**Д****НІПРОВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ**

**ІМЕНІ ОЛЕСЯ ГОНЧАРА**

**ФАКУЛЬТЕТ ПРИКЛАДНОЇ МАТЕМАТИКИ КАФЕДРА КОМП’ЮТЕРНИХ ТЕХНОЛОГІЙ**

**ЗВІТ**

**З НАВЧАЛЬНОЇ ПРАКТИКИ**

**КОМП’ЮТЕРНО - ТЕХНОЛОГІЧНОЇ**

Освітньо-професійна програма

Комп’ютерне моделювання та технології програмування

Спеціальність 113 Прикладна математика

Галузь знань 11 *Математика* і статистика

Рівень вищої освіти перший (бакалаврський)

Студента 2 курсу групи ПА-22-2

\_Овдієнко А.В,\_\_\_

(прізвище та ініціали)

Керівник \_Сірик С.Ф.\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_

(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Кількість балів \_\_\_\_\_\_\_\_\_\_\_

Національна шкала \_\_\_\_\_\_\_

Члени комісії:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Зайцев В.Г.

(підпис) (прізвище та ініціали)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Золотько К.Є.

(підпис) (прізвище та ініціали)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Сірик С.Ф.

(підпис) (прізвище та ініціали)

м. Дніпро, 2024 р.

**Зміст**

# Вступ

Проходження навчальної комп’ютерно - технологічної практики є невід’ємною складовою частиною під час навчального процесу – оскільки вона являється ефективним засобом для закріплення отриманих під час навчання знань, умінь і навичок, а також практика дозволяє студенту як майбутньому спеціалісту освоїти додаткові знання та отримати базовий досвід роботи з різними середовищами програмування.

При проходженні комп’ютерно – технологічної практики була поставлена задача - побудови графіку функції за індивідуальним завданням.

Для реалізації даного завдання потрібно:

* реалізувати побудову графіків функцій;
* надати оформлення: вісі (розмітка вісей, підписи вісей);
* вміти реалізувати побудову декількох графіків одночасно в одній системі координат: функції індивідуального завдання; функції sin(x); функції cos(x); функції, заданої рандомними числами; функції, заданої в текстовому файлі дискретно;
* зробити паралельний перенос графіку відносно вісей абсцис / ординат;
* зробити стиснення / розтягнення графіку функції.

Для виконання індивідуального завдання необхідно вдосконалити знання з мови програмування C++ у середовищі Visual Studio Code. Закріпити навички з курсу програмування, математичного аналізу, отримані під час навчання. Ознайомитися з базовою літературою, яку надано в начальній практиці.

# Постановка задачі

**Примітка:** ункціональні можливості повинні бути реалізовані в тексті програми (засобами мови) і в інтерфейсі користувача (засобами візуальних компонент)

За допомогою графічних методів візуального середовища створити графіки функцій відповідно до індивідуального завдання №15

1. Вивчити базові можливості візуального програмування;
2. Вивчити базові можливості програмування графіки;
3. Зобразити блок-схему алгоритму для виведення на екран графіка функції;
4. Написати програму для виведення на екран графіка функції, а також програму для підготовки даних (номер варіанта Індивідуального завдання збігається з номером в студентському журналі);
5. Підготувати звіт про виконання завдання для навчальної практики: комп'ютерно – технологічної.

Технічні вимоги:

1. Вигляд (побудова) графіка;
2. вісі (розмітка вісей, підписи вісей, оформлення);
3. можливість виведення декількох графіків;
4. вивести графік функції, заданої в текстовому файлі дискретно, тобто значеннями функції в точках масштабування графіку;
5. переміщення графіка (паралельний перенос);
6. стиснення / розтягнення графіка функції щодо вісі абсцис / ординат.

# Реалізація графічного інтерфейсу програми

Інтерфейс(Рисунок 1.1) складається з прямокутників, які поступово перекривають один одного, відповідних текстових конструкторів, полів для вводу, ліній (віссі абцис та ординат),а також заднього фону – всі ці елементи розташовані у вікні sfml(Рисунок 1.1). Розмітка виконана так, щоб користувачеві було зручно працювати з програмою: одна частина призначена для відображення графіків (Рисунок 1.2), інша для головного меню (Рисунок 1.3), а також шапка(Рисунок 1.4) - для штучної панелі інструментів.

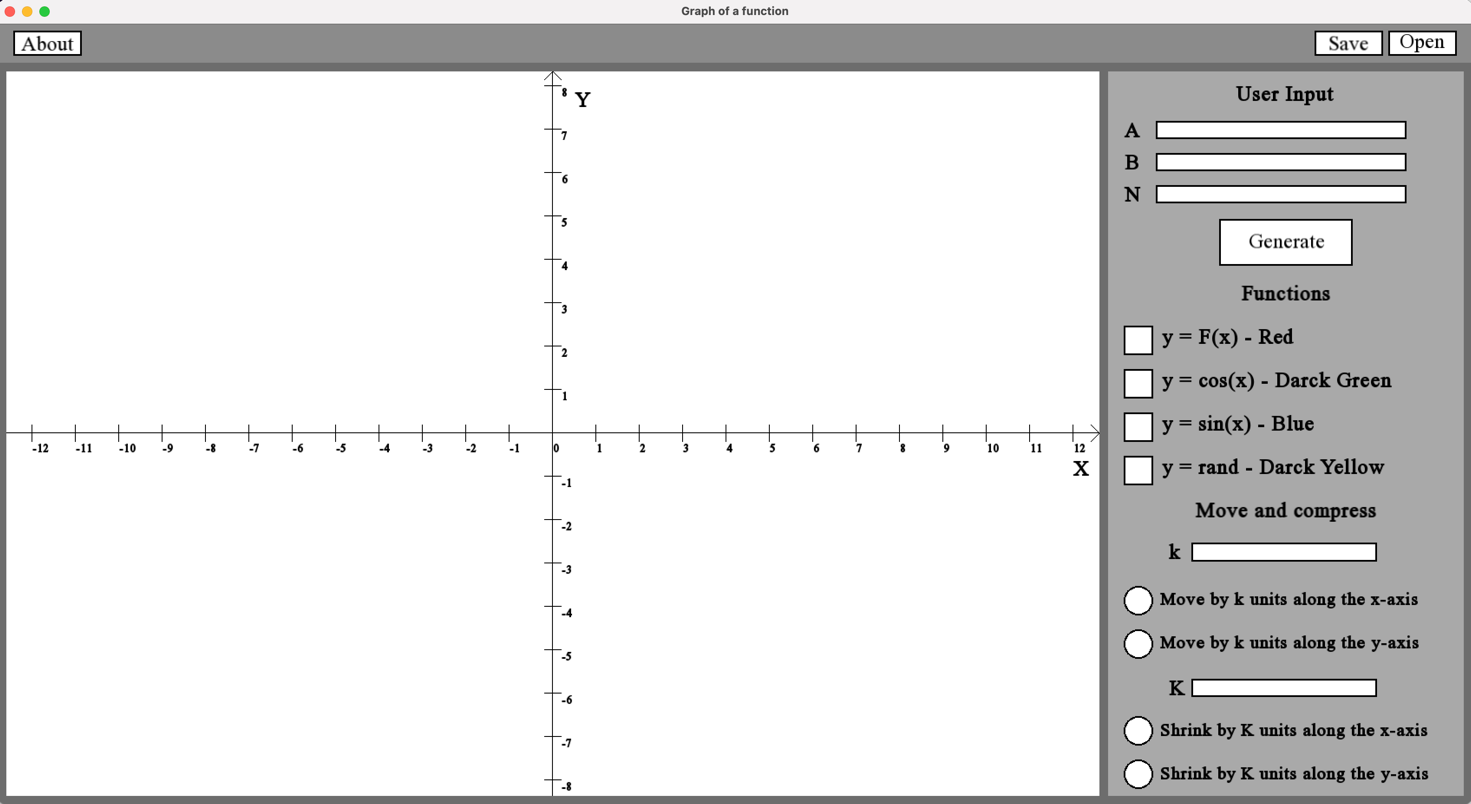


Рисунок 1.1 – Головне вікно проорами.

