

# **Отчет по лабораторной работе №2**

**по дисциплине: Математическое моделирование**

**Ким Михаил Алексеевич**

# Содержание

<b>1 Цель работы</b>	<b>4</b>
<b>2 Задание</b>	<b>5</b>
<b>3 Теоретическое введение</b>	<b>6</b>
3.1 Julia . . . . .	6
3.2 OpenModelica . . . . .	6
<b>4 Выполнение лабораторной работы</b>	<b>8</b>
4.1 Подготовка системы для работы . . . . .	8
4.1.1 Установка Julia . . . . .	8
4.1.2 Установка Visual Studio Code . . . . .	13
4.1.3 Установка OpenModelica . . . . .	14
4.2 Выполнение лабораторной работы . . . . .	17
4.2.1 Julia . . . . .	17
4.2.2 OpenModelica . . . . .	24
4.2.3 Математические вычисления . . . . .	26
<b>5 Выводы</b>	<b>28</b>
<b>Список литературы</b>	<b>29</b>

# Список иллюстраций

4.1	Загрузка архива с официального сайта . . . . .	8
4.2	Распаковка и установка архива . . . . .	8
4.3	Копирование архива в директорию /opt . . . . .	8
4.4	Создание символьической ссылки . . . . .	9
4.5	Установка завершена . . . . .	9
4.6	Ввод символа «]» инициализирует менеджер пакетов . . . . .	10
4.7	Добавление библиотеки для работы с диф. уравнениями . . . . .	10
4.8	Установка библиотеки . . . . .	11
4.9	Добавление библиотеки для отрисовки графиков . . . . .	12
4.10	Установка библиотеки . . . . .	12
4.11	Нахождение номера варианта задания к лабораторной работе . .	12
4.12	Вывод «Hello, world!» в консоль . . . . .	13
4.13	Интерактивное окно скачивания . . . . .	13
4.14	Установка пакетов . . . . .	14
4.15	Специальные символы доступны на Julia . . . . .	14
4.16	Процесс установки OpenModelica. 1 . . . . .	15
4.17	Процесс установки OpenModelica. 2 . . . . .	15
4.18	Получение версии OpenModelica . . . . .	15
4.19	Установка библиотеки Modelica в редакторе OMEedit . . . . .	16
4.20	Добавление библиотеки в перечень используемых в OMEedit . .	16
4.21	Установка библиотеки ModelicaByExample в редакторе OMEedit .	17
4.22	Код лабораторной работы на Julia. 1 . . . . .	21
4.23	Код лабораторной работы на Julia. 2 . . . . .	22
4.24	Результат для первого случая . . . . .	23
4.25	Результат для второго случая . . . . .	23
4.26	Код лабораторной работы на Modelica для первого случая . .	25
4.27	Код лабораторной работы на Modelica для второго случая . .	25
4.28	Результат для первого случая . . . . .	26
4.29	Результат для второго случая . . . . .	26
4.30	Вычисления . . . . .	27

# 1 Цель работы

Ознакомиться с базовым функционалом языка программирования Julia, а также интерактивной командной строкой REPL. Ознакомиться с базовым функционалом языка моделирования Modelica и программным обеспечением OpenModelica. Используя эти средства, построить математическую модель, представляющую собой частный случай задачи о погоне.

## 2 Задание

1. Записать уравнение, описывающее движение катера, с начальными условиями для двух случаев (в зависимости от расположения катера относительно лодки в начальный момент времени).
2. Построить траекторию движения катера и лодки для двух случаев.
3. Найти точку пересечения траектории катера и лодки

# 3 Теоретическое введение

## 3.1 Julia

Julia является высокоуровневым языком программирования, разработанным с целью создания высокоэффективного и легкого для использования языка программирования. Он объединяет лучшие аспекты других языков программирования, таких как Python, R и Matlab [1].

Julia может использоваться в любой области применения, включая инженерные вычисления, анализ данных, машинное обучение и научные вычисления. Она также поддерживает многопоточность и масштабируемость, позволяя программистам писать более эффективные программы, способные обрабатывать большие данные [2].

В целом, Julia является мощным и гибким языком программирования, который может использоваться для решения различных задач. Он предоставляет программистам широкий диапазон возможностей для создания более мощных и высокопроизводительных приложений [3].

## 3.2 OpenModelica

OpenModelica – это свободно распространяемый открытый исходный инструментальный комплекс для моделирования, анализа и разработки систем и проектирования. Он был основан в 1999 году на базе проекта Modelica [4].

OpenModelica является бесплатным программным обеспечением, поддержи-

вает интеграцию с различными языками программирования и позволяет пользовательским приложениям и инструментам работать в единой среде. Области применения OpenModelica включают системное моделирование и проектирование, моделирование сложных автоматизированных систем и многое другое. Он также используется в целях обучения [5].

