График зависимости изменения численности хищников от изменения численности жертв (при начальных условиях x = 7, y = 29)

1. Изменим начальные значения в вышеуказанном скрипте, при которых будет достигаться положение равновесия (не зависящее от времени решение). (рис. ??, ??, ??)

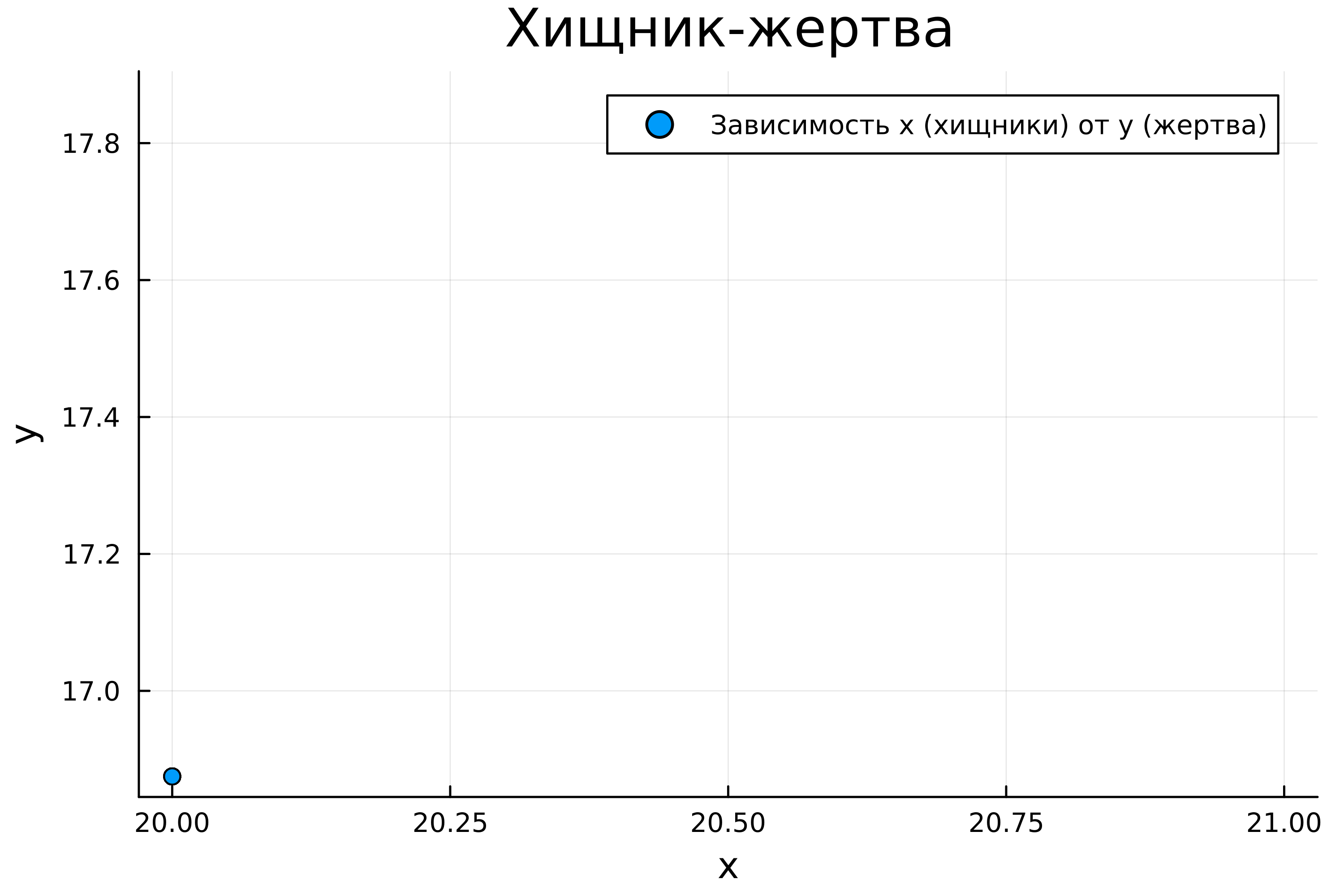
u0 = [c/d, a/b] #!  
prob = ODEProblem(HunterPray!, u0, t)  
sol = solve(prob, dtmax=0.05)  
  
