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 В. М.

Студент Крячко А.К

Група КН-41/2

Варіант 5

Суми – 2025

**Зміст**

[Постановка задачі 3](#_Toc199626955)

[Теоретичний матеріал із теми 5](#_Toc199626956)

[Опис структури даних та вимоги для них 7](#_Toc199626957)

[Алгоритм роботи програми 9](#_Toc199626958)

[Опис функцій користувача 12](#_Toc199626959)

[Опис файлів та їх призначення 15](#_Toc199626960)

[Список використаних бібліотек 17](#_Toc199626961)

[Інструкція для роботи з програмою 19](#_Toc199626962)

[Приклад тестування та результати роботи програми 20](#_Toc199626963)

[Графіки 22](#_Toc199626964)

[Висновки 23](#_Toc199626965)

[Список літератури 24](#_Toc199626966)

# Постановка задачі

Варіант 5

Описати масив структур із трьох елементів. Кожна структура об’єднує дані для одного варіанта розрахунку. Необхідно для кожного варіанта на відрізку часу від 0 до Т із кроком Δt побудувати графік зміни активного струму ( ) a I A через конденсатор:

Зображення, що містить Шрифт, типографія, каліграфія, білий

Вміст, створений ШІ, може бути неправильним.

де U – напруга (В); ω – колова частота (рад/с); c – ємність конденсатора (ф);

tgδ – тангенс кута діелектричних втрат.

Зображення, що містить текст, Шрифт, почерк, знімок екрана

Вміст, створений ШІ, може бути неправильним.



Вхідні дані зчитуються з файла. Результати обчислень занести до іншого файла. Передбачити окремі функції для обчислення ω, k, U

Зображення, що містить текст, Шрифт, білий

Вміст, створений ШІ, може бути неправильним.

Зображення, що містить текст, Шрифт, білий, знімок екрана

Вміст, створений ШІ, може бути неправильним.

# Теоретичний матеріал із теми

* **Огляд мови програмування C та її застосування у розробленій програмі**
* Мова програмування C є фундаментальною у світі розробки, відома своєю давньою історією та значним впливом, що сприяв формуванню багатьох сучасних мов програмування. Вона належить до процедурного парадигми та вирізняється можливістю низькорівневого доступу до пам'яті, що забезпечує високоефективне управління системними ресурсами. Завдяки цим характеристикам C активно застосовується у створенні операційних систем, драйверів пристроїв, вбудованих систем та іншого програмного забезпечення, де критичною є висока продуктивність.
* У процесі розробки програми були використані наступні ключові програмні елементи:
* **Структури**: Це складні типи даних, які дозволяють групувати змінні різних типів під єдиним логічним іменем.
* **Функції**: Представляють собою відокремлені блоки коду, призначені для виконання конкретних завдань, і можуть бути викликані з будь-якої частини програми. У даній програмі функції k\_func, omega\_func, U\_func та calculate\_and\_save відповідають за обчислення проміжних значень та активної потужності, тоді як функція write\_results використовується для запису отриманих результатів у файл.
* **Масиви**: Це колекції елементів одного типу, що зберігаються у суміжних ділянках пам'яті. У ході виконання завдання були використані масиви time та pa для зберігання відповідних значень часу та потужності на кожному кроці розрахунку.
* **Константи**: Це змінні, значення яких залишається незмінним протягом всього виконання програми. Для спрощення обчислень були визначені константи PI
* Програма також активно використовує низку стандартних функцій:
* **Файлові операції (fopen(), fscanf(), fprintf(), fclose())**: Ці функції забезпечують взаємодію з файловою системою, дозволяючи відкривати файли, зчитувати та записувати дані, а також закривати файлові потоки. Вони були використані для читання вхідних даних з файлу input.c та збереження результатів у файл results.txt.
* **Математична функція exp()**: Призначена для обчислення експоненціальної функції.
* **Виведення на екран printf()**: Використовується для відображення інформації на консолі.
* **Системна команда System(“chcp 65001”)**: Ця функція виконує команду операційної системи, що дозволяє коректно відображати символи кирилиці.