# 4 Выполнение лабораторной работы

## 4.1 Подготовка системы для работы

### 4.1.1 Установка Julia

1. Устанавливаем язык программирования Julia вместе с интерактивной командной строкой REPL при помощи команд (рис. 4.1, 4.2, 4.3, 4.4).

```
wget https://julialang-s3.julialang.org/bin/linux/x64/1.8/julia-  
1.8.5-linux-x86_64.tar.gz  
tar zxvf julia-1.8.5-linux-x86_64.tar.gz  
sudo cp -r julia-1.8.5 /opt/  
sudo ln -s /opt/julia-1.8.5/bin/julia /usr/local/bin/julia
```

```
makim@makim:~$ wget https://julialang-s3.julialang.org/bin/linux/x64/1.8/julia-1.8.5-linux-x86_64.tar.gz
```

Рис. 4.1: Загрузка архива с официального сайта

```
makim@makim:~$ tar julia-1.8.5-linux-x86_64.tar.gz
```

Рис. 4.2: Распаковка и установка архива

```
makim@makim:~$ sudo cp -r julia-1.8.5 /opt/
```

Рис. 4.3: Копирование архива в директорию /opt

```
makim@makim:~$ sudo ln -s /opt/julia-1.8.5/bin/julia /usr/local/bin/julia
```

Рис. 4.4: Создание символьической ссылки

Рис. 4.5: Установка завершена

## 2. Устанавливаем дополнительные библиотеки (рис. 4.6, 4.7, 4.8, 4.9, 4.10).

add DifferentialEquations  
add Plots

```
(@v1.8) pkg>
```

Рис. 4.6: Ввод символа «]» инициализирует менеджер пакетов

```
(@v1.8) pkg> add DifferentialEquations
```

Рис. 4.7: Добавление библиотеки для работы с диф. уравнениями

```
Downloaded artifact: Sundials
Downloaded artifact: Rmath
Downloaded artifact: OpenSpecFun
Updating `~/.julia/environments/v1.8/Project.toml`
[0c46a032] + DifferentialEquations v7.6.0
Updating `~/.julia/environments/v1.8/Manifest.toml`
[79e6a3ab] + Adapt v3.5.0
[ec485272] + ArnoldiMethod v0.2.0
[4fba245c] + ArrayInterface v6.0.25
[30b0a656] + ArrayInterfaceCore v0.1.29
[6ba088a2] + ArrayInterfaceGPUArrays v0.2.2
[015c0d05] + ArrayInterfaceOffsetArrays v0.1.7
[b0d46f97] + ArrayInterfaceStaticArrays v0.1.5
[dd5226c6] + ArrayInterfaceStaticArraysCore v0.1.3
[4c555306] + ArrayLayouts v0.8.18
[aae01518] + BandedMatrices v0.17.12
[62783981] + BitTwiddlingConvenienceFunctions v0.1.5
[764a87c0] + BoundaryValueDiffEq v2.11.0
[fa961155] + CEnum v0.4.2
[2a0fbf3d] + CPUSummary v0.2.2
[49dc2e85] + Calculus v0.5.1
[d360d2e6] + ChainRulesCore v1.15.7
[9e997f8a] + ChangesOfVariables v0.1.5
[fb6a15b2] + CloseOpenIntervals v0.1.11
[38540f10] + CommonSolve v0.2.3
[bbf7d656] + CommonSubexpressions v0.3.0
[34da2185] + Compat v4.6.0
[187b0558] + ConstructionBase v1.4.1
[adafc99b] + CpuId v0.3.1
[9a962f9c] + DataAPI v1.14.0
[864edb3b] + DataStructures v0.18.13
```

Рис. 4.8: Установка библиотеки

```
(@v1.8) pkg> add Plots
```

Рис. 4.9: Добавление библиотеки для отрисовки графиков

```
Installed Lzo_jll v2.10.1+0
Installed PlotThemes v3.1.0
Installed FriBidi_jll v1.0.10+0
Installed fzf_jll v0.29.0+0
Installed UnicodeFun v0.4.1
Installed TranscodingStreams v0.9.11
Installed GLFW_jll v3.3.8+0
Installed MbedTLS v1.1.7
Installed FreeType2_jll v2.10.4+0
Installed x264_jll v2021.5.5+0
Installed JLFzf v0.1.5
Installed CodecZlib v0.7.1
Installed Colors v0.12.10
Installed Xorg_libxcb_jll v1.13.0+3
Installed libaom_jll v3.4.0+0
Installed libpng_jll v1.6.38+0
Installed Scratch v1.1.1
```

Рис. 4.10: Установка библиотеки

3. Проверим правильность установки языка (рис. 4.11, 4.12).

```
julia> (1032201664 % 70) + 1
5
```

Рис. 4.11: Нахождение номера варианта задания к лабораторной работе

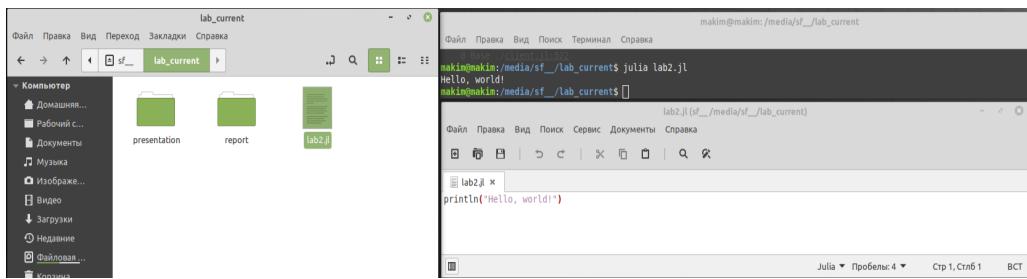


Рис. 4.12: Вывод «Hello, world!» в консоль

#### 4.1.2 Установка Visual Studio Code

1. Скачиваем файл .deb с официального сайта (рис. 4.13, 4.14, 4.15).

```
sudo apt install ./code_1.75.1-1675893397_amd64.deb
```