X = [u[1] for u in sol.u]  
Y = [u[2] for u in sol.u]  
  
plt01 = plot(sol,  
 dpi=500,  
 plot\_title = "Хищник-жертва",  
 xlabel="Время (s)",  
 ylabel="x, y",  
 label=["x - хищники" "y - жертва"])  
savefig(plt01, "artifacts/JL.lab05\_02-1.png")  
  
plt02 = scatter(X, Y,  
 dpi=500,  
 plot\_title = "Хищник-жертва",  
 xlabel="x",  
 ylabel="y",  
 label="Зависимость x (хищники) от y (жертва)")  
savefig(plt02, "artifacts/JL.lab05\_02-2.png")  
  
println("Success!")



Julia. Скрипт. Модель “Хищник-жертва” (стационарное состояние)



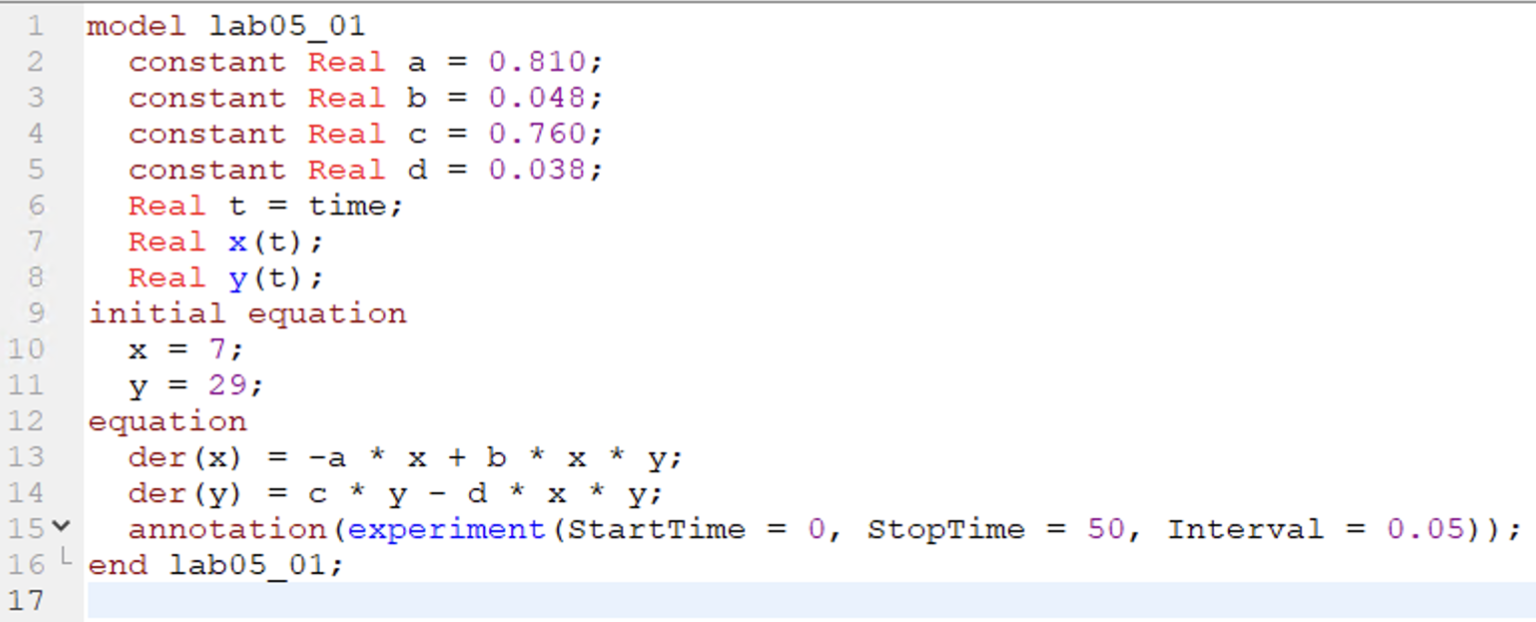
Julia. Модель. Графики функций изменения численности хищников и изменения численности жертв (стационарное состояние)



