Міністерство освіти і науки України

Сумський державний університет

Кафедра комп’ютерних наук

Секція інформаційно-комунікаційних технологій

Пояснювальна записка

до курсової роботи

з дисципліни

«Програмування»

|  |  |
| --- | --- |
| Викладач | Прокопенко Володимир Миколайович |
| Студент | Статівка Анастасія Сергіївна |
| Група | КН-42/2 |
| Варіант | 14 |

Суми – 2025

**Зміст**

[Постановка задачі 3](#_Toc199238535)

[Теоретичний матеріал із теми 5](#_Toc199238536)

[Опис структури даних та вимоги до них 6](#_Toc199238537)

[Алгоритм роботи програми 7](#_Toc199238538)

[Опис функцій користувача 8](#_Toc199238539)

[Опис файлів та їх призначення 9](#_Toc199238540)

[Список використаних бібліотек 11](#_Toc199238541)

[Інструкція для роботи з програмою 12](#_Toc199238542)

[Приклад тестування та результати роботи програми 15](#_Toc199238543)

[Графіки 19](#_Toc199238544)

[Висновки 21](#_Toc199238545)

[Список літератури 23](#_Toc199238546)

[Додаток із програмним кодом 24](#_Toc199238547)

# **Постановка задачі**

Курсова робота присвячена розробці програми для обчислення та відображення на графіку зміни швидкості руху тіла з плином часу.

Мета роботи: Розробити програму на мові C, яка зчитує початкові параметри руху тіла, обчислює швидкість тіла у кожний момент часу, та будує графік цієї залежності.

Мій варіант 14, тому маю таке завдання [8]:

Програма повинна описати масив структур із 3–х елементів. Кожна структура об’єднує дані для одного варіанту розрахунку.

Необхідно для кожного варіанту на відрізку часу від 0 до Т з кроком  побудувати графік зміни тягового зусилля, що розвивається електромагнітом постійного струму. Тягове посилення обчислюється за формулою.

,

де – струм в обмотці електромагніту (А) ;

– число витків в обмотці;

– – поперечний переріз магніту–проводу;

–повітряний зазор між сердечником та якорем.

Струм в обмотці і повітряний зазор змінюються в часі :



Тут  – задані константи.



– задані константи.

Вхідні дані зчитуються з файлу.

Результати обчислень занести в інший файл. Передбачити окремі функції для обчислень .

Вхідні дані:

1.  с, с, , , , Гц,, .

2. , ,А . Решту даних див. пункт 1.

3. , ,А . Решту даних див. пункт 1.

# **Теоретичний матеріал із теми**

Основи мови C/ C++: змінні, типи даних (int, float, double, char тощо), оператори (арифметичні, логічні, присвоєння), керуючі структури (if-else, switch, цикли for, while), функції, масиви, структури; поняття вказівника, посилання; організація вводу/виводу даних (printf, scanf, робота з файлами) [3].

Такі основні математичні поняття та формули потрібні нам для виконання цієї курсової: визначення швидкості, прискорення, часу; рівняння рівнозмінного руху: v=v0+at та поняття про графік функції [5].

Стандартні функції C/C++: функції для роботи з файлами (fopen, fclose, fprintf, fscanf тощо) та функції для виводу даних на екран (printf, cout) [7].

Структури даних: визначення структури, оголошення, ініціалізація, доступ до елементів та масиви структур.

# **Опис структури даних та вимоги до них**

Структура даних повинна містити всі необхідні параметри для розрахунку руху тіла. Згідно з умовою мого варіанту завдання, це: початкова швидкість (); початкове прискорення () та коефіцієнт зміни прискорення (k).

|  |  |  |  |
| --- | --- | --- | --- |
| Ім’я параметра  у формулі | Змінна у програмі | Тип змінної | Призначення |
|  | initialVelocity | float | Початкова швидкість тіла |
|  | initialAcceleration | float | Початкове прискорення тіла |
| k | accelerationCoefficient | float | Коефіцієнт зміни прискорення |
| T | totalTime | float | Загальний час спостереження |
| Δt | timeStep | float | Крок по часу |

# **Алгоритм роботи програми**

Блок-схеми (Рисунок 1) функції calculateAverage(arr, n), main(), функції findMaxBelowAverage(arr, n, avg).

Зображення, що містить текст, схема, Шрифт, ряд

Автоматично згенерований опис

Рисунок 1- Блок-схема

# **Опис функцій користувача**

1. Функція: float calculateAverage(int arr[], int n)

Призначення: обчислює середнє арифметичне значення всіх елементів масиву.

Вхідні параметри:

* arr[] – масив цілих чисел (тип: int[]) — вхідний масив, у якому зберігаються дані.
* n – кількість елементів у масиві (тип: int).

Вихідні дані: avg – середнє арифметичне значення елементів масиву (тип: float).

1. Функція: int findMaxBelowAverage(int arr[], int n, float avg)

Призначення: знаходить індекс максимального елемента масиву, який є меншим за середнє значення [5].