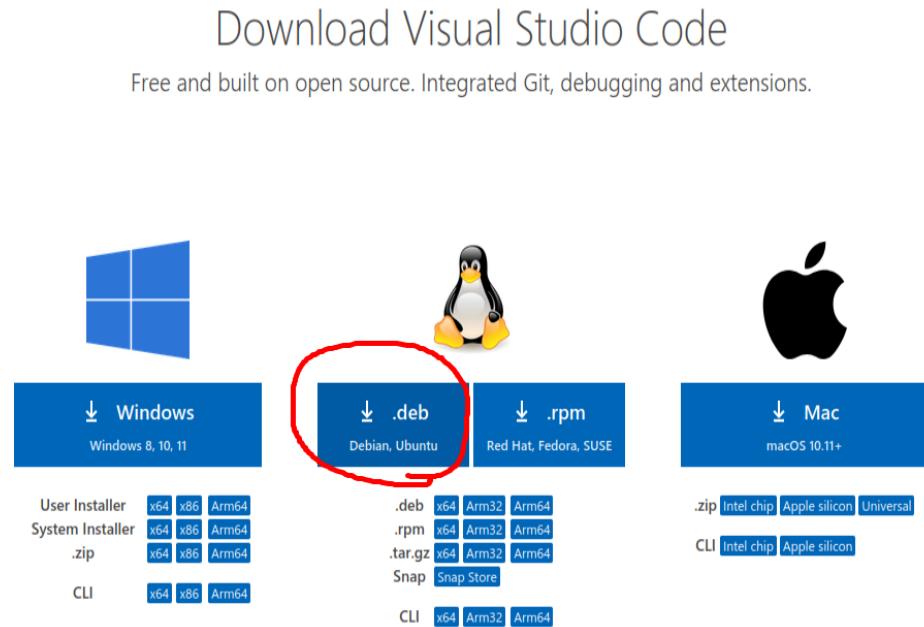


Рис. 4.13: Интерактивное окно скачивания

```
makim@makim:/media/st__/lab_current$ cd ~/Загрузки
makim@makim:~/Загрузки$ sudo apt install ./code_1.75.1-1675893397_amd64.deb
[sudo] пароль для makim:
Чтение списков пакетов... Готово
Построение дерева зависимостей
Чтение информации о состоянии... Готово
Заметьте, вместо «./code_1.75.1-1675893397_amd64.deb» выбирается «code»
Следующие пакеты устанавливались автоматически и больше не требуются:
  linux-headers-5.4.0-89 linux-headers-5.4.0-89-generic linux-image-5.4.0-89-
  linux-modules-extra-5.4.0-89-generic
Для их удаления используйте «sudo apt autoremove».
Следующие НОВЫЕ пакеты будут установлены:
  code
```

Рис. 4.14: Установка пакетов

2. Использование UTF-символов в коде Julia в VSC (рис. 4.15).



Рис. 4.15: Специальные символы доступны на Julia

### 4.1.3 Установка OpenModelica

1. Устанавливаем программное обеспечение OpenModelica при помощи команд (рис. 4.16, 4.17).

```
for deb in deb deb-src; do echo "$deb http://build.openmodelica.org/apt `lsb-
cs` release"; done | sudo tee /etc/apt/sources.list.d/openmodelica.list
wget -q http://build.openmodelica.org/apt/openmodelica.asc -
0- | sudo apt-key add -
sudo apt update
sudo apt install openmodelica
```

```
makim@makim:~$ for deb in deb deb-src; do echo "$deb http://build.openmodelica.org/apt `lsb_release -cs` release"; done | sudo tee /etc/apt/sources.list.d/openmodelica.list
[sudo] пароль для makim:
deb http://build.openmodelica.org/apt uma release
deb-src http://build.openmodelica.org/apt uma release
makim@makim:~$ wget -q http://build.openmodelica.org/apt/openmodelica.asc -O- | sudo apt-key add -
OK
makim@makim:~$ sudo apt update
Сущ:1 http://mirror.docker.ru/ubuntu focal InRelease
Сущ:2 http://mirror.docker.ru/ubuntu focal-updates InRelease
Сущ:3 http://mirror.docker.ru/ubuntu focal-backports InRelease
```

Рис. 4.16: Процесс установки OpenModelica. 1

```
makim@makim:~$ sudo apt install openmodelica
Чтение списков пакетов... Готово
Построение дерева зависимостей
Чтение информации о состоянии... Готово
Следующие пакеты устанавливались автоматически и больше
  linux-headers-5.4.0-89 linux-headers-5.4.0-89-generic
  linux-image-5.4.0-89-generic linux-modules-5.4.0-89-g
  linux-modules-extra-5.4.0-89-generic
Для их удаления используйте «sudo apt autoremove».
Будут установлены следующие дополнительные пакеты:
  aqlfn binfmt-support clang clang-10 drcontrol drmodel
```

Рис. 4.17: Процесс установки OpenModelica. 2

2. Проверяем правильность установки (рис. 4.18).

```
/usr/bin/omc --version
```

```
makim@makim:~$ /usr/bin/omc --version
OpenModelica 1.20.0
```

Рис. 4.18: Получение версии OpenModelica

3. Устанавливаем дополнительные библиотеки (рис. 4.19, 4.20, 4.21).