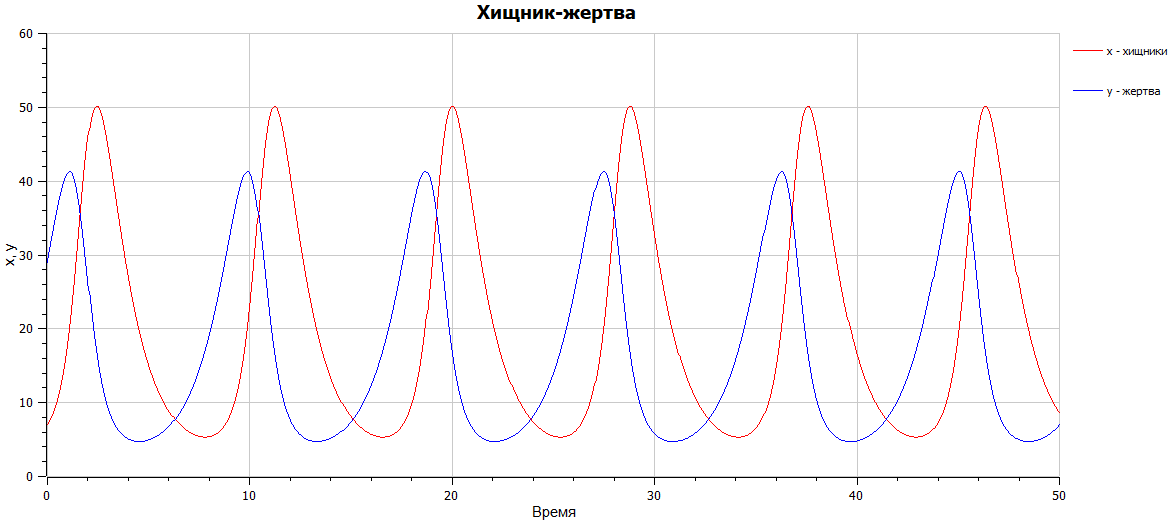
Julia. Модель. График зависимости изменения численности хищников от изменения численности жертв (стационарное состояние)

1. Построим график зависимости численности хищников от численности жертв, а также графики изменения численности хищников и численности жертв при начальных условиях на Modelica. (рис. ??, ??, ??)

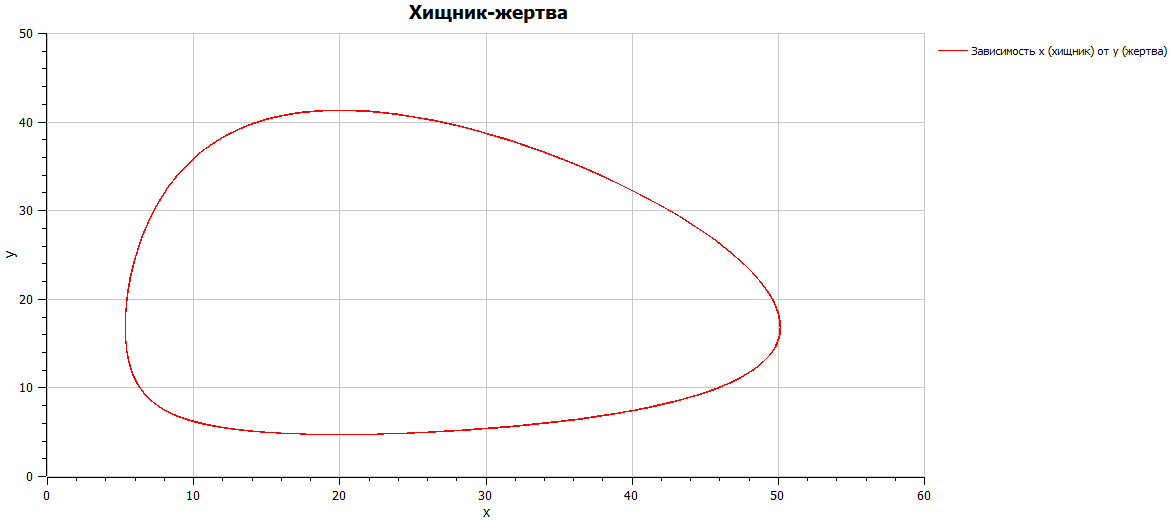
model lab05\_01  
 constant Real a = 0.810;  
 constant Real b = 0.048;  
 constant Real c = 0.760;  
 constant Real d = 0.038;  
 Real t = time;  
 Real x(t);  
 Real y(t);  
initial equation  
 x = 7;  
 y = 29;  
equation  
 der(x) = -a \* x + b \* x \* y;  
 der(y) = c \* y - d \* x \* y;  
 annotation(experiment(StartTime = 0, StopTime = 50, Interval = 0.05));  
end lab05\_01;