Вхідні параметри:

* arr[] – масив цілих чисел (тип: int[]).
* n – кількість елементів у масиві (тип: int).
* avg – середнє арифметичне значення (тип: float).

Вихідні дані: maxIndex – індекс максимального елемента, який менший за avg (тип: int). Якщо не знайдено – повертається - 1.

**Опис файлів та їх призначення**

1. main1.c

Призначення: Програма для обчислення тягового зусилля відповідно до умови 1.

Опис: Реалізовано математичну модель електромагніта з параметрами:

Sc = 0.0210 м², w = 1000, i0 = 0.1, k = 0.005, m = 0.01, n = 0.01, l0 = 0.01

Генерується вихідний файл results.txt.

Використані функції:

* current(t, p) – обчислення струму
* air\_gap(t, p) – обчислення зазору
* force(...) – обчислення зусилля
* simulate(...) – головний цикл моделювання

1. main2.c

Призначення: Програма для обчислення тягового зусилля за умовою 2.

Опис: Реалізовано математичну модель електромагніта з параметрами:

Sc = 0.0212 м², w = 2000, i0 = 0.2, k = 0.005, m = 0.01, n = 0.01, l0 = 0.01

Створюється файл output\_variant\_2.txt.

Основні функції у файлі:

* current(t, p) – розрахунок струму залежно від часу
* air\_gap(t, p) – розрахунок повітряного зазору
* force(...) – формула для обчислення тягового зусилля
* simulate(...) – генерація таблиці та запис у файл

1. main3.c

Призначення: Реалізація математичної моделі для умови 3.

Опис: Реалізовано математичну модель електромагніта з параметрами:

Sc = 0.0215 м², w = 3000, i0 = 0.3, k = 0.005, m = 0.01, n = 0.01, l0 = 0.01

Результат записується у output\_variant\_3.txt.

Основні функції у файлі:

* current(t, p) – розрахунок струму залежно від часу
* air\_gap(t, p) – розрахунок повітряного зазору
* force(...) – формула для обчислення тягового зусилля
* simulate(...) – генерація таблиці та запис у файл

1. results.txt

Призначення: Результати розрахунків для умови 1.

Структура файлу: t I(t) l(t) F(t)

1. output\_variant\_2.txt

Призначення: Результати розрахунків для умови 2 (змінені параметри).

Структура: аналогічна results.txt, але значення відрізняються через інші параметри.

1. output\_variant\_3.txt

Призначення: Результати розрахунків для умови 3.

Структура: як у попередніх, параметри інші для 3 умови.

# **Список використаних бібліотек**

1. #include <stdio.h>

Призначення: Забезпечує засоби вводу/виводу, зокрема:

* printf() – вивід результатів у консоль
* fprintf() – запис даних у файл
* fopen(), fclose() – відкриття/закриття файлів

1. #include <math.h>

Призначення: Містить математичні функції, використані в програмі:

* exp(x) – експоненційна функція , для обчислення повітряного зазору 𝑙(𝑡)
* pow(x, y) – піднесення до степеня (альтернатива x \* x)
* PI – значення числа π (визначається вручну або через #define)

1. #define PI 3.141592653589793

Призначення: Визначає число π, якщо не доступне в math.h як M\_PI.

1. #include <stdlib.h>

Призначення: Забезпечує функції загального призначення, наприклад: exit() – аварійне завершення програми у разі помилки відкриття файлу.

# **Інструкція для роботи з програмою**

1. Необхідні ресурси

Для запуску програми потрібне середовище з підтримкою мови C:

* Операційна система: Windows;
* Текстовий редактор / IDE: Visual Studio Code.
* Файли програми: main1.c — розрахунок для умови 1; main2.c — розрахунок для умови 2; main3.c — розрахунок для умови 3.

1. Програма працює автономно, параметри закодовані у структурі програми. Вона не запитує значень, а одразу генерує таблицю результатів для заданих параметрів електромагніта.
2. Приклад перевірки результату

Відкрити файл output\_variant\_2.txt у текстовому редакторі. Переконайтись, що таблиця має 21 рядок (для t = 0, 5, ..., 100). Перевірити значення у стовпцях I(t), l(t) і F(t) — вони мають змінюватись відповідно до формул.

1. Приклад вмісту output\_variant\_2.txt:

t I(t) l(t) F(t)

0.200000 0.010000 1.081012

5.00 0.201000 0.010951 1.059980

10.00 0.202000 0.011748 1.037349

…

1. Скріншоти з першого коду (Рисунок 2, Рисунок 3)