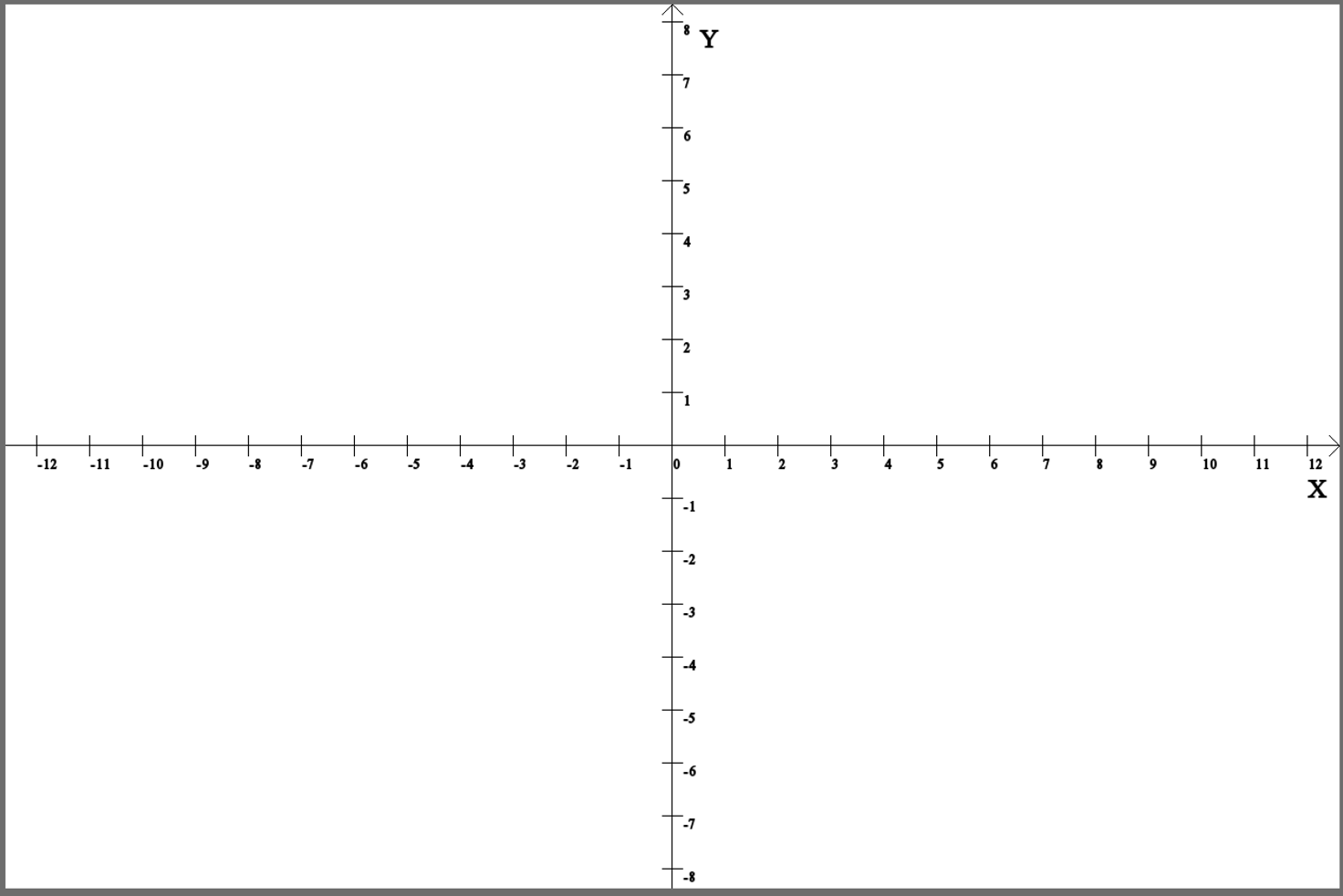


Рисунок 1.2 - Область для відображення графіків.

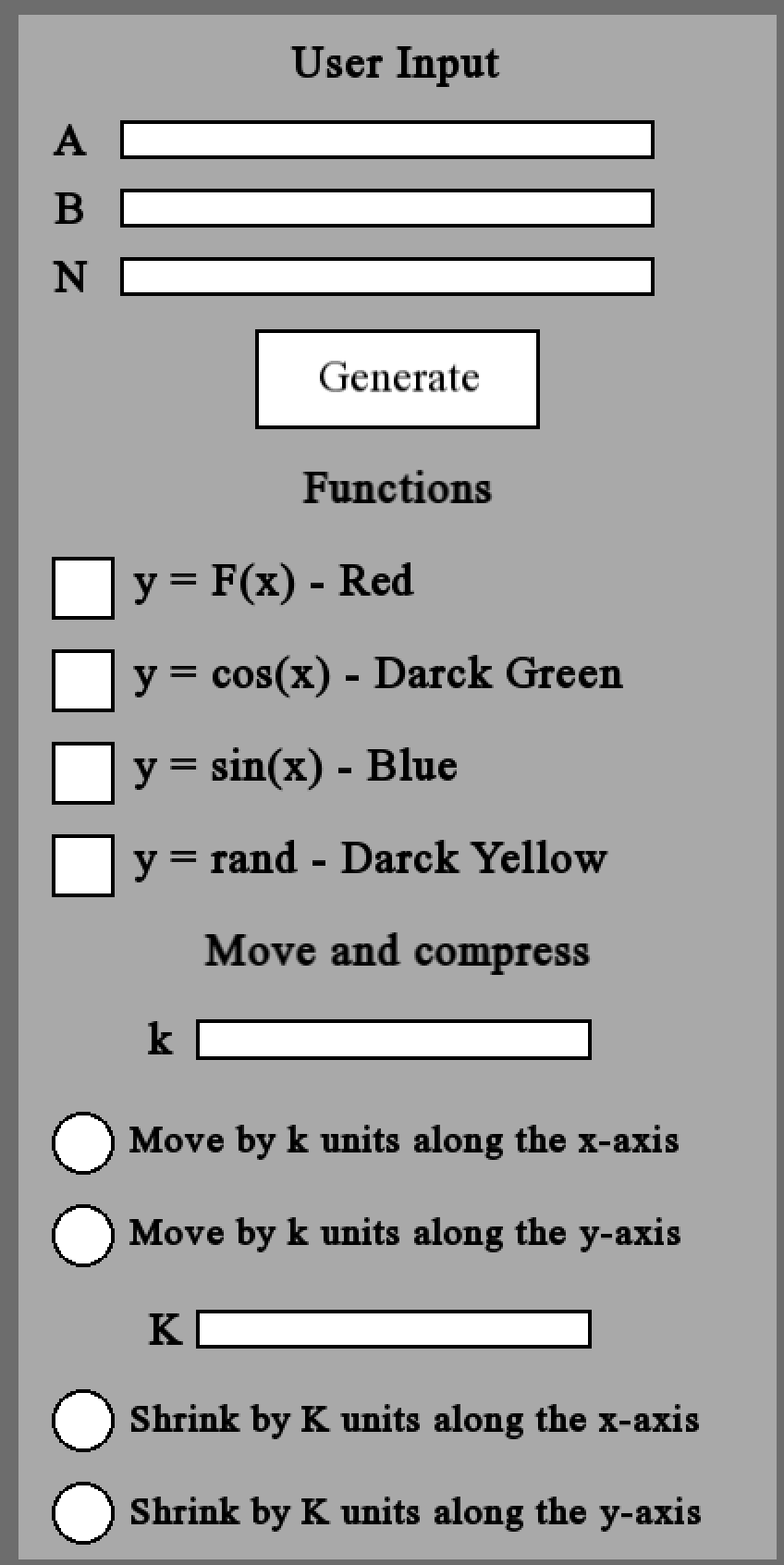


Рисунок 1.3 – Область головного меню для користувача.



Рисунок 1.4 – Область штучної панелі інструментів.

Звичайно в процесі роботи можна буде змінити позицію тексту, прямокутників, полів вводу, ліній за допомогою зміни позиції крайньої лівої верхньої точки обʼєкта.

## Додавання границь функції

Для додавання границь функції було використано два обʼєкта A(Рисунок 1.5) і B(Рисунок 1.5) , які мали: текстовий тип, який відповідає за підказки, що саме ми вводимо; прямокутник, щоб виділити область де будемо вводити текст; стрінг, для вводу текста, щоб перевіряти на коректність вводу символи, і звичайно щоб можна було вводити лише числа в правильній формі, тобто мінус лише першим символом, крапку лише один раз; змінну дробного типа, у якій буде зберігатися наше число для подальшого використання; змінну булівського типа, яка відповідає за натискання на проміжуток поля вводу; і звичайно текстове поле, яке буде відображати наш ввід. Обидва обʼєкта (Рисунок 1.5) в підказка так і підписані: A, B – для зручності.

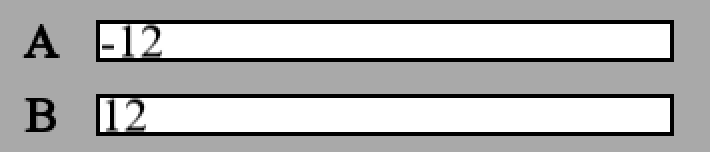


Рисунок 1.5 – Границі функцій.

