|  |
| --- |
| МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ  федеральное государственное автономное образовательное учреждение высшего образования  «САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ  АЭРОКОСМИЧЕСКОГО ПРИБОРОСТРОЕНИЯ» |

Кафедра компьютерных технологий и программной инженерии

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ОТЧЁТ ПО ПРАКТИКЕ  ЗАЩИЩЁН С ОЦЕНКОЙ  Руководитель |  | | | |
| Ст. преп. |  |  |  | М.Д. Поляк |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

ОТЧЁТ ПО ПРАКТИКЕ

|  |  |  |
| --- | --- | --- |
| вид практики | производственная | |
| тип практики | По получению профессиональных умений и опыта профессиональной деятельности | |
| на тему индивидуального задания | | «Разработка программы для расчёта и вывода формулы |
| целевой функции из функционального уравнения» | | | |
|  | | | |

|  |  |
| --- | --- |
| выполнен | Фокин Артур Сафарович |
| фамилия, имя, отчество обучающегося в творительном падеже | |

|  |  |  |  |
| --- | --- | --- | --- |
| по направлению подготовки | 09.03.04 |  | Программная инженерия |
|  | код |  | наименование направления |
|  | | | |
| наименование направления | | | |
| направленности | 02 |  | Проектирование программных систем |
|  | код |  | наименование направленности |
|  | | | |
| наименование направленности | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Обучающийся группы № | 4134к |  |  |  | А.С. Фокин |
|  | номер |  | подпись, дата |  | инициалы, фамилия |

Санкт–Петербург 2024

**Задание**

Цель проекта: разработать программу, которая решает функциональные уравнения, с возможностью задавать любые правые части дифференциальных уравнений и целевую функцию u.

**Функциональные требования**:

Язык программирования:

* Программа написана на Python с использованием библиотеки SymPy для символьных вычислений.

Решение дифференциальных уравнений:

* Программа рассчитывает производные и функциональные уравнения.
* Выводит формулу целевой функции (цели)

Вывод результата:

* Программа выводит результат в формате LaTeX.
* Автоматическая генерация PDF-документа с подробным решением задачи и выводом всех промежуточных шагов.

Использование интерфейса командной строки:

* Пользователь взаимодействует с программой через ввод команд в консоли (например, ввод правых частей уравнений и цели).

Поддержка Latex:

* Программа автоматически компилирует решение задачи в PDF-файл, используя систему LaTeX, и выводит его в удобочитаемом виде.

**Тестовые данные**

**Запускаем программу:** указываем хотим мы ввести данные вручную или использовать базовые значения



Рисунок 1 – Запуск программы

Программа показывает значения которые используются по умолчанию в рамках задачи

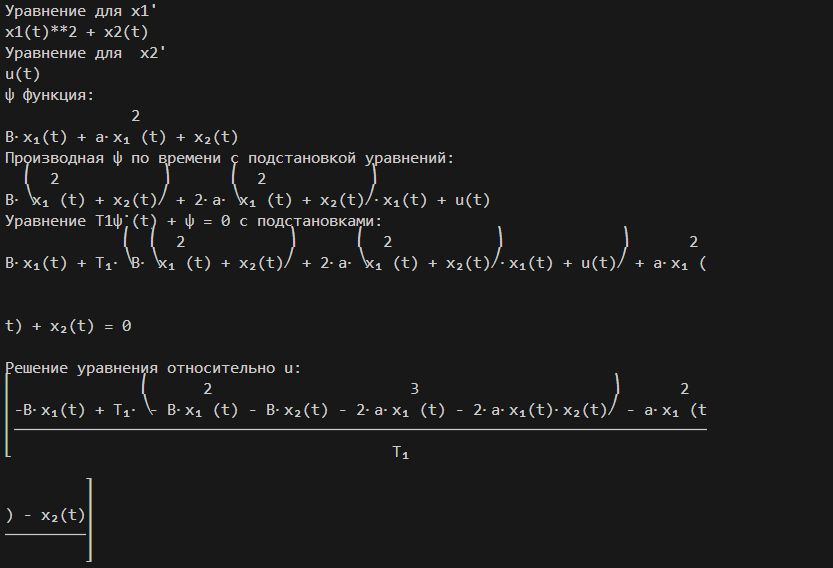


Рисунок 2 – Расчёты программы

Значения можно ввести вручную и задать цель:



Рисунок 3 – Ручной ввод дифференциальных уравнений

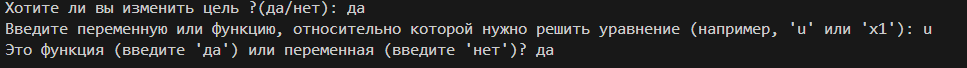


Рисунок 4 – Ручной ввод цели

После этого программа генерирует pdf файл



Рисунок 5 – Генерация pdf файла

В pdf файле можно увидеть решение задачи

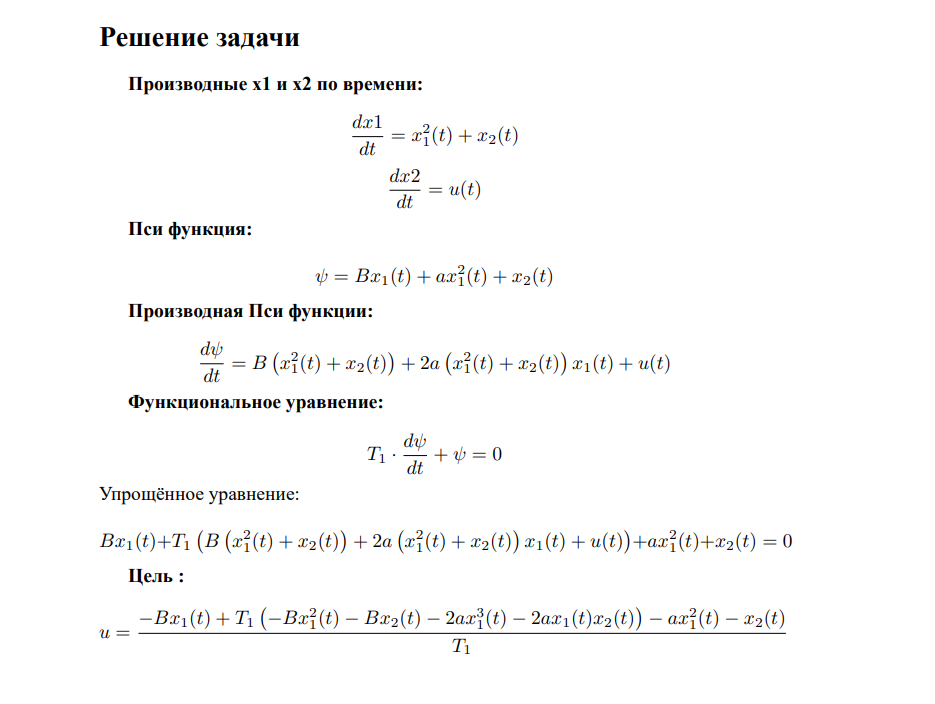


Рисунок 6 – Сгенерированный pdf файл

**Код**

import os

import subprocess

from sympy import symbols, Function, Eq, diff, simplify, solve, pprint,Abs,dsolve,Subs,latex

# Определение символов

t = symbols('t')

x1 = Function('x1')(t)

x2 = Function('x2')(t)

u = Function('u')(t)

B, a, T1 = symbols('B a T1')

# module задать модуль отдельно

# Запрос у пользователя, хочет ли он ввести свои функции

use\_custom\_functions = input("Хотите ли вы ввести свои правые части для дифференциальных уравнений и пси функцию? (да/нет): ")

if use\_custom\_functions.lower() == 'да':

    # Запрос у пользователя его функций

    x1\_uravnenie\_str = input("Введите правую часть для x1'(t) (используйте синтаксис SymPy): ")

    x2\_uravnenie\_str = input("Введите правую часть для x2'(t) (используйте синтаксис SymPy): ")

    psi\_function\_str = input("Введите выражение для ψ (используйте синтаксис SymPy): ")

    # Преобразование строковых функций пользователя в выражения SymPy

    x1\_uravnenie = eval(x1\_uravnenie\_str)

    x2\_uravnenie = eval(x2\_uravnenie\_str)

    ψ\_psi = eval(psi\_function\_str)

else:

    # Уравнения для x1' и x2' из примера

    x1\_uravnenie = x1\*\*2 + x2

    x2\_uravnenie = u

    ψ\_psi = x2 + B\*x1 + a\*x1\*x1  # Примерное выражение для ψ

print( "Уравнение для x1' ")

print(x1\_uravnenie)

print( "Уравнение для  x2' ")

print(x2\_uravnenie)

print("ψ функция:")

pprint(ψ\_psi)

# Производная ψ по времени с учётом того, что x1 и x2 - функции времени

ψ\_psi\_diff = diff(ψ\_psi, t)

# Подстановка уравнений для x1' и x2'

ψ\_psi\_diff = ψ\_psi\_diff.subs({diff(x1, t): x1\_uravnenie, diff(x2, t): x2\_uravnenie})

# Вывод производной ψ по времени

print("Производная ψ по времени с подстановкой уравнений:")

pprint(ψ\_psi\_diff)

# Выражаем уравнение T1 \* ψ˙(t) + ψ = 0

equation = Eq(T1 \* ψ\_psi\_diff + ψ\_psi, 0)

# Упрощение уравнения

simplified\_equation = simplify(equation)