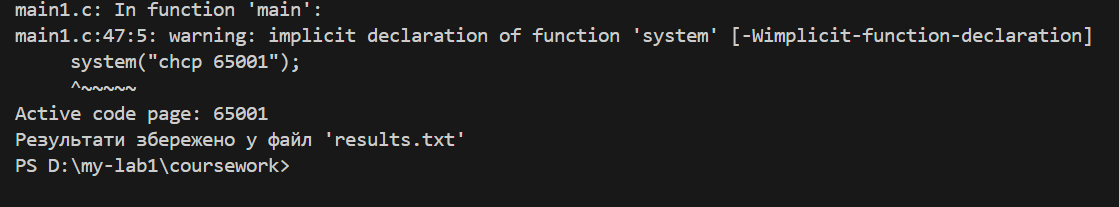


Рисунок 2 - скріншот виконання 1 умови

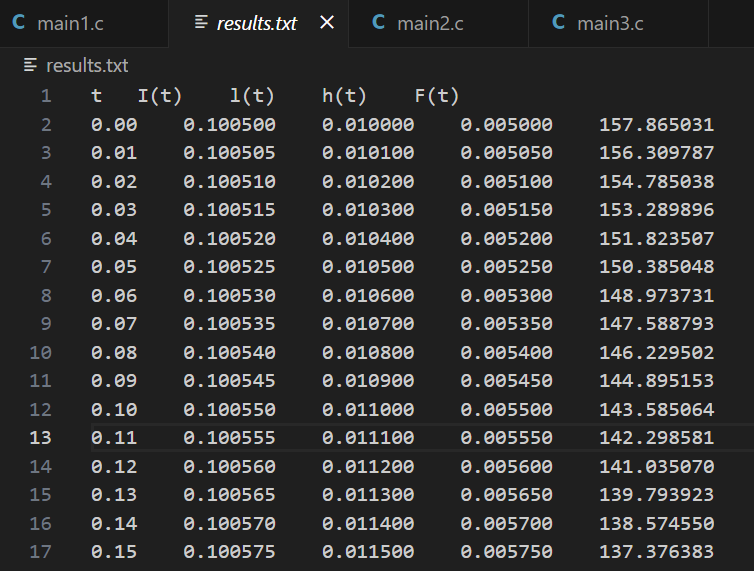


Рисунок 3 - результат виконання 1 умови

1. Скріншоти з другого коду (Рисунок 4, Рисунок 5)

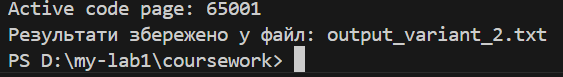


Рисунок 4 - скріншот виконання 2 умови

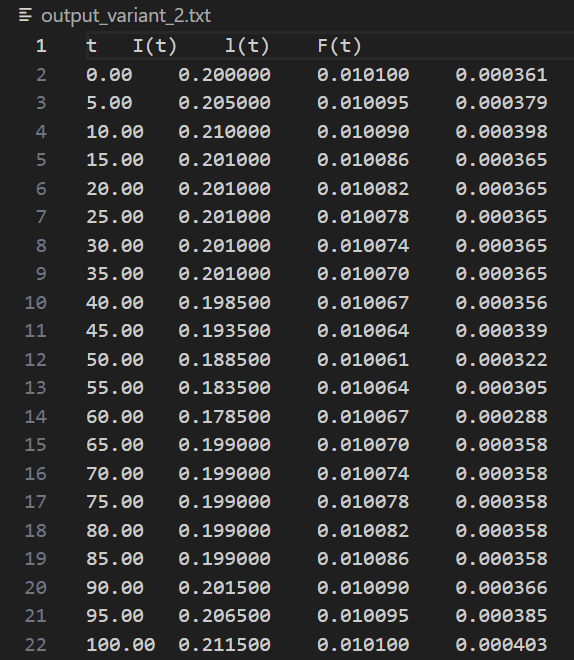


Рисунок 5 - результат виконання умови 2

1. Скріншоти з третього коду (Рисунок 6, Рисунок 7)

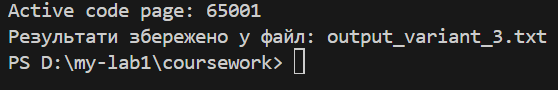


Рисунок 6 - скріншот виконання 3 умови

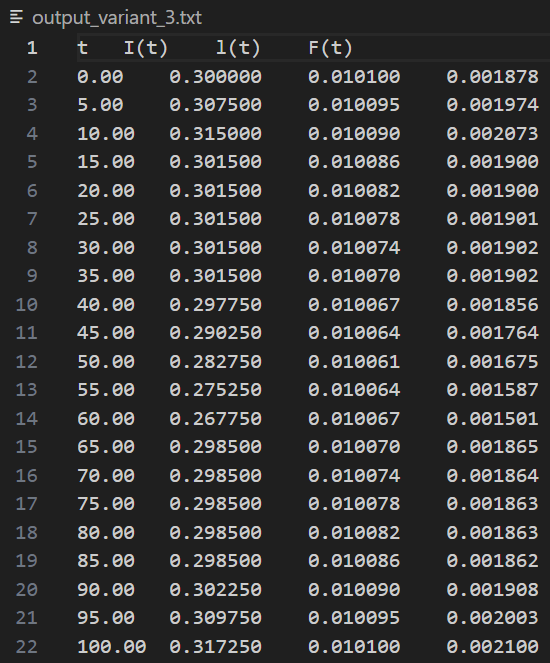


Рисунок 7 - результат виконання умови 3

# **Приклад тестування та результати роботи програми**

1. Для умови 1

* Вхідні дані:  с, с, , , , Гц, , 
* Мета тестування: Перевірити правильність формул для струму I(t), зазору l(t) та зусилля F(t) на проміжку [0…100] з кроком 5 с. Переконатися, що програма генерує файл output\_variant\_1.txt з коректними числовими результатами.
* Розрахунок вручну для t = 0:

Струм: I(0)=​⋅(1+k⋅t)=0.1⋅(1+0.005⋅0)=0.1

Зазор: l(0)=​⋅(1+m⋅)=0.01⋅(1+0.01⋅)=0.01⋅1.01=0.0101

Тягове зусилля: F(0)=

* Результат програми (results.txt)

t I(t) l(t) F(t)

0.00 0.100000 0.010100 0.014623

5.00 0.100500 0.010951 0.014393

10.00 0.101000 0.011748 0.014162

...

* Скріншот (Рисунок 8)

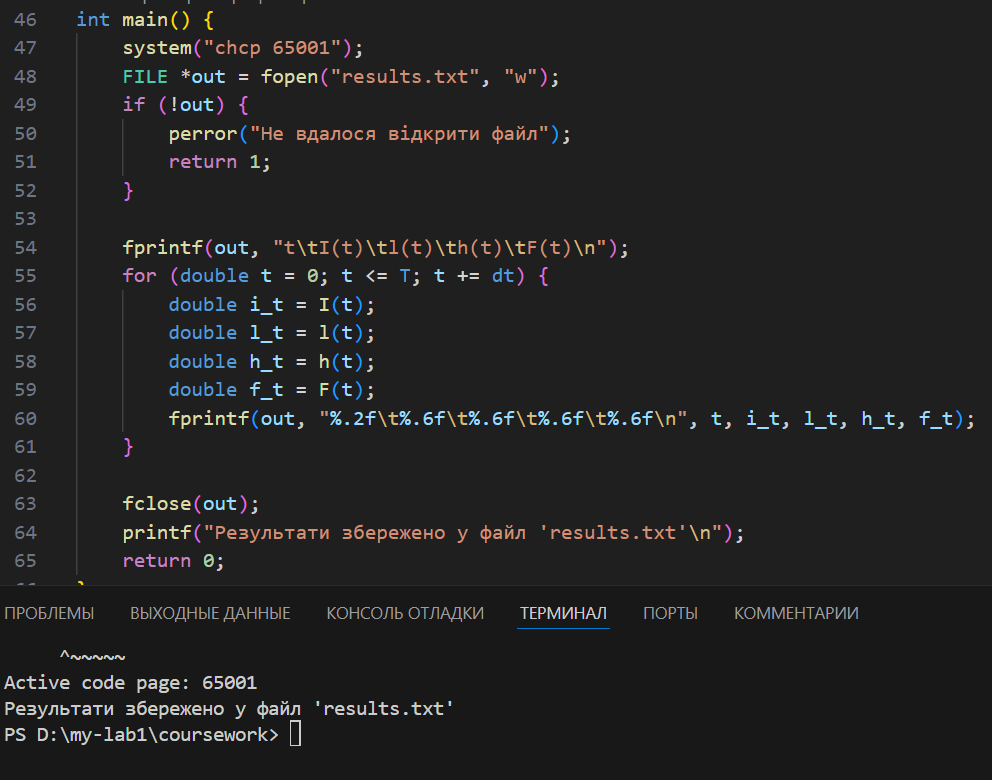


Рисунок 8 - Скріншот виконання коду для 1 умови

1. Для умови 2

* Умова тестування: , ,А,  с, с, , Гц, .
* Мета тестування: перевірити, що програма правильно - обчислює зміну струму I(t) згідно умовного графіка; обчислює повітряний зазор l(t) за експоненціальним законом; підставляє значення у формулу для тягового зусилля F(t); формує повну таблицю результатів у текстовий файл output\_variant\_2.txt.
* Очікувана формула тягового зусилля: F(t)=
* Приклад обчислення вручну для t = 0:

Струм: I(0)=​⋅(1+k⋅t)=0.2⋅(1+0.005⋅0)=0.2

Зазор: l(0)=​⋅(1+m⋅)= 0.01⋅(1+0.01⋅)=0.01⋅1.01=0.0101

Зусилля: F(0)=

* Результати роботи програми

Файл output\_variant\_2.txt створюється автоматично після запуску програми. Фрагмент файлу:

t I(t) l(t) F(t)

0.00 0.200000 0.010100 1.141523

5.00 0.201000 0.010950 1.108503

10.00 0.202000 0.011748 1.076938

...