# Опис структури даних та вимоги для них

Таблиця 1 – Опис даних

|  |  |  |  |
| --- | --- | --- | --- |
| Ім`я  параметра  у формулі | Змінна  у програмі | Тип  змінної | Призначення |
| t (час) | t | double | Поточне значення часу, яке ітерується в циклі для розрахунків. |
| T (період) | d->T | double | Загальний період часу для розрахунку. |
| ω0​ (початкова кутова частота) | d->omega0 | double | Початкове значення кутової частоти для розрахунків. |
| n (коефіцієнт зміни частоти) | d->n | double | Коефіцієнт, що впливає на зміну кутової частоти. |
| U0​ (початкова напруга) | d->U0 | double | Початкове значення напруги для розрахунків. |
| c (ємність/інший коефіцієнт) | d->c | double | Коефіцієнт, що використовується для обчислення струму Ia |
| tan(δ) (тангенс кута втрат) | d->tg\_delta | double | Тангенс кута втрат, використовується для обчислення струму Ia |
| dt (крок часу) | d->dt | double | Крок приросту часу в циклі розрахунку. |
| k(t) (коефіцієнт k у момент часу t) | k | double | Розраховане значення коефіцієнта k на основі часу t та вхідних даних. |
| ω(t) (кутова частота у момент часу t) | omega | double | Розраховане значення кутової частоти на основі часу t та вхідних даних. |
| U(t) (напруга у момент часу t) | U | double | Розраховане значення напруги на основі часу t та вхідних даних. |
| Ia​ (активний струм) | Ia | double | Розраховане значення активного струму |
| π (число Пі) | M\_PI | Double(Макрос) | Математична константа π |

# Алгоритм роботи програми



Рисунок 1 – Блок-схема main.c()