Якщо користувач введе початкове значення інтервала, яке більше за кінцеве, то програма автоматично поміняє місцями ці два значення та всі їхні допоміжні обʼєкти, а також виведе повідомлення(Рисунок 1.6) на єкран у вигляді ще одного вікна(Рисунок 1.6), яка має домівнювання над основним.

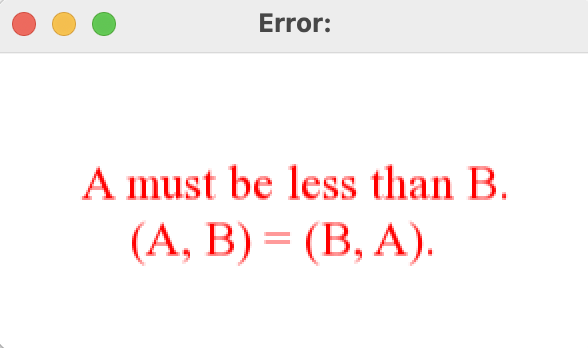


Рисунок 1.6 – Повідомлення про некоректні границі функції.

Для повідомлень типу(Рисунок 1.6) для закриття слід нажати на кнопку закриття зліва вгорі, яка є в кожної програми.

## Додавання кількості точок графіку

Графік будується по точкам на проміжку, який задав користувач. Так само, як і проміжки, він має ввести й кількість точок із яких буде складатися графік. Тобто є ще один обʼєкт N(Рисунок 1.7), який потрібен лише для кількісті точок. Також має: текстовий тип, який відповідає за підказки, що саме ми вводимо; прямокутник, щоб виділити область де будемо вводити текст; стрінг для вводу текста, щоб перевіряти на коректність вводу символи, і звичайно щоб можна було вводити лише натуральні числа не більше ста тисяч; змінну цілочисельного типа, у якій буде зберігатися наше число для подальшого використання; змінну булівського типа, яка відповідає за натискання на проміжуток поля вводу; і звичайно текстове поле, яке буде відображати наш ввід.



Рисунок 1.7 – Кількість точок функції на певному проміжку.

І якщо ми не введемо проміжки A(Рисунок 1.5), B(Рисунок 1.5) чи кількість точок N(Рисунок 1.7), або введемо занадто велике N(Рисунок 1.7), то отримаємо повідомлення(Рисунок 1.8) з вказівками про помилку вводу.

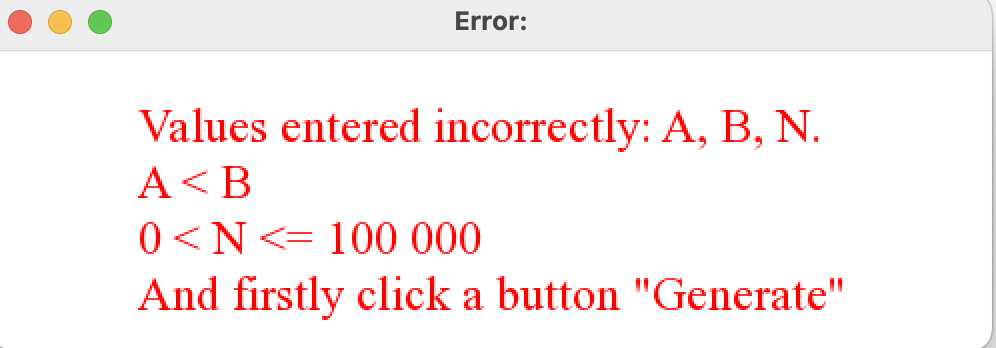


Рисунок 1.8 – Повідомлення про помилку вводу границь чи кількості точок.

## Кнопка генерації та панель вибору графіка

Після введення проміжків(Рисунок 1.5) функції та кількості точок(Рисунок 1.7) графіків слід натиснути на кнопку генерації(Рисунок 1.9), щоб графіки сворилися.



Рисунок 1.9 – Кнопка генерації графіків.

Тепер вже можна обирати графіки(Рисунок 1.10) функцій для відображення.

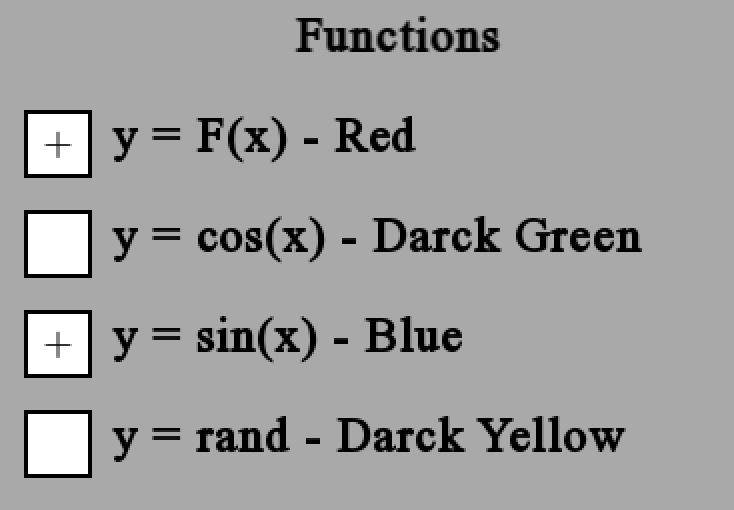


Рисунок 1.10 – Функції, які можна вибрати для відображення графіків.

Зазначимо, що були створені додаткові функції sin(x) та cos(x) для демонстрації правильної побудови. Поруч із функцією написан колір – це колір графіку. Також можна помітити рандомну функцію, якщо ми захочемо її відобразити, то відкривається вікно(Рисунок 1.11) для вибору границь для рандомної функції.

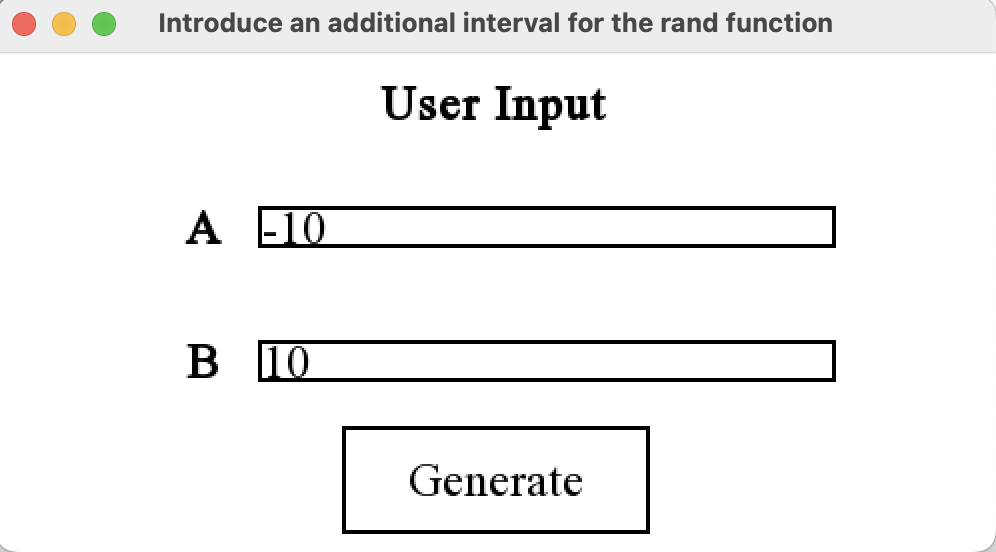


Рисунок 1.11 – Вікно для вибору границь рандомної функції.

Якщо користувач введе початкове значення інтервала, яке більше за кінцеве, то програма автоматично поміняє місцями ці два значення та всі їхні допоміжні обʼєкти, а також виведе повідомлення(Рисунок 1.6) на єкран у вигляді ще одного вікна(Рисунок 1.6), яка має домівнювання над основним. І слід натиснути на кнопку генерації(Рисунок 1.12) у цьому вікні(Рисунок 1.11).

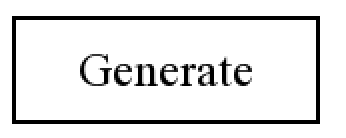


Рисунок 1.12 – Кнопка генерації рандомної функції.

І побудується графік з рандомними функціями у кожній точці, а кількість точок(Рисунок 1.7) візьметься з головного меню(Рисунок 1.3).

І щойно згенерується рандомний графік – на області вибору функцій для відображення зʼявиться проміжок(Рисунок 1.14) для рандомної функції на виборі функцій(Рисунок 1.13).