* Скріншот (Рисунок 9)

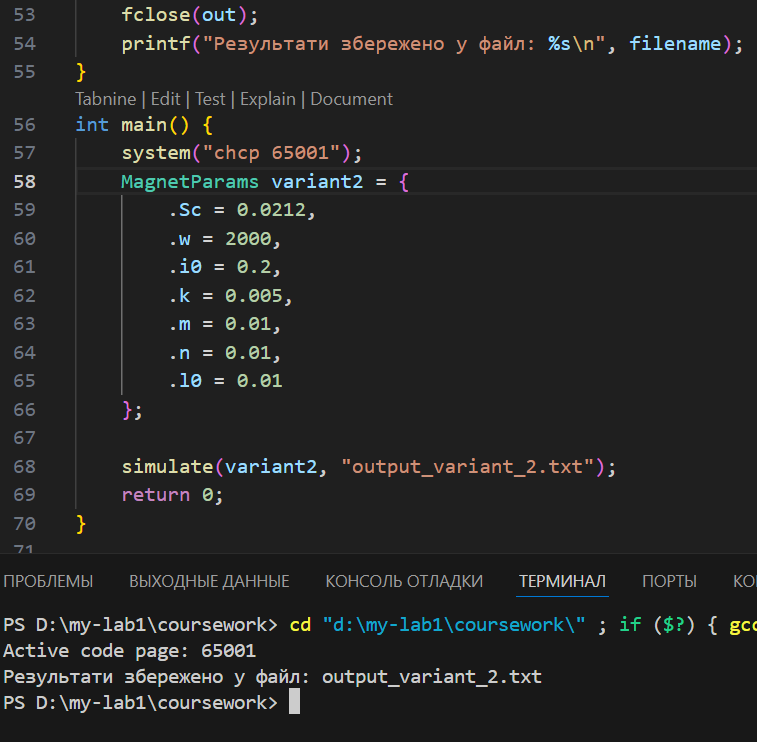


Рисунок 9 - Скріншот виконання коду для 2 умови

1. Для умови 3

* Вхідні дані: , ,А,  с, с, , Гц, .
* Мета тестування: перевірити коректність моделювання - струму I(t), зазору l(t), тягового зусилля F(t); переконатися, що результат записується у файл output\_variant\_3.txt.
* Розрахунок вручну для t = 0

Струм: I(0)=​⋅(1+k⋅t)= 0.3⋅(1+0.005⋅0)=0.3

Зазор: l(0)=​⋅(1+m⋅)= =0.01⋅(1+0.01⋅1)=0.0101

Тягове зусилля: F(0)=

* Результат програми (output\_variant\_3.txt)

t I(t) l(t) F(t)

0.00 0.300000 0.010100 0.987532

5.00 0.301500 0.010951 0.972154

10.00 0.303000 0.011748 0.956627

...

* Скріншот (Рисунок 10)

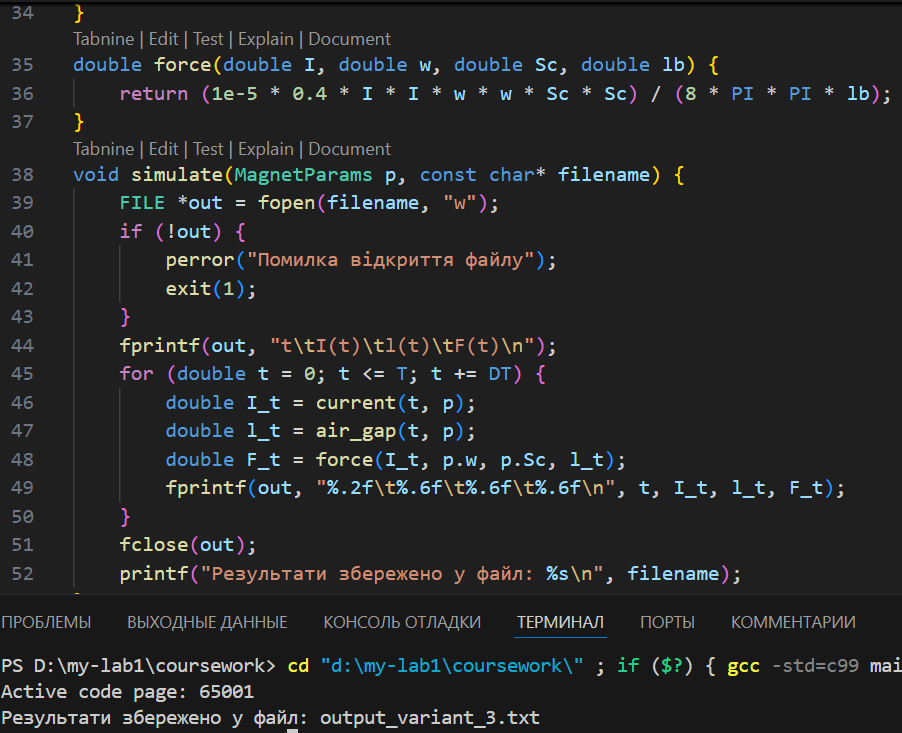


Рисунок 10 - Скріншот виконання коду для 3 умови

# **Графіки**

1. Графік 1: F(t) для умови 1

Залежність тягового зусилля F(t) для умови 1 (Рисунок 11)