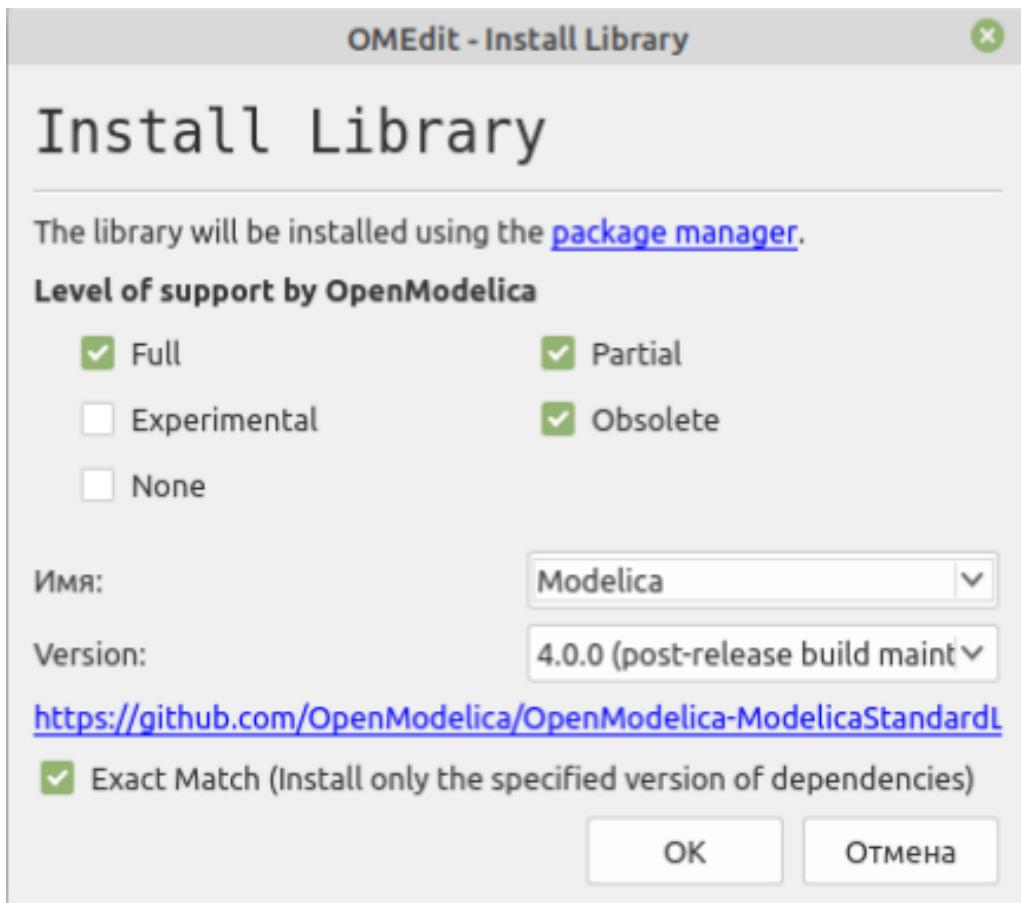


Рис. 4.19: Установка библиотеки Modelica в редакторе OMEedit

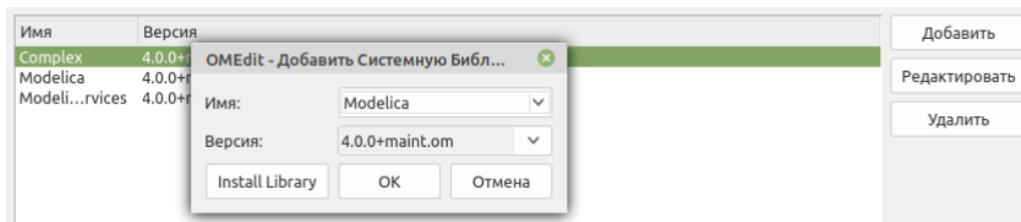


Рис. 4.20: Добавление библиотеки в перечень используемых в OMEedit

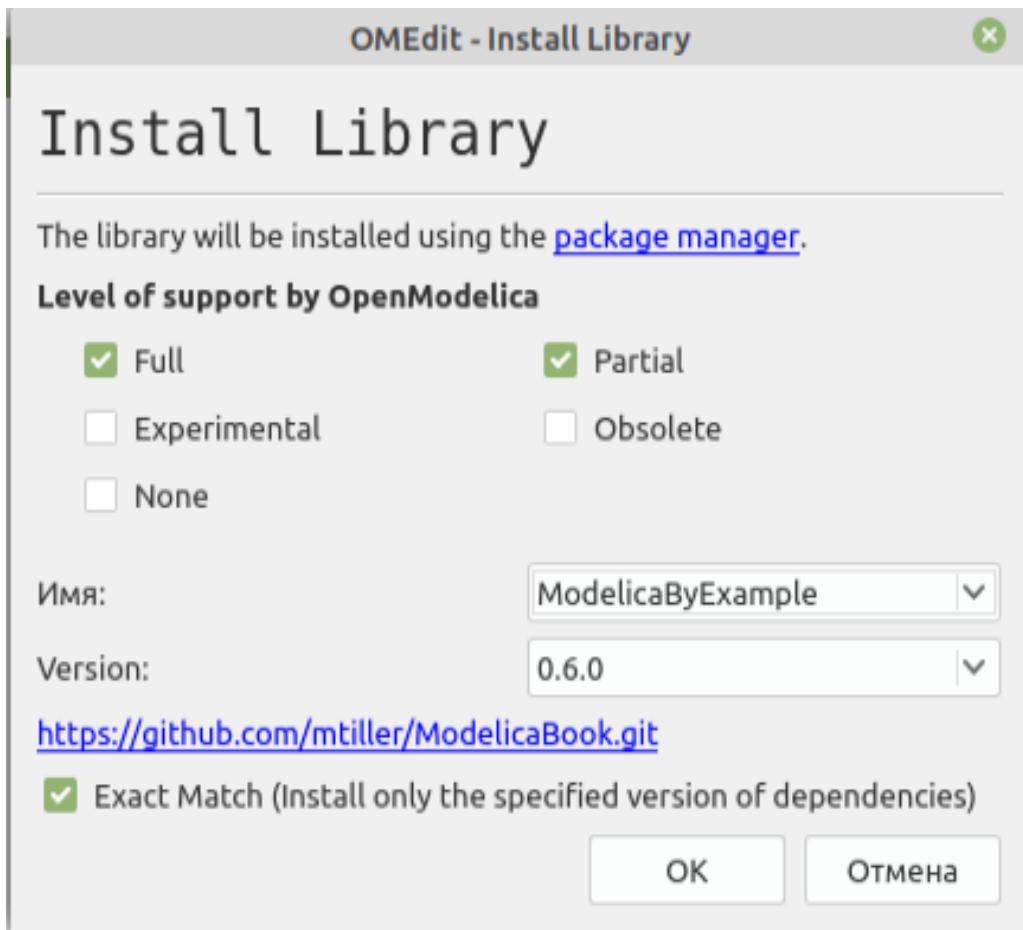


Рис. 4.21: Установка библиотеки ModelicaByExample в редакторе OMEdit

## 4.2 Выполнение лабораторной работы

### 4.2.1 Julia

1. Пишем программу, воспроизводящую модель на языке программирования Julia (рис. 4.22, 4.23).

```
# Дополнительная информация
# dr(r) = r / √(3)           # Короткая запись функции, sqrt(3) ~ \sqrt + 3
# x = range(0, 2π, 1000)      # 100 значений от 0 до 2π
# @show x                    # Макрос, позволяющий выводить не только зна
```

```

# rfloat = rand(0.0:0.0001:2π) # Случайное float64 значение от 0.0 до 2π с

# Подключение библиотек
using Plots
using DifferentialEquations

const n = 2.5

"Правая часть ОДУ: u - переменная (скаляр или массив), p - параметры (корте
function F(u, p, t)
    # аргументы p, t нужны для работы метода
    return u / √(n * n - 1)
end

"Функция решения задачи, a - начальное расстояние катера от лодки, n - отнош
function the_chase_curve(a, n, flag)
    "Расстояние, на котором катер начнет описывать спираль"
    r = 0.0
    "Интервал спирали"
    T = (0.0, 0.0)

    if flag == 0
        r = a / (n + 1)
        T = (0, 2π)
    elseif flag == 1
        r = -(a / (n - 1))
    end
end

```

```

T = (-π, π)
else
    println("Неправильно выбран флаг")
    return -1
end