Modelica. Скрипт. Модель “Хищник-жертва” (при начальных условиях x = 7, y = 29)



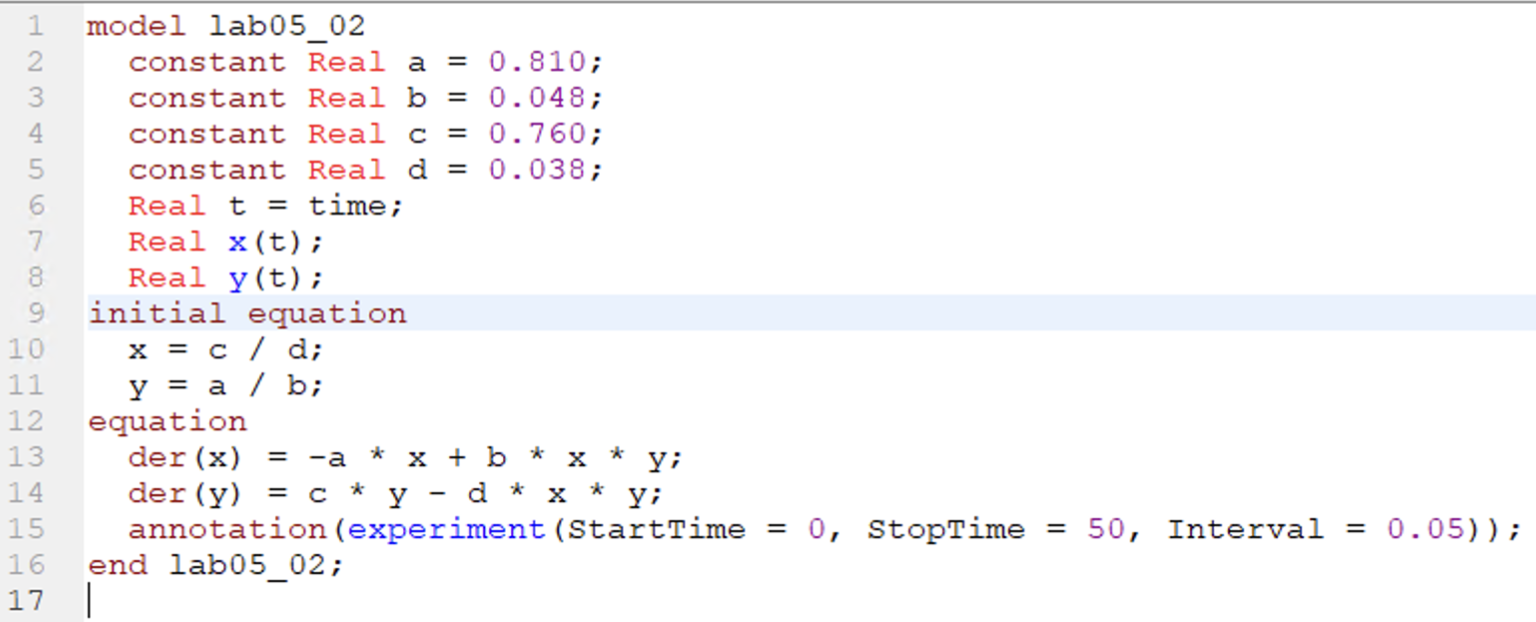
Modelica. Модель. Графики функций изменения численности хищников и изменения численности жертв (при начальных условиях x = 7, y = 29)