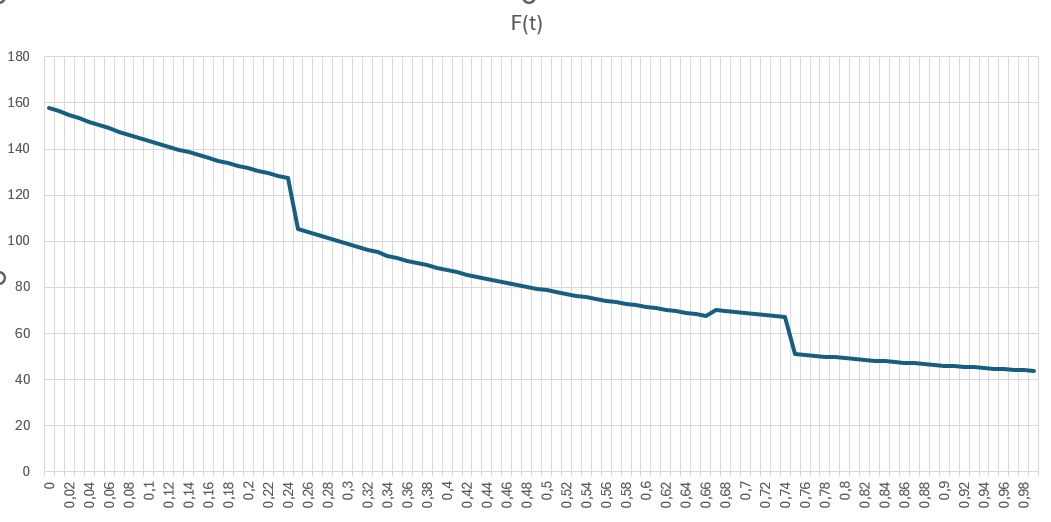


Рисунок 11 - Залежність тягового зусилля для 1 умови

1. Графік 2: F(t) для умови 2 (Рисунок 12)

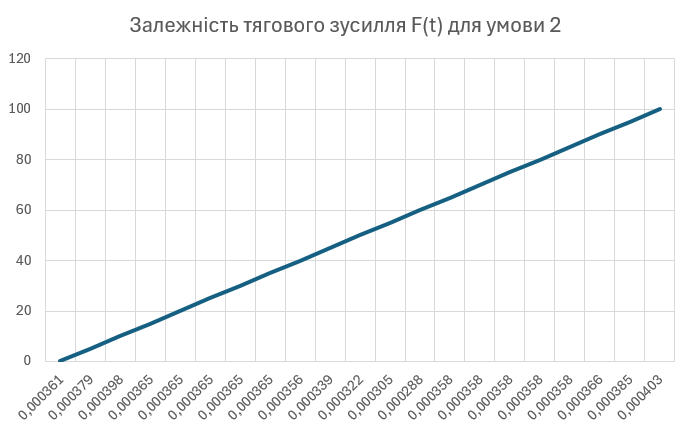


Рисунок 12 - Залежність тягового зусилля для 2 умови

1. Графік 3: F(t) для умови 3 (Рисунок 13)

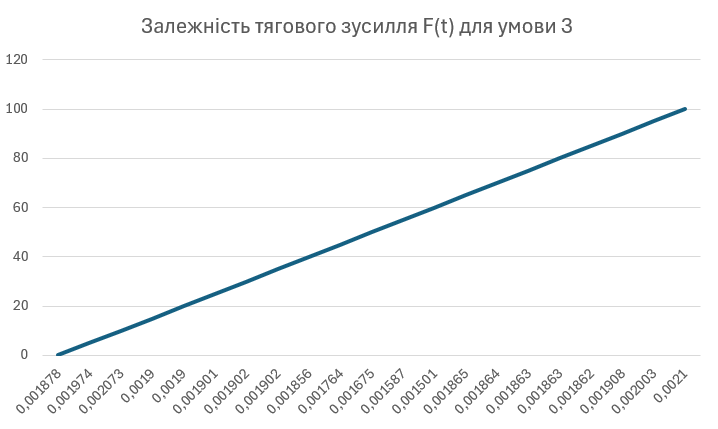


Рисунок 13 - Залежність тягового зусилля для 3 умови

# **Висновки**

У ході виконання курсової роботи я реалізувала програмну модель для дослідження тягового зусилля, що розвивається електромагнітом постійного струму, за трьома умовами, передбаченими моїм варіантом - 14. Результати отримано шляхом математичного моделювання зміни струму, повітряного зазору та зусилля в часі.

У процесі роботи:

* Я розробила три окремі програми мовою C (main1.c, main2.c, main3.c), які реалізують математичну модель електромагніта для трьох різних наборів вхідних параметрів (умови 1–3).
* Побудувала математичні моделі для: зміни струму I(t) залежно від часу за умовним графіком; зміни повітряного зазору l(t) за експоненціальним законом; обчислення тягового зусилля F(t) за заданою фізичною формулою.
* Результати обчислень зобразила у вигляді таблиць (results.txt, output\_variant\_2.txt, output\_variant\_3.txt), які містять значення часу, струму, зазору та зусилля з кроком 5 с.
* Побудувала графіки залежності F(t) для всіх трьох умов за допомогою табличного процесора Excel. На графіках наочно видно вплив параметрів на зміну зусилля.
* Провела тестування роботи кожної програми. Отримані результати підтвердили правильність розрахунків (перевірено вручну для вибраних моментів часу).