# Задача
prob = ODEProblem(F, abs(r), T)

# Решение задачи
sol = solve(
    prob,
    dtmax=0.5
)

# Задание пустого пространства
plt = plot(
    proj=:polar,
    ylims=(0.0, round(abs(sol.u[size(sol.u)[1]])) + 10.0),
    aspect_ratio=:equal,      # Размер одного деления по осям всегда одинаковый
    dpi=600,
    legend=true              # Отображение легенды
    # bg=:black
)

@show sol.t
@show sol.u

r_idx = rand(1:size(sol.t)[1])

```

```

r_angle = sol.t[r_idx]
# r_angle_list = [sol.t[r_idx] for n in 1:size(sol.t)[1]]

plot!(plt, xlabel="x", ylabel="r(t)", title="Кривая погони. Частный случай")
savefig(plt, "lab2_" * string(flag) * "_0")

plot!(plt, [0.0, 0.0], [a, r], label="Начальное катера движение по прямой")
scatter!(plt, [0.0], [a], label=" ", mc=:brown, ms=0.4)
plot!(plt, [r_angle, r_angle], [0.0, sol.u[1]], label="Траектория лодки")
scatter!(plt, [r_angle], [sol.u[1]], label=" ", mc=:blue, ms=0.01)
plot!(plt, [sol.t[1], sol.t[1]], [sol.u[1], sol.u[1]], label="Траектория")
savefig(plt, "lab2_" * string(flag) * "_1")

for i in 2:size(sol.t)[1]
    # Добавление параметров в пространство
    plot!(plt, [r_angle, r_angle], [sol.u[i-1], sol.u[i]], label=" ", color=:black)
    scatter!(plt, [r_angle], [sol.u[i]], label=" ", mc=:blue, ms=0.01)

    plot!(plt, [sol.t[i-1], sol.t[i]], [sol.u[i-1], sol.u[i]], label=" ", color=:black)
    scatter!(plt, [sol.t[i]], [sol.u[i]], label=" ", mc=:red, ms=0.01)

    savefig(plt, "lab2_" * string(flag) * "_" * string(i))
end

return 0
end

the_chase_curve(6.2, n, 1)