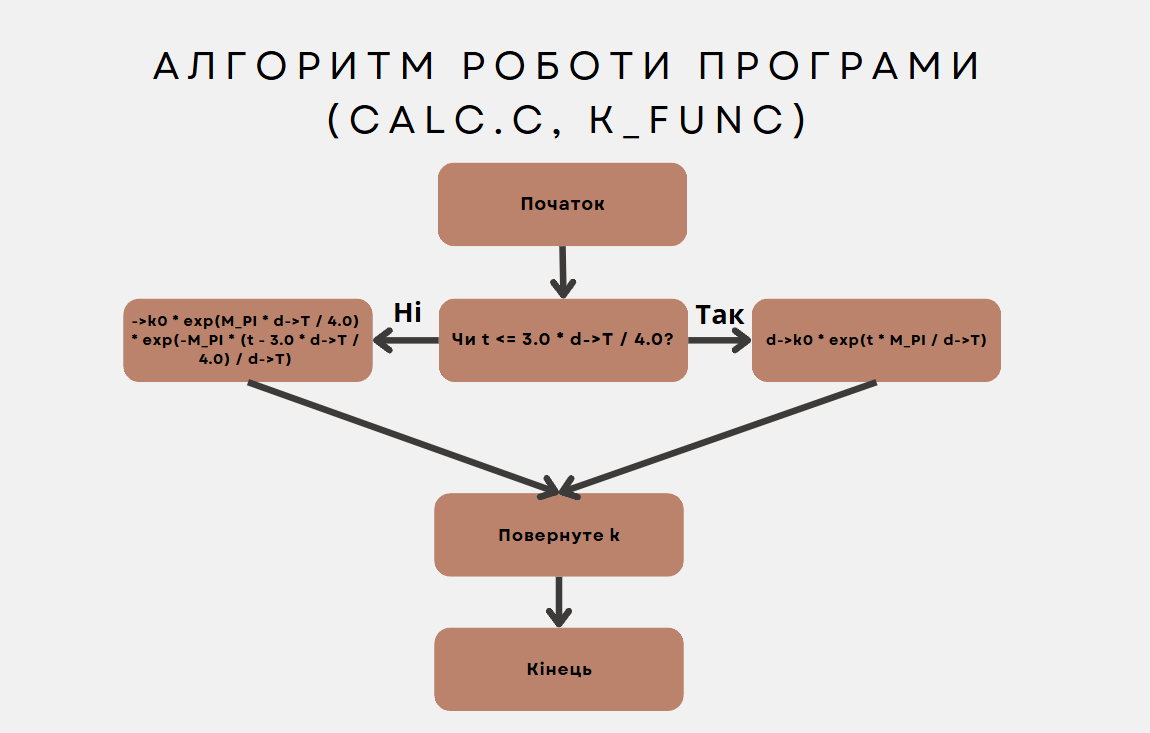


Рисунок 2 – Блок-схема k\_func



Рисунок 3 – Блок-схема omega\_func

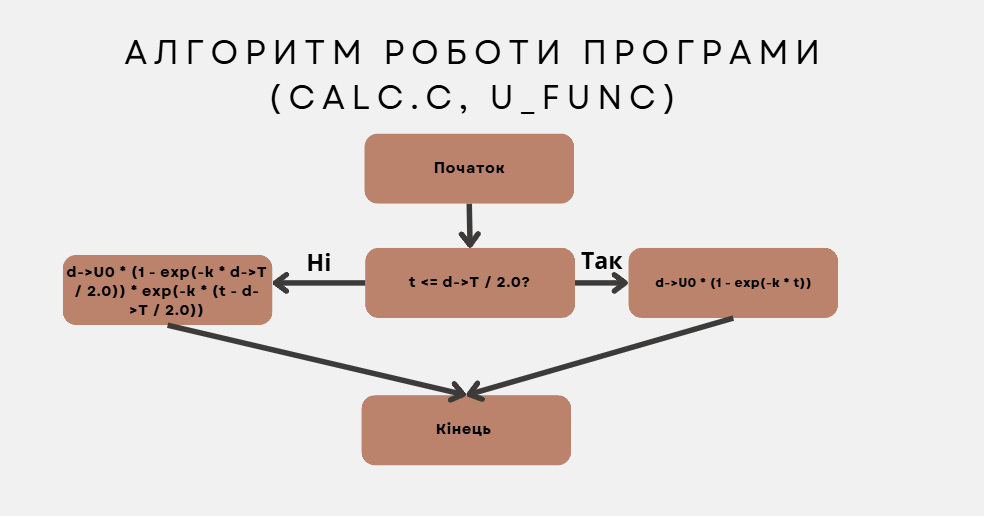


Рисунок 4 – Блок-схема u\_func



Рисунок 5 – Блок-схема calc\_and\_save

# Опис функцій користувача

У процесі розробки програми було створено 4 функції, які реалізують логіку розрахунку активного струму Ia та збереження результатів.