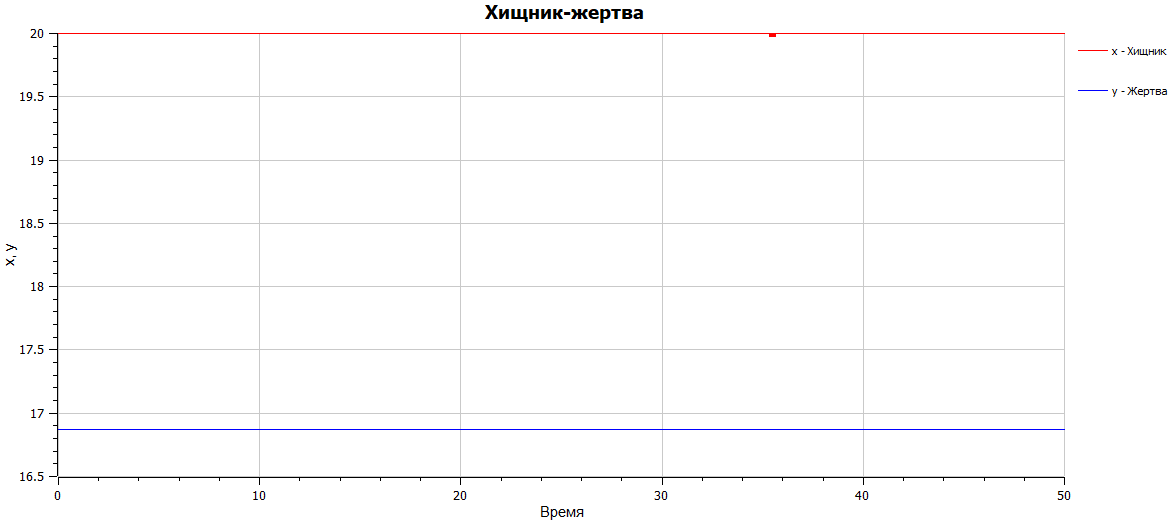
Modelica. Модель. График зависимости изменения численности хищников от изменения численности жертв (при начальных условиях x = 7, y = 29)

1. Построим график зависимости численности хищников от численности жертв, а также графики изменения численности хищников и численности жертв в стационарном состоянии на Modelica. (рис. ??, ??, ??)