```

```

1  # Дополнительная информация
2  # dr(r) = r / sqrt(3)          # Короткая запись функции, sqrt(3) ~ \sqrt{3}
3  # 0 = range(0, 2π, 1000)       # 100 значений от 0 до 2π
4  # @show φ                    # Макрос, позволяющий выводить не только значение, но и имя переменной.
5  # rfloat = rand(0.0:0.0001:2π) # Случайное float64 значение от 0.0 до 2π с шагом 0.0001
6
7
8  # Подключение библиотек
9  using Plots
10 using DifferentialEquations
11
12
13 const n = 2.5
14
15
16 "Правая часть ОДУ: u - переменная (скаляр или массив), p - параметры (кортеж, tuple), t - аргумент (скаляр, время)."
17 function F(u, p, t)
18     # аргументы p, t нужны для работы метода
19     return u / sqrt(n * n - 1)
20 end
21
22 "Функция решения задачи, a - начальное расстояние катера от лодки, n - отношение скорости катера к лодке"
23 function the_chase_curve(a, n, flag)
24     "Расстояние, на котором катер начнет описывать спираль"
25     r = 0.0
26     "Интервал спирали"
27     T = (0.0, 0.0)
28
29     if flag == 0
30         r = a / (n + 1)
31         T = (0, 2π)
32     elseif flag == 1
33         r = -(a / (n - 1))
34         T = (-π, π)
35     else
36         println("Неправильно выбран флаг")
37         return -1
38     end
39
40     # Задача
41     prob = ODEProblem(F, abs(r), T)
42
43     # Решение задачи
44     sol = solve(
45         prob,
46         dtmax=0.5
47     )

```

Рис. 4.22: Код лабораторной работы на Julia. 1

```

49     # Задание пустого пространства
50     plt = plot(
51         proj=:polar,
52         ylims=(0.0, round(abs(sol.u[size(sol.u)[1]])) + 10.0),
53         aspect_ratio=:equal,      # Размер одного деления по осям всегда одинакова
54         dpi=600,
55         legend=true            # Отображение легенды
56         # bg=:black
57     )
58
59     @show sol.t
60     @show sol.u
61
62     r_idx = rand(1:size(sol.t)[1])
63     r_angle = sol.t[r_idx]
64     # r_angle_list = [sol.t[r_idx] for n in 1:size(sol.t)[1]]
65
66     plot!(plt, xlabel="θ", ylabel="r(t)", title="Кривая погони. Частный случай", legend=:outerbottom)
67     savefig(plt, "lab2_" * string(flag) * "_θ")
68
69     plot!(plt, [0.0, 0.0], [a, r], label="Начальное катера движение по прямой", color=:brown, lw=0.4)
70     scatter!(plt, [0.0], [a], label="", mc=:brown, ms=0.4)
71     plot!(plt, [r_angle, r_angle], [0.0, sol.u[1]], label="Траектория лодки", color=:blue, lw=0.4)
72     scatter!(plt, [r_angle], [sol.u[1]], label="", mc=:blue, ms=0.01)
73     plot!(plt, [sol.t[1], sol.t[1]], [sol.u[1], sol.u[1]], label="Траектория катера", color=:red, lw=0.4)
74     savefig(plt, "lab2_" * string(flag) * "_1")
75
76     for i in 2:size(sol.t)[1]
77         # Добавление параметров в пространство
78         plot!(plt, [r_angle, r_angle], [sol.u[i-1], sol.u[i]], label="", color=:blue, lw=0.4)
79         scatter!(plt, [r_angle], [sol.u[i]], label="", mc=:blue, ms=0.01)
80
81         plot!(plt, [sol.t[i-1], sol.t[i]], [sol.u[i-1], sol.u[i]], label="", color=:red, lw=0.4)
82         scatter!(plt, [sol.t[i]], [sol.u[i]], label="", mc=:red, ms=0.01)
83
84         savefig(plt, "lab2_" * string(flag) * "_" * string(i))
85     end
86
87     return 0
88 end
89
90
91 the_chase_curve(6.2, n, 1)
92

```

Рис. 4.23: Код лабораторной работы на Julia. 2

2. Любуемся результатом (рис. 4.24, 4.25).