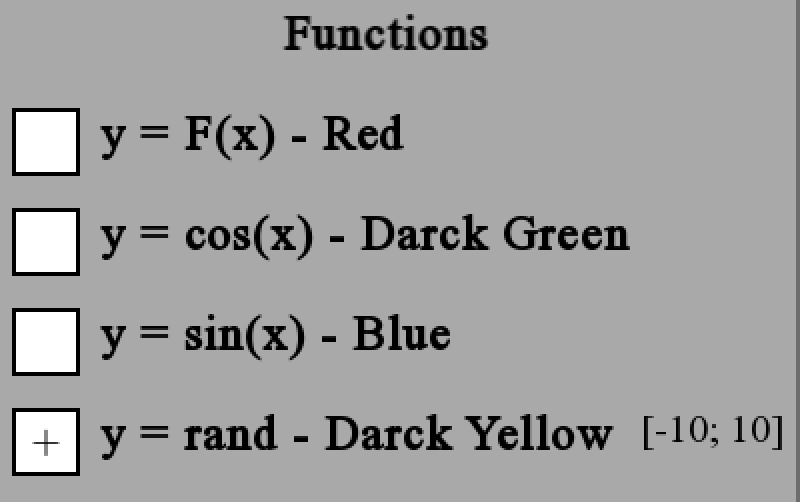


Рисунок 1.13 – Виділений графік функції із заданими проміжками.

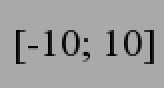


Рисунок 1.14 – Заданий проміжок рандомної функції.

І якщо користувач вирішив змінити кількість точок(Рисунок 1.7), але не натиснув на кнопку генерації(Рисунок 1.9) – то вивідеться повідомлення про помилку(Рисунок 1.8), що слід також натиснути на кнопку генерації(Рисунок 1.9).

## Переміщення та стиснення графіків

Для переміщення та стиснення графіків використовувалися поля для введення руху та стиснення графіка.

Поле для вводу переміщення графіків(Рисунок 1.15): текстовий тип, який відповідає за підказки, що саме ми вводимо; прямокутник, щоб виділити область де будемо вводити текст; стрінг, для вводу текста, щоб перевіряти на коректність вводу символи, і звичайно щоб можна було вводити лише числа в правильній формі, тобто мінус лише першим символом, крапку лише один раз; змінну дробного типа, у якій буде зберігатися наше число для подальшого використання; змінну булівського типа, яка відповідає за натискання на проміжуток поля вводу; і звичайно текстове поле, яке буде відображати наш ввід.

Поле для вводу стискання графіків(Рисунок 1.16): текстовий тип, який відповідає за підказки, що саме ми вводимо; прямокутник, щоб виділити область де будемо вводити текст; стрінг, для вводу текста, щоб перевіряти на коректність вводу символи, і звичайно щоб можна було вводити лише числа в правильній формі, тобто лише позитивні числа, крапку лише один раз; змінну дробного типа, у якій буде зберігатися наше число для подальшого використання; змінну булівського типа, яка відповідає за натискання на проміжуток поля вводу; і звичайно текстове поле, яке буде відображати наш ввід.



Рисунок 1.15 – Поле вводу для переміщення графіків.



Рисунок 1.16 – Поле вводу для стискання графіків.

Також є кнопки для застосування переміщення по осям(Рисунок 1.17), натиснувши на яких береться число із поля для переміщення(Рисунок 1.15) і переміщуються вибрані графіки(Рисунок 1.10).

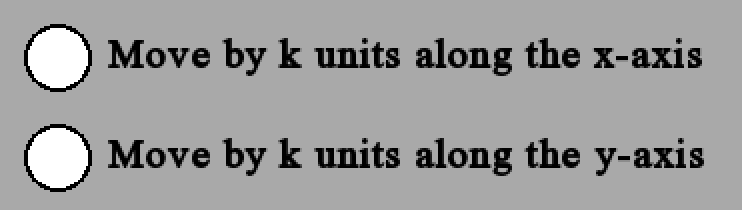


Рисунок 1.17 – Кнопки з підказками для переміщення функцій по осям.

Також є кнопки для застосування стиснення графіків по осям(Рисунок 1.18), натиснувши на яких береться число із поля для стиснення(Рисунок 1.16) і свужуються вибрані графіки(Рисунок 1.10).

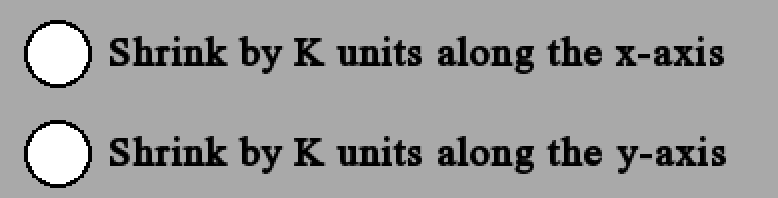


Рисунок 1.18 – Кнопки з підказками для стиснення функцій по осям.

І якщо ми нічого не введемо і cпробуємо перемістити графіки – програма виведе повідомлення(Рисунок 1.19), де говорить про дії, які немають сенсу: додавання нуля, множення на нуль.

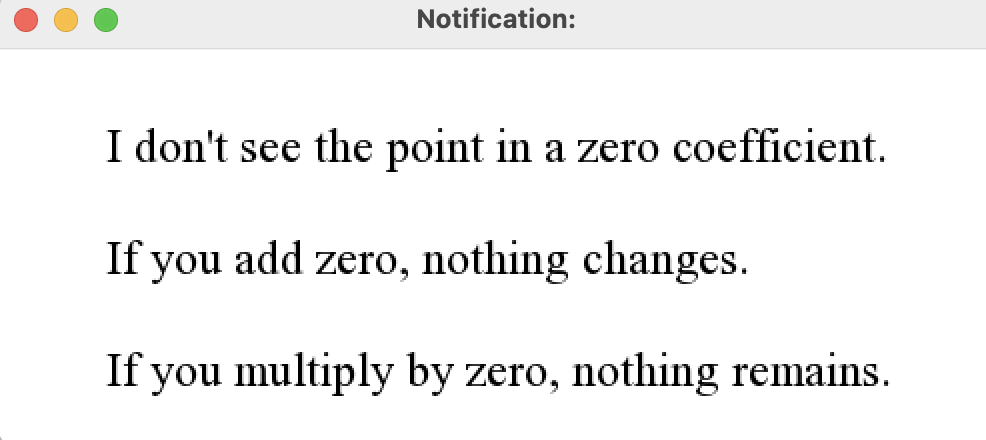


Рисунок 1.19 – Повідомлення про некоректне переміщення чи стискання.

## Шапка – штучна панель інструментів

Штучна панель інструментів(Рисунок 1.4) представляє собою три однакові кнопки, але різні за назвами: “About”, “Save”, “Open”.

Якщо людина натисне на кнопку «About», то відобразиться вікно (Рисунок 1.20), де описана коротка інформація про розробника та програму.