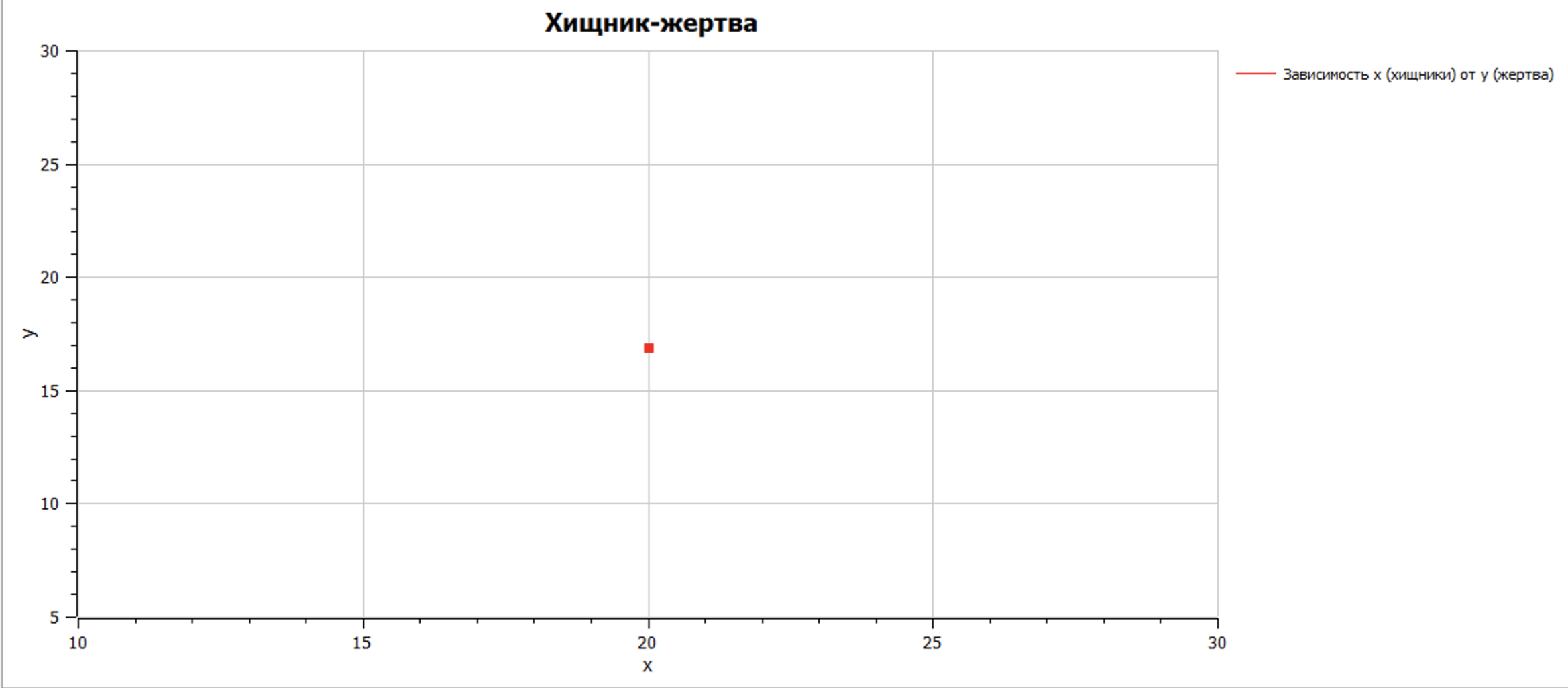
model lab05\_02  
 constant Real a = 0.810;  
 constant Real b = 0.048;  
 constant Real c = 0.760;  
 constant Real d = 0.038;  
 Real t = time;  
 Real x(t);  
 Real y(t);  
initial equation  
 x = c / d;  
 y = a / b;  
equation  
 der(x) = -a \* x + b \* x \* y;  
 der(y) = c \* y - d \* x \* y;  
 annotation(experiment(StartTime = 0, StopTime = 50, Interval = 0.05));  
end lab05\_02;



Modelica. Скрипт. Модель “Хищник-жертва” (стационарное состояние)



Modelica. Модель. Графики функций изменения численности хищников и изменения численности жертв (стационарное состояние)



Modelica. Модель. График зависимости изменения численности хищников от изменения численности жертв (стационарное состояние)

# 6 Анализ результатов

Работа выполненна без непредвиденных проблем в соответствии с руководством. Ошибок и сбоев не произошло.

Моделирование на OMEdit было проще и быстрее, чем при использовании средств Julia. Скрипт на Modelica вышел более лакончиным, понятным и коротким. Более того OpenModelica быстрее обрабатывала скрипт и симмулировала модель. Стоит отметить, что OpenModelica имеет множество разлиных полезных инструментов для настройки с симмуляцией и работой с ней. К плюсам Julia можно отнести, что она является языком программирования, который хорошо подходит для математических и технических задач.

# 7 Выводы

Мы улучшили практические навыки в области дифференциальных уравнений, улучшили навыки моделирования на Julia, также навыки моделирования на OpenModelica. Изучили модель взаимодействия двух видов типа «хищник — жертва», а именно модель Лотки-Вольтерры.

# Список литературы

1. Julia [Электронный ресурс]. URL: <http://www.unn.ru/books/met_files/JULIA_tutorial.pdf>.

2. OpenModelica [Электронный ресурс]. URL: <https://ru.wikipedia.org/wiki/OpenModelica>.

3. Модель хищник-жертва [Электронный ресурс]. RUDN. URL: <https://esystem.rudn.ru/mod/resource/view.php?id=967245>.

4. Pluto [Электронный ресурс]. URL: <https://plutojl.org/>.

5. Plots in Julia [Электронный ресурс]. URL: <https://docs.juliaplots.org/latest/tutorial/>.

6. Differential Equations in Julia [Электронный ресурс]. URL: <https://docs.sciml.ai/DiffEqDocs/stable/getting_started/>.