### Кривая погони. Частный случай



Рис. 4.24: Результат для первого случая

### Кривая погони. Частный случай



Рис. 4.25: Результат для второго случая

## 4.2.2 OpenModelica

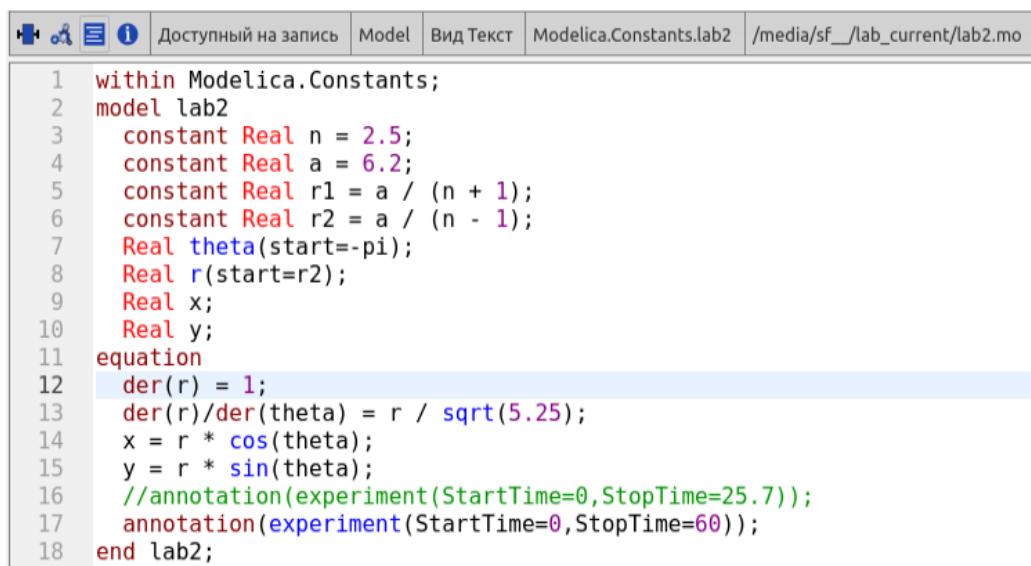
1. Пробуем написать программу, воспроизводящую модель на языке моделирования Modelica (рис. 4.26, 4.27).

```
within Modelica.Constants;  
model lab2  
  constant Real n = 2.5;  
  constant Real a = 6.2;  
  constant Real r1 = a / (n + 1);  
  constant Real r2 = a / (n - 1);  
  Real theta(start=-pi);  
  Real r(start=r2);  
  Real x;  
  Real y;  
equation  
  der(r) = 1;  
  der(r)/der(theta) = r / sqrt(5.25);  
  x = r * cos(theta);  
  y = r * sin(theta);  
  //annotation(experiment(StartTime=0,StopTime=25.7));  
  annotation(experiment(StartTime=0,StopTime=60));  
end lab2;
```



```
1 within Modelica.Constants;
2 model lab2
3 constant Real n = 2.5;
4 constant Real a = 6.2;
5 constant Real r1 = a / (n + 1);
6 constant Real r2 = a / (n - 1);
7 Real theta(start=0);
8 Real r(start=r1);
9 Real x;
10 Real y;
11 equation
12 der(r) = 1;
13 der(r)/der(theta) = r / sqrt(5.25);
14 x = r * cos(theta);
15 y = r * sin(theta);
16 annotation(experiment(StartTime=0,StopTime=25.7));
17 //annotation(experiment(StartTime=0,StopTime=60));
18 end lab2;
```

Рис. 4.26: Код лабораторной работы на Modelica для первого случая



```
1 within Modelica.Constants;
2 model lab2
3 constant Real n = 2.5;
4 constant Real a = 6.2;
5 constant Real r1 = a / (n + 1);
6 constant Real r2 = a / (n - 1);
7 Real theta(start=-pi);
8 Real r(start=r2);
9 Real x;
10 Real y;
11 equation
12 der(r) = 1;
13 der(r)/der(theta) = r / sqrt(5.25);
14 x = r * cos(theta);
15 y = r * sin(theta);
16 //annotation(experiment(StartTime=0,StopTime=25.7));
17 annotation(experiment(StartTime=0,StopTime=60));
18 end lab2;
```

Рис. 4.27: Код лабораторной работы на Modelica для второго случая

2. Смотрим на результат (рис. 4.28, 4.29).