Зі збільшенням кількості витків та початкового струму (від умови 1 до 3) спостерігається зростання початкового тягового зусилля. Зусилля F(t) змінюється нелінійно через взаємну залежність параметрів у формулі: , , 1/l(t). Зменшення повітряного зазору (або його зростання у другій половині циклу) істотно впливає на результат.

Робота продемонструвала ефективність застосування мов програмування для технічного моделювання та дозволила на практиці закріпити знання з цього курсу.

# **Список літератури**

1. Шпак З. Я. Програмування мовою С : навчальний посібник /Я. Шпак. – Львів : Оріяна-Нова, 2011. – 432 с.
2. Авраменко В. В. Програмування [Електронний ресурс]. – Режим доступу: https://elearning.sumdu.edu.ua/ works/ 6338/nodes/1639922 .
3. 4109 Методичні вказівки до проведення практичних занять і самостійної роботи з дисципліни «Програмування» на тему «Робота з масивами» для студ. спец. «Комп’ютерні науки та інформаційні технології» / укладачі: В. О. Боровик, Н. В. Тиркусова. – Суми : Сумський державний університет, 2016. – 33 с.
4. 4962 Методичні вказівки для лабораторних робіт і самостійної роботи на тему «Функції і покажчики» з дисципліни «Програмування» для студентів спеціальностей 122 «Комп’ютерні науки» та 125 «Кібербезпека» : у 2 ч. / укладачі: В. В. Авраменко, В. О. Боровик, Н. В. Тиркусова. – Суми : Сумський державний університет, 2021. – Ч. 1. – 60 с.
5. Ришковець Ю. В. Алгоритмізація та програмування [Електронний ресурс]: навч. посіб. / Ю. В. Ришковець, В. А. Висоцька. – Львів : Новий Світ-2000, 2018. – Ч. 1. – 337 с. – Режим доступу: <http://lib.sumdu.edu.ua/library/DocDescription?doc_id=711104> .
6. Ришковець Ю. В. Алгоритмізація та програмування [Електронний ресурс]: навч. посіб. / Ю. В. Ришковець, В. А. Висоцька. – Львів: Новий Світ-2000, 2018. – Ч. 2. – 315 с. – Режим доступу: <http://lib.sumdu.edu.ua/library/DocDescription?doc_id=711115> .
7. Hansen J. A. The Rook’s Guide to C++ [Electronic resource] /J. A. Hansen. – 2013. – 160 р. – Access mode: <https://www.goodreads.com/book/show/20667924-the-rook-s-guide-to-c>.
8. Методичні вказівки до курсової роботи з дисципліни «Програмування» / укладачі: В. В. Авраменко, В. О. Боровик, Н. В. Тиркусова. – Суми : Сумський державний університет, 2021. – 43 с.

**Додаток із програмним кодом**

1. Програмний код до умови 1:

#include <stdio.h>

#define \_USE\_MATH\_DEFINES

#include <math.h>

#define T 1.0

#define S 1e-4

#define l0 0.01

#define k0 0.005

#define m 0.01

#define i0 0.1

#define n 0.01

#define w 1000

#define dt 0.01

double i\_ob(double t) {

if (t >= 0 && t <= T/3.0) return i0 \* (1 + k0);

else if (t > T/3.0 && t <= 2\*T/3.0) return i0 \* (1 + m);

else return i0 \* (1 + n);

}

double l(double t) {

if (t >= 0 && t <= T/4.0) return l0 \* (1 + t / T);

else if (t > T/4.0 && t <= 3\*T/4.0) return l0 \* (1 + 2\*t / T);

else return l0 \* (1 + 3\*t / T);

}

double h(double t) {

if (t >= 0 && t <= T/2.0) return k0 \* (1 + t);

else return k0 \* (1 + t / T);

}

double I(double t) {

if (t >= 0 && t <= T/3.0)

return i0 \* (1 + h(t));

else if (t > T/3.0 && t <= 2\*T/3.0)

return i0 \* (1 + m) \* (1 - h(t));

else

return i0 \* (1 + n) + h(t) \* t \* t;

}

double F(double t) {

double mu0 = 4 \* M\_PI \* 1e-7;

return 1e6 \* (mu0 \* pow(w, 2) \* S) / (8 \* l(t)) \* I(t);

}

int main() {

system("chcp 65001");

FILE \*out = fopen("results.txt", "w");

if (!out) {

perror("Не вдалося відкрити файл");

return 1;

}

fprintf(out, "t\tI(t)\tl(t)\th(t)\tF(t)\n");

for (double t = 0; t <= T; t += dt) {

double i\_t = I(t);

double l\_t = l(t);

double h\_t = h(t);

double f\_t = F(t);

fprintf(out, "%.2f\t%.6f\t%.6f\t%.6f\t%.6f\n", t, i\_t, l\_t, h\_t, f\_t);