# Вывод уравнения

print("Уравнение T1ψ˙(t) + ψ = 0 с подстановками:")

pprint(simplified\_equation)

vopr\_u = input("Хотите ли вы изменить цель ?(да/нет): ")

if vopr\_u.lower() == 'да':

    # Спросим пользователя, какую переменную или функцию использовать

    target\_var = input("Введите переменную или функцию, относительно которой нужно решить уравнение (например, 'u' или 'x1'): ")

    # Спросим, является ли это функцией или переменной

    is\_function = input(f"Это функция (введите 'да') или переменная (введите 'нет')? ").lower()

    # Преобразуем введенный текст в символьную переменную или функцию

    if is\_function == 'да':

        target\_var\_sym = Function(target\_var)(t)

    else:

        target\_var\_sym = symbols(target\_var)

    # Проверка на наличие переменной в уравнении

    equation\_atoms = simplified\_equation.atoms(Function, symbols)  # Собираем все функции и символы

    if target\_var\_sym in equation\_atoms:

    # Решение уравнения относительно выбранной переменной

        solved\_target = solve(simplified\_equation, target\_var\_sym)

        print(f"\nРешение уравнения относительно {target\_var}:")

        pprint(solved\_target)

    else:

        print(f"Переменная {target\_var} отсутствует в уравнении.")

else:

     # Если пользователь выбрал "нет", решаем уравнение относительно функции u

    target\_var\_sym = u  # Решаем уравнение относительно u

    solved\_target = solve(simplified\_equation, target\_var\_sym)

    # Решение уравнения относительно выбранной переменной или функции

    solved\_target = solve(simplified\_equation, target\_var\_sym)

    print(f"\nРешение уравнения относительно u:")

    pprint(solved\_target)

# Получение LaTeX-кода для уравнения

latex\_output = latex(solved\_target)

print("\nВывод в LaTeX формате:")

print(latex\_output)

latex\_document = r"""

\documentclass{article}

\usepackage{fontspec}         % Поддержка шрифтов Unicode

\usepackage{polyglossia}       % Поддержка многоязычности

\setmainlanguage{russian}      % Основной язык документа

\setmainfont{Times New Roman}  % Установка шрифта, который поддерживает Unicode

\begin{document}

\section\*{Решение задачи}

\textbf{Производные x1 и x2 по времени:} \\

\[

\frac{dx1}{dt} = """ + latex(x1\_uravnenie) + r"""

\]

\[

\frac{dx2}{dt} = """ + latex(x2\_uravnenie) + r"""

\]

\textbf{Пси функция:} \\

\[

\psi = """ + latex(ψ\_psi) + r"""

\]

\textbf{Производная Пси функции:} \\

\[

\frac{d\psi}{dt} = """ + latex(ψ\_psi\_diff) + r"""

\]

\textbf{Функциональное уравнение:} \\

\[

T\_1 \cdot \frac{d\psi}{dt} + \psi = 0

\]

Упрощённое уравнение: \\

\[

""" + latex(simplified\_equation) + r"""

\]

\textbf{Цель :} \\

\[

u = """ + latex(solved\_target[0]) + r"""

\]

\end{document}

"""

# Запись в файл с кодировкой UTF-8

with open("solution.tex", "w", encoding="utf-8") as f:

    f.write(latex\_document)

# Компиляция LaTeX в PDF с помощью xelatex

subprocess.run(["xelatex", "solution.tex"])

# Удаление временных файлов

temp\_files = ["solution.aux", "solution.log", "solution.synctex.gz"]

for temp\_file in temp\_files:

    if os.path.exists(temp\_file):

        os.remove(temp\_file)

print("PDF файл solution.pdf успешно создан!")

**Вывод**

Проект программы для расчёта формулы целевой функции u успешно реализован. В ходе работы были выполнены следующие задачи:

**Производительность:**

Программа позволяет задавать произвольные правые части дифференциальных уравнений, что делает её гибкой для различных типов задач.

Автоматическая генерация LaTeX-документов с расчётами и выводом целевой функции ускоряет процесс подготовки отчётов и представления результатов.

**Простота использования**:

Взаимодействие через командную строку позволяет пользователю интуитивно вводить необходимые данные для решения задачи.

Программа автоматически компилирует PDF-документ с результатами, что упрощает процесс получения итогового решения.

**Масштабируемость**:

Программа легко расширяется для работы с более сложными функциональными уравнениями и дополнительными функциями.

Проект предоставляет удобный инструмент для расчёта и вывода целевой функции, что делает его полезным в рамках различных математических и инженерных задач.

**Источники**

* https://www.sympy.org/en/index.html
* http://scp.ictis.sfedu.ru/study/2-6.htmlКонец формы
* https://docs.google.com/document/d/1HYbbrJyv\_rf\_rrufKAKyqvhEdYJN3KOi/edit