1. **Функція static double k\_func(double t, const InputData\* d)**
   * **Призначення:** Функція призначена для обчислення коефіцієнта k залежно від часу t. Цей коефіцієнт використовується у формулі для розрахунку напруги U.
   * **Вхідні параметри:**
     + t (double): Поточний момент часу (с), для якого виконується розрахунок.
     + d (const InputData\*): Вказівник на константну структуру, яка містить загальний час дослідження T та початкове значення коефіцієнта k0.
   * **Вихід:**
     + Функція повертає значення k типу double, яке використовується для подальших обчислень у функції U\_func().
2. **Функція static double omega\_func(double t, const InputData\* d)**
   * **Призначення:** Ця функція обчислює значення кутової частоти omega у певний момент часу t. Залежність omega від часу є кусково-заданою.
   * **Вхідні параметри:**
     + t (double): Поточний момент часу (с), для якого обчислюється кутова частота omega.
     + d (const InputData\*): Вказівник на константну структуру InputData, що містить загальний час T, початкову кутову частоту omega0 та коефіцієнт n.
   * **Вихід:**
     + Функція повертає значення кутової частоти omega типу double, яке передається до функції calculate\_and\_save() для розрахунку активного струму Ia.
3. **Функція static double U\_func(double t, const InputData\* d)**
   * **Призначення:** Обчислює миттєву напругу U на основі часу t та змінного коефіцієнта k. Логіка функції є кусковою та залежить від попередньо обчисленого k.
   * **Вхідні параметри:**
     + t (double): Поточний момент часу (с).
     + d (const InputData\*): Вказівник на константну структуру InputData, що містить загальний час T та початкове значення напруги U0.
   * **Логіка роботи:**
     + Функція спочатку викликає k\_func(t, d) для отримання значення k.
     + На першому інтервалі (коли t <= d->T / 2.0), напруга розраховується за формулою d->U0 \* (1 - exp(-k \* t)).
     + В іншому випадку (коли t > d->T / 2.0), напруга обчислюється як d->U0 \* (1 - exp(-k \* d->T / 2.0)) \* exp(-k \* (t - d->T / 2.0)).
   * **Вихід:**
     + Функція повертає значення напруги U типу double у відповідний момент часу, яке використовується для розрахунку активного струму Ia у функції calculate\_and\_save().
4. **Функція void calculate\_and\_save(const InputData\* d)**
   * **Призначення:** Ця функція є головною для виконання всіх розрахунків та збереження результатів у файл "output.txt". Вона не повертає жодного значення (void).
   * **Вхідні параметри:**
     + d (const InputData\*): Вказівник на константну структуру InputData, що містить усі необхідні вхідні дані для розрахунків, включаючи d->T, d->dt, d->c та d->tg\_delta.
   * **Логіка роботи:**
     + Функція намагається відкрити файл "output.txt" у режимі запису.
     + Якщо файл не може бути відкрито, виводиться повідомлення про помилку на консоль, і функція завершується.
     + У разі успішного відкриття, функція виводить заголовки результатів на консоль та у файл.
     + Далі, вона ітерує змінну часу t від 0 до d->T з кроком d->dt.
     + У кожній ітерації циклу обчислюються U за допомогою U\_func(), omega за допомогою omega\_func(), а потім активний струм Ia за формулою U \* omega \* d->c \* d->tg\_delta.
     + Обчислені значення t та Ia виводяться на консоль та записуються у файл "output.txt".
     + Після завершення всіх розрахунків та запису, файл "output.txt" закривається.

# Опис файлів та їх призначення

Для реалізації програми було задіяно 3 файли.

1. main.c

Файл main.c: Цей файл містить основну логіку програми, функцію main(), з якої починається виконання. Він відповідає за:

* Встановлення кодування для коректного відображення кирилиці (system("chcp 65001");).
* Запит у користувача номера варіанту розрахунку (від 1 до 3).
* Завантаження вхідних даних відповідно до обраного варіанту за допомогою функції load\_input().
* Обробку помилок, якщо введено невірний номер варіанту.
* Виклик функції calculate\_and\_save() для виконання розрахунків та збереження результатів.
* Повідомлення користувача про завершення розрахунку.

1. input.h

Це заголовний файл, який містить оголошення структури InputData

Структура InputData об'єднує всі необхідні вхідні параметри для розрахунків, такі як T, dt, U0, k0, m, omega0, n, c, tg\_delta.

Також у цьому файлі оголошено прототип функції load\_input(), яка відповідає за завантаження даних для різних варіантів.

1. Input.c

Цей файл містить реалізацію функції load\_input()

Вона приймає вказівник на структуру InputData та номер варіанту.

Використовуючи оператор switch, функція присвоює відповідні значення параметрам структури InputData залежно від вибраного варіанту (1, 2 або 3).