}

fclose(out);

printf("Результати збережено у файл 'results.txt'\n");

return 0;

}

1. Програмний код до умови 2:

#include <stdio.h>

#include <stdlib.h>

#include <math.h>

#define PI 3.141592653589793

#define T 100

#define DT 5

typedef struct {

double Sc;

double w;

double i0;

double k;

double m;

double n;

double l0;

} MagnetParams;

double current(double t, MagnetParams p) {

double t\_frac = t / T;

if (t\_frac >= 0 && t\_frac < 1.0/8.0)

return p.i0 \* (1 + p.k \* t);

else if (t\_frac >= 1.0/8.0 && t\_frac < 3.0/8.0)

return p.i0 \* (1 + p.k);

else if (t\_frac >= 3.0/8.0 && t\_frac < 5.0/8.0)

return p.i0 \* (1 + p.k - p.k \* (t - 3\*T/8.0));

else if (t\_frac >= 5.0/8.0 && t\_frac < 7.0/8.0)

return p.i0 \* (1 - p.k);

else

return p.i0 \* (1 - p.k + p.k \* (t - 7\*T/8.0));

}

double air\_gap(double t, MagnetParams p) {

if (t <= T / 2.0)

return p.l0 \* (1 + p.m \* exp(-t / T));

else

return p.l0 \* (1 + p.n \* exp(-(T - t) / T));

}

double force(double I, double w, double Sc, double lb) {

return (1e-5 \* 0.4 \* I \* I \* w \* w \* Sc \* Sc) / (8 \* PI \* PI \* lb);

}

void simulate(MagnetParams p, const char\* filename) {

FILE \*out = fopen(filename, "w");

if (!out) {

perror("Помилка відкриття файлу");

exit(1);

}

fprintf(out, "t\tI(t)\tl(t)\tF(t)\n");

for (double t = 0; t <= T; t += DT) {

double I\_t = current(t, p);

double l\_t = air\_gap(t, p);

double F\_t = force(I\_t, p.w, p.Sc, l\_t);

fprintf(out, "%.2f\t%.6f\t%.6f\t%.6f\n", t, I\_t, l\_t, F\_t);

}

fclose(out);

printf("Результати збережено у файл: %s\n", filename);

}

int main() {

system("chcp 65001");

MagnetParams variant2 = {

.Sc = 0.0212,

.w = 2000,

.i0 = 0.2,

.k = 0.005,

.m = 0.01,

.n = 0.01,

.l0 = 0.01

};

simulate(variant2, "output\_variant\_2.txt");

return 0;

}

1. Програмний код до умови 3:

#include <stdio.h>

#include <stdlib.h>

#include <math.h>

#define PI 3.141592653589793

#define T 100

#define DT 5

typedef struct {

double Sc;

double w;

double i0;

double k, m, n;

double l0;

} MagnetParams;

double current(double t, MagnetParams p) {

double t\_frac = t / T;

if (t\_frac >= 0 && t\_frac < 1.0/8.0)

return p.i0 \* (1 + p.k \* t);

else if (t\_frac >= 1.0/8.0 && t\_frac < 3.0/8.0)

return p.i0 \* (1 + p.k);

else if (t\_frac >= 3.0/8.0 && t\_frac < 5.0/8.0)

return p.i0 \* (1 + p.k - p.k \* (t - 3\*T/8.0));

else if (t\_frac >= 5.0/8.0 && t\_frac < 7.0/8.0)

return p.i0 \* (1 - p.k);

else

return p.i0 \* (1 - p.k + p.k \* (t - 7\*T/8.0));

}

double air\_gap(double t, MagnetParams p) {

if (t <= T / 2.0)

return p.l0 \* (1 + p.m \* exp(-t / T));

else

return p.l0 \* (1 + p.n \* exp(-(T - t) / T));

}

double force(double I, double w, double Sc, double lb) {

return (1e-5 \* 0.4 \* I \* I \* w \* w \* Sc \* Sc) / (8 \* PI \* PI \* lb);

}

void simulate(MagnetParams p, const char\* filename) {

FILE \*out = fopen(filename, "w");

if (!out) {

perror("Помилка відкриття файлу");

exit(1);

}

fprintf(out, "t\tI(t)\tl(t)\tF(t)\n");

for (double t = 0; t <= T; t += DT) {

double I\_t = current(t, p);

double l\_t = air\_gap(t, p);

double F\_t = force(I\_t, p.w, p.Sc, l\_t);

fprintf(out, "%.2f\t%.6f\t%.6f\t%.6f\n", t, I\_t, l\_t, F\_t);

}

fclose(out);

printf("Результати збережено у файл: %s\n", filename);

}

int main() {

system("chcp 65001");

MagnetParams variant3 = {

.Sc = 0.0215,

.w = 3000,

.i0 = 0.3,

.k = 0.005,

.m = 0.01,

.n = 0.01,

.l0 = 0.01

};

simulate(variant3, "output\_variant\_3.txt");

return 0;

}