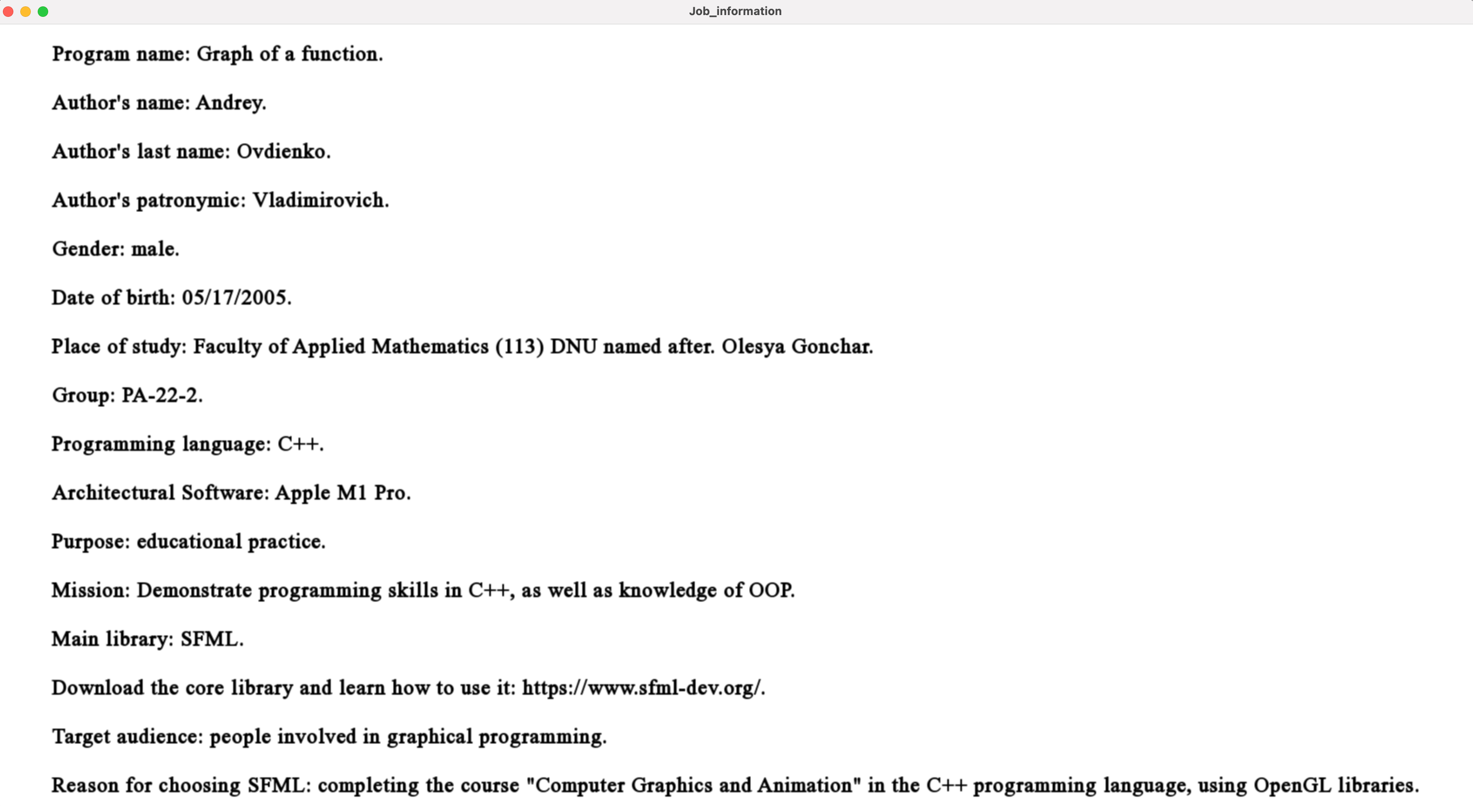


Рисунок 1.20 – Інформація про розробника та програму.

Якщо людина натисне на кнопку «Save», то відобразиться вікно(Рисунок 1.21), де можна написати імʼя файлу, чи повний шлях, зберігти у тестовому файлі в форматі “.txt” - натиснувши на кнопку “Save” у цьому вікні(Рисунок 1.21).

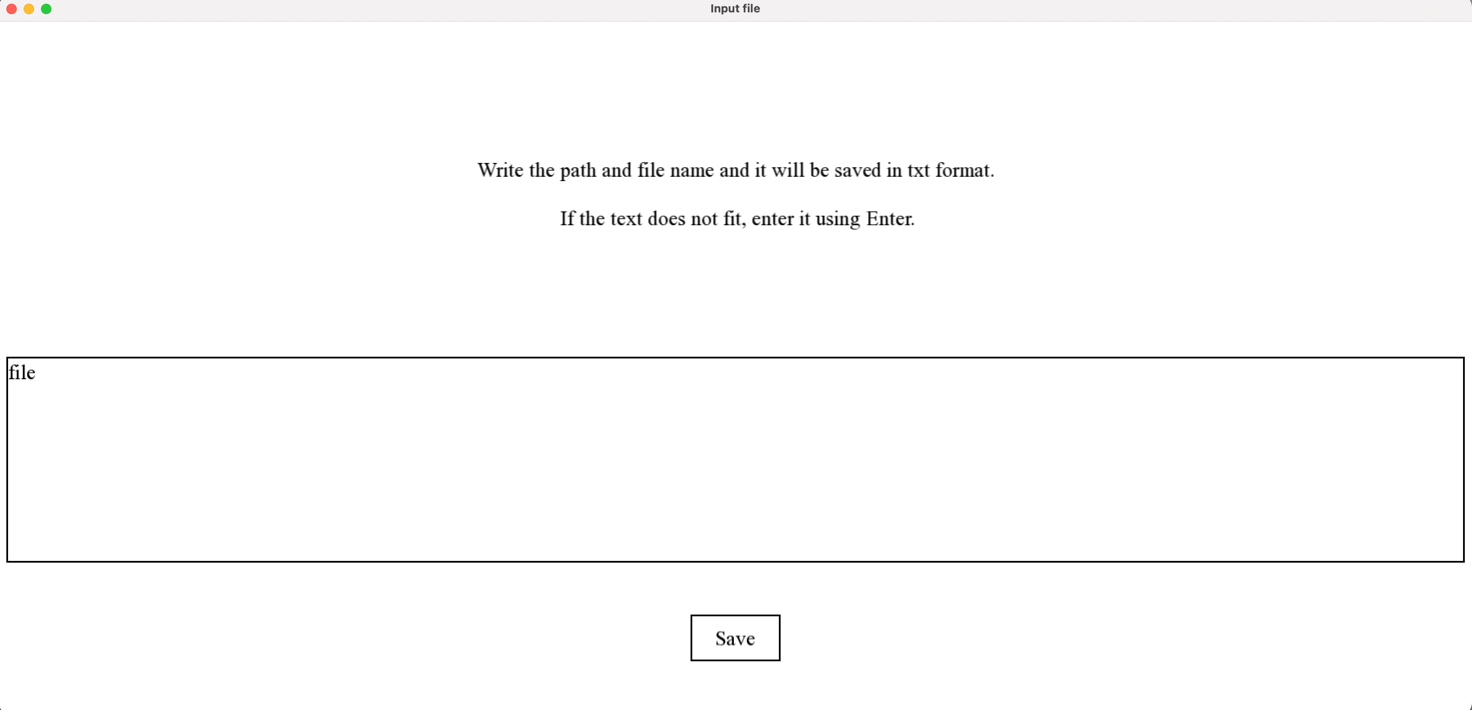


Рисунок 1.21 – Вікно для вводу назви файлу, у який все збережеться.

Після збереження ввивідиться вікно(Рисунок 1.22), що файл збережено та його повна назва. Щоб закрити вікно – натиснути на кнопку “OK” у цьому вікні(Рисунок 1.22). Тобто ми зберігли всі графіки, в один файл.



Рисунок 1.22 – Вікно про збереження данних у відповідний файл.

Майже те саме для “Open”. У нас є відповідне вікно(Рисунок 1.23), куди можна ввести назву файлу, який буде відкритий

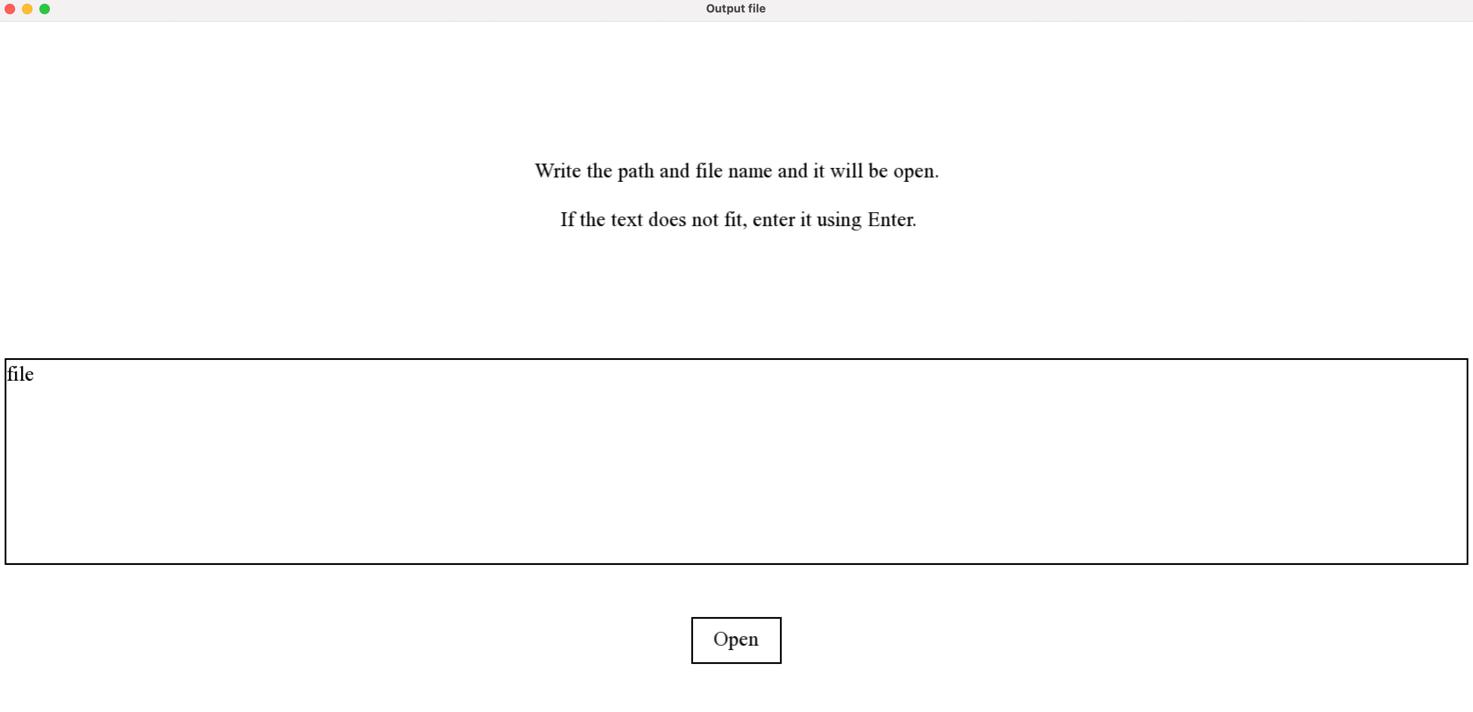


Рисунок 1.23 – Вікно для вводу файлу, який ми хочемо відкрити.