Повертає 1 у разі успішного завантаження даних і 0, якщо номер варіанту недійсний

1. Calc.h

Цей заголовний файл містить оголошення функції calculate\_and\_save().

Він включає input.h, що дозволяє використовувати структуру InputData.

Основна мета цього файлу – надати прототип функції, щоб інші файли могли її викликати.

1. calc.c

Цей файл містить реалізації функцій, що виконують математичні розрахунки, та функцію для збереження результатів.

* **k\_func()**: Обчислює коефіцієнт k залежно від часу t та вхідних даних.
* **omega\_func()**: Обчислює кутову частоту omega залежно від часу t та вхідних даних.
* **U\_func()**: Обчислює напругу U залежно від часу t та обчисленого коефіцієнта k.
* **calculate\_and\_save()**: Ця функція відкриває файл "output.txt", виконує цикл за часом, обчислює U, omega та активний струм Ia на кожному кроці, виводить їх на консоль та записує у файл. Вона також обробляє можливі помилки відкриття файлу.

1. output.txt

Це текстовий файл, який генерується програмою під час виконання.

Він містить результати розрахунків для кожного обраного варіанту.

Дані представлені у вигляді таблиці, де для кожного моменту часу t вказується відповідне значення активного струму

1. Makefile

Це текстовий файл, який автоматизує процес компіляції програми з вихідних файлів. Він містить набір правил та інструкцій для утиліти make, яка визначає, які файли потрібно перекомпілювати, щоб отримати кінцевий виконуваний файл

# Список використаних бібліотек

1. <stdio.h>

Бібліотека <stdio.h> є однією з найважливіших і найчастіше використовуваних у мові програмування C. Вона відповідає за введення та виведення інформації, як на екран, так і в файли.

Функції, що використовуються в програмі:

* fopen() / fclose() - відповідають за відкриття та закриття файлів.
* fscanf() - використовується для зчитування вхідних параметрів з текстового файлу. Особливістю є те, що вона дозволяє форматувати вхідні дані, що критично важливо для коректної ініціалізації параметрів розрахунків.
* fprintf() - забезпечує запис результатів у вихідний файл у чітко структурованому вигляді.
* printf() - виводить проміжні результати розрахунків у консоль, що значно полегшує процес налагодження програми.

1. <math.h>

Бібліотека <math.h> надає потужний набір математичних функцій, які є невід'ємною частиною наукових та інженерних розрахунків. У нашому випадку вона використовується для реалізації складних обчислень, що описують фізичні процеси.

У програмі застосована лише одна функція із цієї бібліотеки:

* exp() - обчислює експоненційну функцію, яка відіграє головну роль у формулах для розрахунку напруги та частоти. Ця функція дозволяє точно моделювати процеси затухання та зростання.

1. <stdlib.h>

Бібліотека <stdlib.h> містить набір утиліт для роботи з операційною системою та виконання базових операцій [3].

Застосування в програмі:

* system() - дозволяє виконувати системні команди. У нашому випадку використовується для зміни кодування консолі на UTF-8 (chcp 65001), що забезпечує коректне відображення кириличних символів.

Також у програмі використовуються макроси для визначення ключових констант:

#define M\_PI 3.14159265358979323846

# Інструкція для роботи з програмою

1. Підготовка до запуску Переконайтесь, що всі вихідні файли програми(main.c, input.c, calc.c, input.h, calc.h) знаходяться у одній папці.

2. Запуск програми Запустіть програму kurs.exe (Windows), або в консолі введіть ./kurs (Linux/MacOS)

3. Введення даних Введіть число 1,2 або 3 відповідно до бажаного варіанту і натисніть Enter. Якщо введено некоректне значення, програма завершить роботу з помилкою.

4. Отримання результатів Після введення коректного варіанту, програма обчислить струм за часом для заданих параметрів. Результати одночасно виведуться у консолі та збережуться у файл output.txt

# Приклад тестування та результати роботи програми

Вхідні дані одразу для трьох варіантів

Зображення, що містить текст, Шрифт, знімок екрана, алгебра

Вміст, створений ШІ, може бути неправильним.