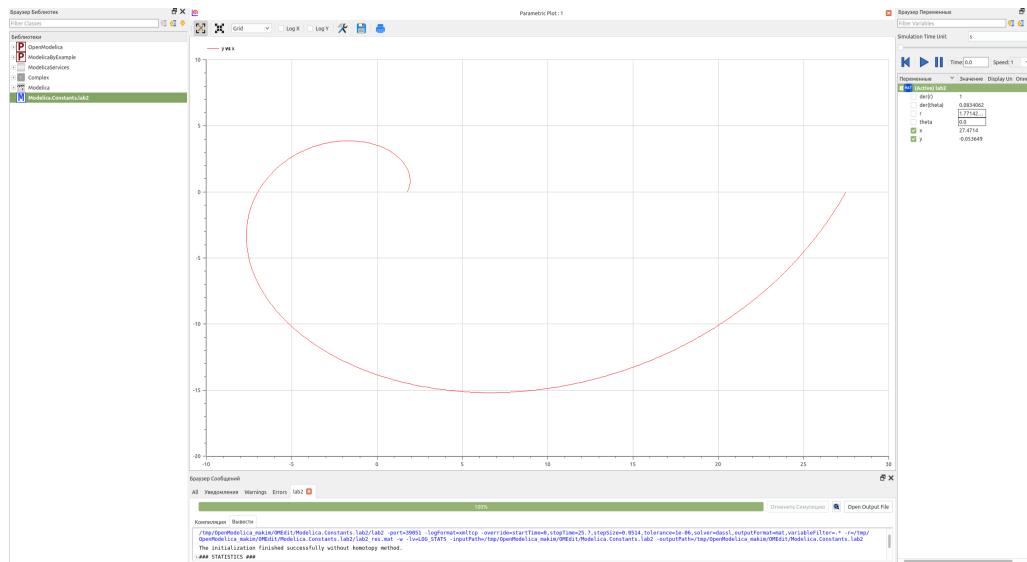


Рис. 4.28: Результат для первого случая

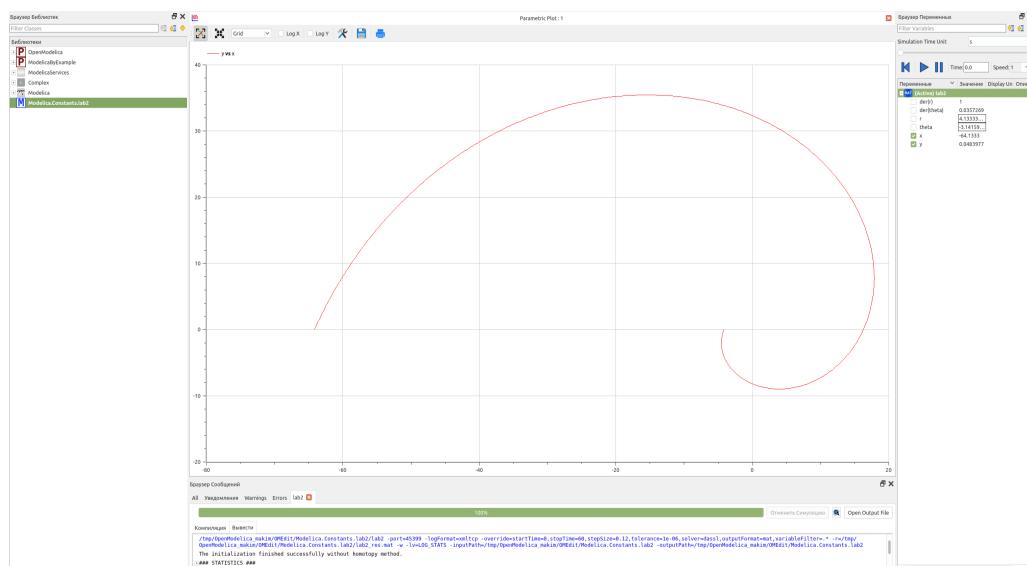


Рис. 4.29: Результат для второго случая

### 4.2.3 Математические вычисления

#### 1. Используемые вычисления (рис. 4.30).

DATE

$t = \frac{r_0}{2,5} = \frac{a - r_0}{\sqrt{3}} = \frac{a - r_0}{2,5\sqrt{3}}$   
 $2,5r_0 = a - r_0$   
 $3,5r_0 = a$   
 $r_0 = \frac{a}{3,5}$   
 $r_0 = \frac{6,2}{3,5}$   
 $\sqrt{v_r^2 + v_\theta^2} = \sqrt{v_r^2 + v_\theta^2}$   
 $(2,5\sqrt{3})^2 = \sqrt{v_r^2 + v_\theta^2}$   
 $\sqrt{v_r^2} = 6,25\sqrt{3}^2 - \sqrt{v_\theta^2}$   
 $\sqrt{v_r^2} = 5,25\sqrt{3}^2$   
 $\sqrt{v_r^2} = \sqrt{3,25} \sqrt{3}$   
 $\left\{ \begin{array}{l} \frac{dr}{dt} = \sqrt{v_r^2} \\ r \frac{d\theta}{dt} = \sqrt{6,25} \sqrt{3} \end{array} \right. \quad \left\{ \begin{array}{l} dt = \frac{dr}{\sqrt{v_r^2}} \\ dt = \frac{r d\theta}{\sqrt{3,25} \sqrt{3}} \end{array} \right.$   
 $\frac{dr}{\sqrt{3,25} \sqrt{3}} = \frac{r d\theta}{\sqrt{3,25} \sqrt{3}} \Rightarrow \frac{dr}{d\theta} = \frac{r}{\sqrt{3,25}}$

$t = \frac{r_1}{2,5} = \frac{r_1 - a}{\sqrt{3}} = \frac{r_1 - a}{2,5\sqrt{3}}$   
 $x - r$   
 $k - a$   
 $2,5r_1 = r_1 - a$   
 $r_1 = \frac{r_1 - a}{2,5}$   
 $\frac{x}{2,5} = \frac{k - x}{2,5}$   
 $\frac{x}{2,5} = x + k$   
 $2,5x = k$   
 $1,5x = k$   
 $3,5r_1 = a$   
 $1,5r_1 = a$   
 $r_1 = \frac{a}{1,5} \approx 1,77$   
 $r_1 = \frac{a}{1,5} \approx 4,15$

(1)

BRAUBERG

Рис. 4.30: Вычисления

## 5 Выводы

Ознакомился с базовым функционалом языка программирования Julia, а также интерактивной командной строкой REPL. Ознакомился с базовым функционалом языка моделирования Modelica и программным обеспечением OpenModelica. Используя эти средства, построил математическую модель, представляющую собой частный случай задачи о погоне.

**Сделал для себя вывод, что для решения данной задачи подходит язык программирования Julia и почти полностью не подходит язык моделирования Modelica.**

# Список литературы

1. Julia. Знакомство [Электронный ресурс]. Habr, 2018. URL: <https://habr.com/ru/post/423811/>.
2. Julia как язык программирования [Электронный ресурс]. MIVOCLLOUD LIMITED, 2022. URL: <https://www.mivocloud.com/ru/blog/Julia-as-a-programming-language>.
3. The Julia Programming Language [Электронный ресурс]. JuliaLang.org, 2022. URL: <https://julialang.org/>.
4. OpenModelica. Introduction [Электронный ресурс]. OpenModelica. URL: <https://openmodelica.org/>.
5. OpenModelica [Электронный ресурс]. Wikimedia Foundation, Inc., 2022. URL: <https://en.wikipedia.org/wiki/OpenModelica>.