Якщо такий файл є і він відкрився – буде відповідне повідомлення(Рисунок 1.24).



Рисунок 1.24 – Вікно про відкриття файлу, де знаходяться данні для програми.

Якщо такого файлу немає то буде альтернативне повідомлення(Рисунок 1.25).

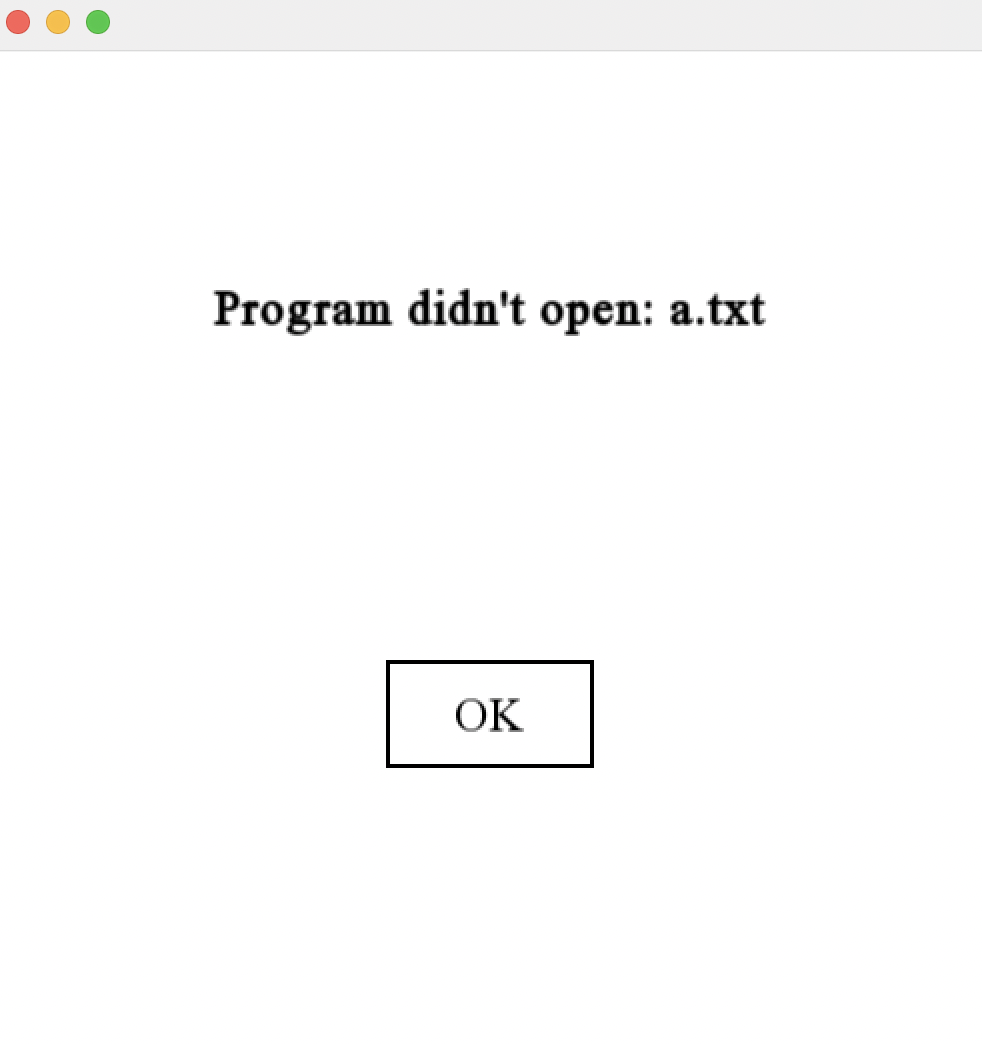


Рисунок 1.25 – Помилка відкриття файлу.

# Опис програмного забезпечення

## Мова програмування C++

Для створення графічного інтерфейсу використаємо бібліотеку sfml, яку можна завантажити із офіційного сайту [SFML](https://www.sfml-dev.org/) (Рисунок 2.1).



Рисунок 2.1 - Головна сторінка офіційного сайту SFML.

Зверху можна побачити розділи де можна навчитися працювати з цією бібліотекою, завантажити її, громада та розробники.

Ми підготували відповідну бібліотеку. Тепер слід налаштувати Visual Studio Code для роботи С++ і звісно встановити його.

Зручне середовище для написання коду, а також маже єдине можливе для C++ - Visual Studio Code. Щоб завантажити цю середу розробки мені допоміг сайт Microsoft(Рисунок 2.2) [Visual Studio](https://visualstudio.microsoft.com/downloads/?utm_medium=post-banner&utm_source=microsoft.com&utm_campaign=channel+banner&utm_content=launch+vs2022&icid=mscom_marcom_dlc_pfdev). Де я міг обрати своою операційну систему та бажану мову програмування.

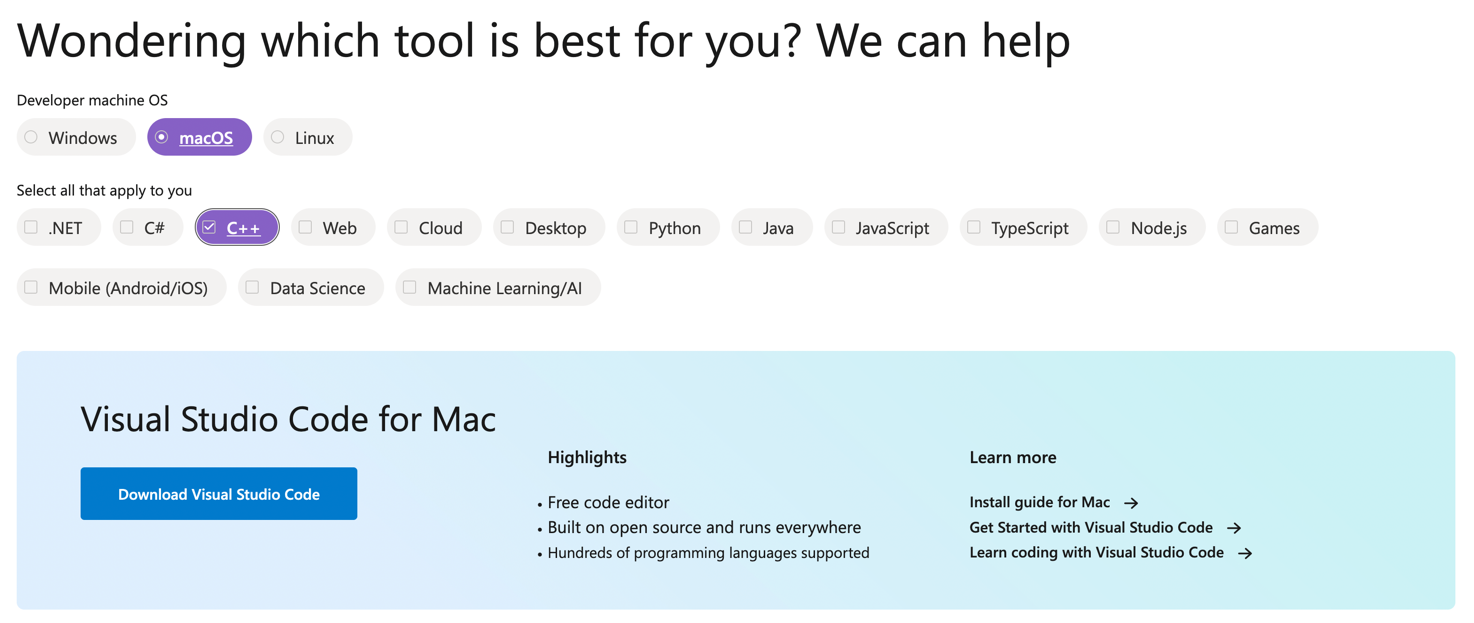


Рисунок 2.2 - Порада Microsoft обрати та встановити середу розробки Visual Studio Code.