Нижче наведені відповідні результати роботи програми (Рисунок 6, Рисунок 7, Рисунок 8)

Зображення, що містить текст, знімок екрана, меню, Шрифт

Вміст, створений ШІ, може бути неправильним.

Рисунок 6 – Результат роботи програми (1 частина)

Зображення, що містить текст, меню, знімок екрана, Шрифт

Вміст, створений ШІ, може бути неправильним.

Рисунок 7 – Результат роботи програми (2 частина)

Зображення, що містить текст, знімок екрана, меню, типографія

Вміст, створений ШІ, може бути неправильним.

Рисунок 8 – Результат роботи програми (3 частина)

# Графіки

**Графіки**

Для візуалізації результатів роботи програми було побудовано графіки залежності

активного струму Ia​ від часу t для кожного з трьох варіантів розрахунків. Враховуючи експоненційний характер зміни параметрів та малі значення струму, для побудови графіків було обрано напівлогарифмічний масштаб, що дозволило наочно відобразити динаміку процесу.

Зображення, що містить текст, ряд, Графік, схема

Вміст, створений ШІ, може бути неправильним.

Рисунок 9 – Графік результатів програми(1-3 варіант)

# Висновки

**Висновки**

Програма успішно реалізує обчислення активного струму Ia​ залежно від часу, ґрунтуючись на змінах напруги U, коефіцієнта k та кутової частоти ω, які також змінюються з часом за кусково-заданими функціями.

**Основні досягнення та результати:**

* **Модульна архітектура:** Програма демонструє ефективне використання модульного підходу, розділяючи логіку на окремі функції (k\_func, omega\_func, U\_func, calculate\_and\_save). Це значно покращує читабельність, підтримуваність та можливість повторного використання коду.
* **Гнучкість вхідних даних:** Завдяки використанню структури InputData та функції load\_input, програма може легко адаптуватися до різних наборів вхідних параметрів, що дозволяє проводити розрахунки для кількох варіантів без зміни основного коду.
* **Ефективна робота з файлами:** Реалізовано зчитування вхідних даних з файлу (через логіку завантаження варіантів) та збереження результатів у файл "output.txt", що забезпечує постійне зберігання даних та зручність аналізу.
* **Математична точність:** Використання типу double для всіх розрахунків та математичної бібліотеки (<math.h>) забезпечує необхідну точність обчислень для експоненційних залежностей та малих значень параметрів.
* **Візуалізація даних:** Наявність згенерованих результатів в output.txt створює основу для подальшої візуалізації (наприклад, побудови графіків залежності Ia​ від t), що дозволяє наочно аналізувати динаміку процесу для кожного варіанту. Враховуючи характер змін, напівлогарифмічний масштаб графіків є доцільним для відображення динаміки малих значень.

Загалом, розроблена програма є надійним інструментом для моделювання та аналізу заданих фізичних процесів, що демонструє ключові аспекти процедурного програмування на C та ефективної обробки даних.

# Список літератури

1. Авраменко В. В. Програмування [Електронний ресурс] : навчальний курс / Віктор Васильович Авраменко. – Режим доступу: <https://mix.sumdu.edu.ua/textbooks/104505/index.html>. – Назва з екрана.
2. Громов Ю.Ю., Татарніков О.М. Електричні та магнітні властивості діелектриків: навч. посібник. – К. : Наукова думка, 2018. – 232 с.
3. Керніган Б., Рітчі Д. Керівництво користувача мови С. – К. : Вільямс. – 2002. – 272 с.
4. Методичні вказівки до курсової роботи з дисципліни «Програмування» / уклад.: Авраменко В. В., Боровик В. О., Тиркусова Н. В. – Суми : Сумський державний університет, 2021. – 43 с.