Слід завантажити додаткові розширення С/С++ від Microsoft(Рисунок 2.3).

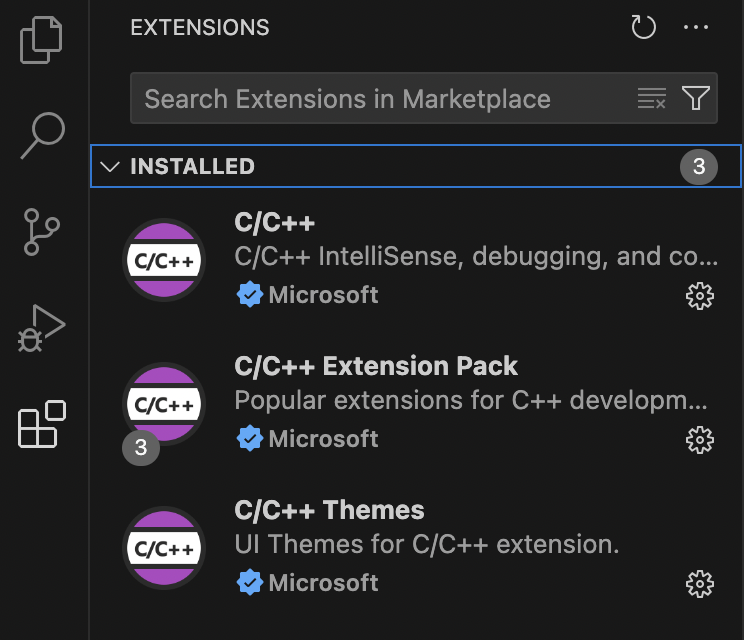


Рисунок 2.3 - – Розширення для роботи на С++ у Visual Studio Code від Microsoft.

Для налаштування коду, тобто щоб компілювалася програма – слід сворити скрипт(Рисунок 2.4) файл формату “.bat”, де ми будемо викликати нашу бібліотеку sfml, створювати та відкривати виконувальний файл.

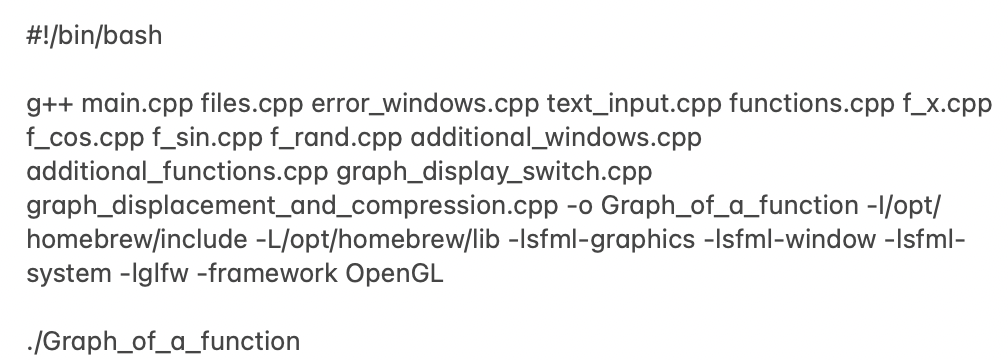


Рисунок 2.4 – Скрипт компіляції та відкриття програми на SFML із додатковими файлами.

## Опис розв’язку завдання

### Вид графіку

Для зовнішнього вигляду графіка я взяв точки типу «CircleShape» радіусом два пікселі, бо це простий спосіб відображення графіка.

### Вивід графіків на екран

Беремо наш інтервал(Рисунок 1.5), який вказав користувач; і кількість точок(Рисунок 1.7), яка також встановлюється користувачем. Якщо у нас одна точка, то зрозуміло, що її треба брати в середині проміжка(Рисунок 1.5). Якщо ми маємо 2 або більше точок, то ми можемо припустити, що якщо ми маємо n кількість відрізків на інтервалі, то ми маємо (n+1) вузлову точку разом із крайовими. Тому наше завдання зводиться до розрахунку відстані цього кроку, тобто кроку між сусідніми вузловими точками, підкреслюю, що для рівномірного відображення крок між точками ставився однаковим. Усі точки зберігаються у масивах xi, yi, які в свою чергу зберігаються у конретному класі функції.

### Вивід та ввід у файл точок

Щоб записати дані у файл я використав додаткову юіюліотеку “fsream” для роботи з файлами. Коли я отримував від користувача назву файлу, то приписував до нього розширення “.txt”, щоб працювати із текстовим файлом. Далі створювався цей файл та відкривався. Першим числом записується кількість точок для графіків F(x), sin(x), cos(x) - усі вони мають однакову кількість точок. Якщо кількість точок нульова, тобто їх немає – пишемо 0, якщо ж там є якесь число, то ми прописуємо проміжки функцій(Рисунок 1.5) - вони однакові, далі записуємо через пробіл точки х та у для функції F(x), так само потім для cos(x) і sin(x). Далі йде перевірка, якщо рандомний графік був створенний, то має свою кількість точок. Якщо точок немає – пишемо нуль та закриваємо файл. Якщо там щось є – знов пишемо розмір а потім наш допоміжний інтервал для рандомної функції(Рисунок 1.11) і так само, як і точки для F(x) запишемо точки для рандомної функції. Закриємо файл та виведемо повідомлення про збереження(Рисунок 1.22).

Щоб дістати дані із фалу, беремо в користувача назву файлу та додаємо розширення “.txt”. Якщо не вдалося відкрити, то виводимо відповідне повідомлення(Рисунок 1.25). Якщо вдалося, то перевірка на нуль. Якщо нуль – значить функцій F(x), cos(x), sin(x) – немає, йдемо далі. Якщо ж там все ж таки щось є, то надаємо нашим масивам точок той розмір, який є у файлі. Далі передаємо інтервал(Рисунок 1.5) до полів вводу(Рисунок 1.5), і зчитуємо х і у точки із функцій: F(x), cos(x), sin(x). За тим йде перевірка на наявність розміра у рандомної функції. Якщо кількість точок рандомної функції нульова – її немає. Якщо у функції щось є - надаємо нашим масивам точок той розмір, який є у файлі. Далі передаємо інтервал(Рисунок 1.11) до полів вводу(Рисунок 1.11), і зчитуємо х і у точки із рандомної функції. Закриваємо файл та виводимо вікно(Рисунок 1.24), що дані взяті з файлу.

### Масштабування графіку

Для маштабування будь-якого графіка бралися коефіцієнти K(Рисунок 1.16), якщо він нульовий, то виводилося повідомлення(Рисунок 1.19) про дивні дії. Якщо не нульовий, то в нас є відповідні точки (xi, yi), точка початку координат (x0,y0) та наш коефіцієнт стискання. Нагадаємо, що розмір точок, з який складається графік, є радіусом 2 байта. Віднімаємо початкові координати від кожної точки, до кожної точки додаємо 2, кожну точку помножуємо на коефіцієнт, знов додаємо початкові координати та віднімаємо 2. Це зроблено так, бо в компʼютері точка (0,0) починається із вірхнього лівого кута.

### Паралельний переніс графіку

Для паралельного переносу будь-якого графіка бралися коефіцієнти k(Рисунок 1.15), якщо він нульовий, то виводилося повідомлення(Рисунок 1.19) про дивні дії. Якщо не нульовий, то в нас є відповідні точки (xi, yi) та наш коефіцієнт зсуву. Додаємо до кожної точки наш коефіцієнт.

### Виведення нового вікна на екран

Для виводу нового вікна використовувалося те, що можна створювати ще вікна sfml. Тобто додаткові вікна створювалися так само, як і головне вікно(Рисунок 1.1), але там були свої розміри назви, кнопки, тексти, поля вводу. Всі вікна закриваються за допомогою команди “window.close()” при натисненні на певну кнопку або крестик.

### Формула для розрахунку кроку

Якщо у нас є одна точка для відображення графіка, то це має бути середнє арифметичне нашого проміжку(Рисунок 1.5). Якщо у нас більше однієї точки, то ми маємо крок, тобто початковий x у точці A(Рисунок 1.5), і рівновіддалені точки чкількості N(Рисунок 1.7) до точки B(Рисунок 1.5). Для цього розрахуємо крок: від кінцевої точки віднімаємо початкову і ділимо на кількість відрізків. Враховуємо, що якщо точок n, то кількість рівномірних відрізків дорівнює (n-1).

# Аналіз отриманих результатів

Перевіримо правильність відображення графіка індивідуального завдання. Для цього побудуємо його на проміжку [-12,12] у мої програмі написаній на С++(Рисунок 3.1) та у [WOLFRAM CLOUD](https://www.wolframcloud.com/)(Рисунок 3.2).

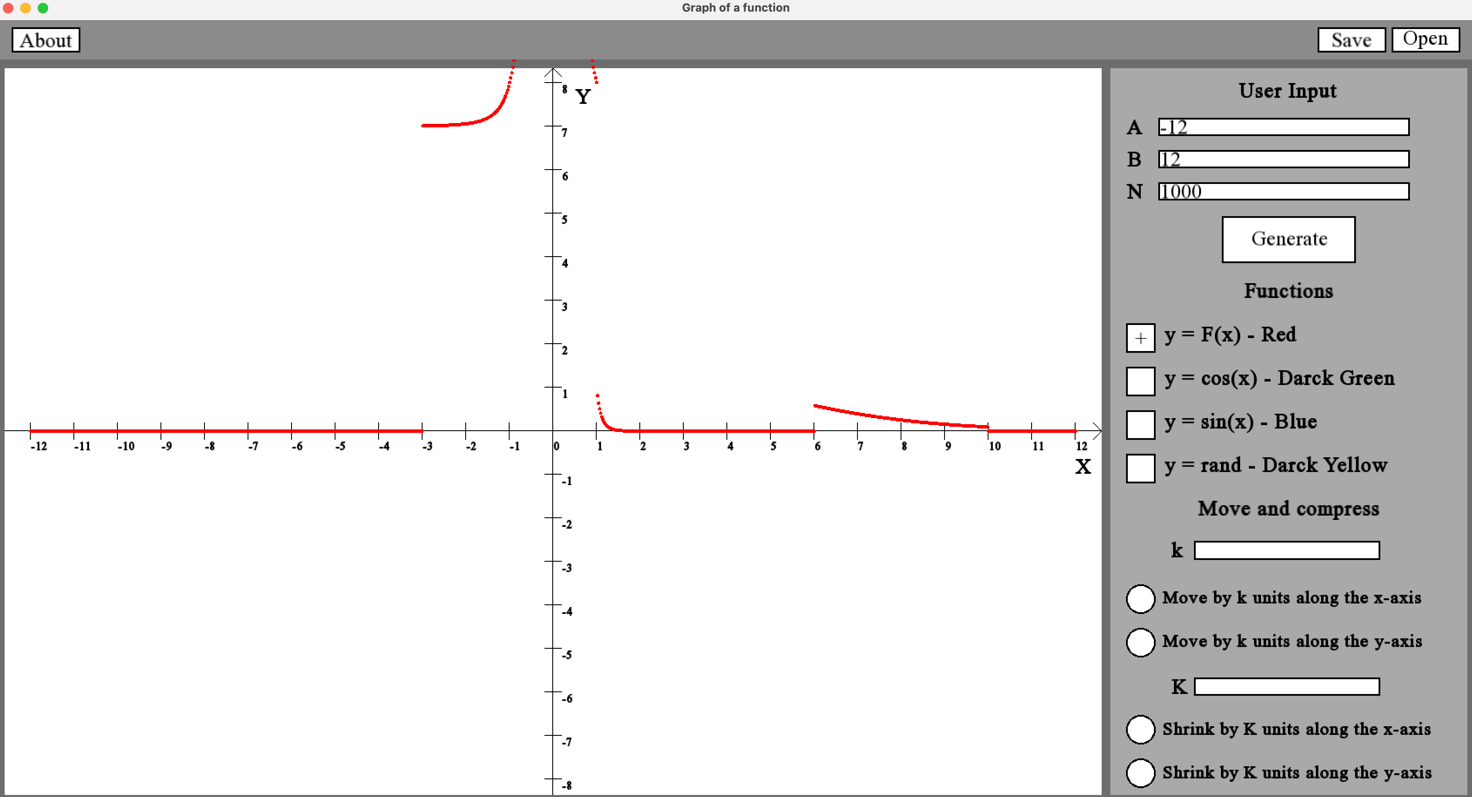


Рисунок 3.1 – Графік індивідуальної функції у зробленій програмі.

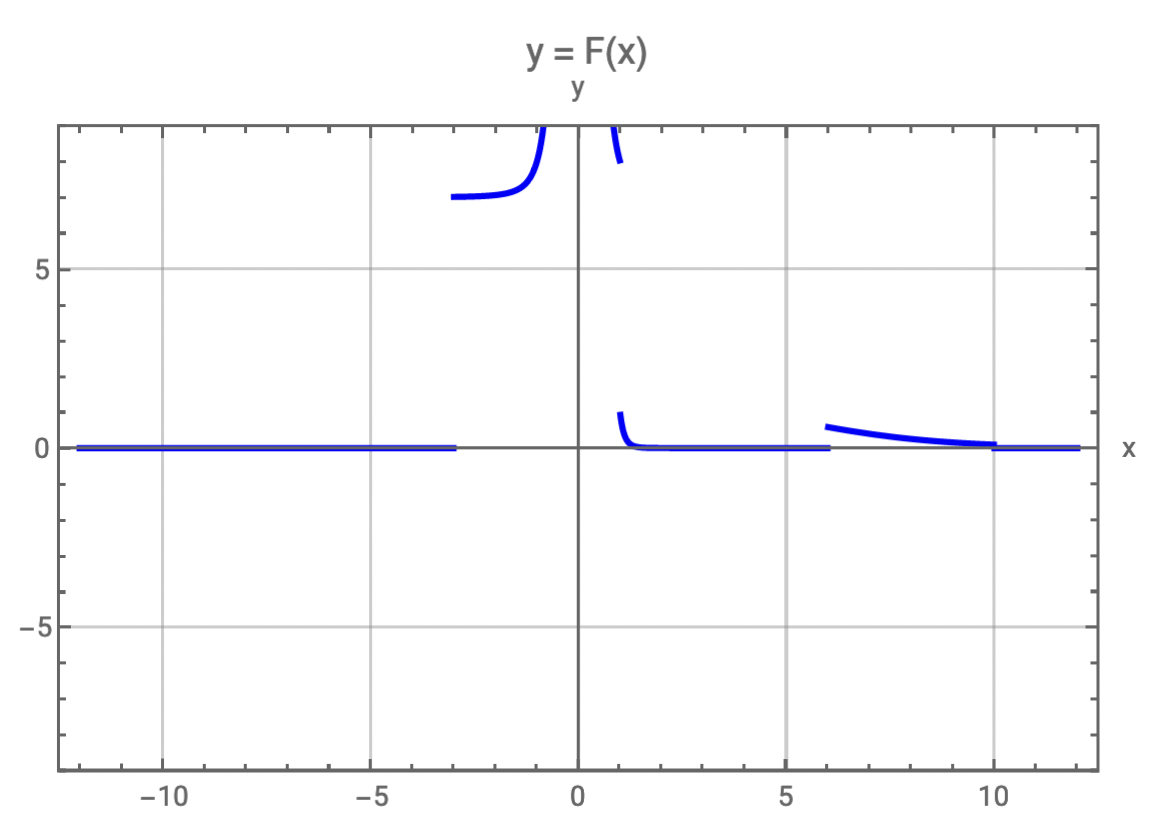


Рисунок 3.2 – Графік індивідуальної функції у WOLFRAM CLOUD.

Можна побачити, що графіки на С++(Рисунок 3.1) та у [WOLFRAM CLOUD](https://www.wolframcloud.com/)(Рисунок 3.2) нічим не відрізняються, крім кольору відмальовки, тобто я зробив правильну функцію індивідуального варіанта та коректно її відобразив.

# Висновки

Під час проходження практики з комп’ютерних технологій я навчився візуальному програмуванню на C++ за допомогою графічної бібліотеки SFML.

Дослідження мало особливо значний вплив на:

* RectangleShape - створення прямокутників
* Font - створення шрифту тексту
* Text – створення тексту
* VertexArray - створення масиву ліній
* CircleShape - створення точок для відображення на них графіків

Тепер я можу створити додаток із різними кольорами та стилями. Знаю, як створювати різні точкові графіки з певними інтервалами та відтінками кольорів.

# Список використаних джерел

**Рекомендовані Джерела**

**Основні**

1. Гук Н.А., Єгошкін Д.І., Сірик С.Ф. Алгоритм класифікації на базі нечіткої логіки з розширюваною кількістю виводів // Зб. наук. праць «Питання прикладної математики і математичного моделювання». – Дніпро. – 2018. – Вип. 18. – С. 33-41.
2. Приклад коду програмування графіки в Microsoft Visual Studio [http://khizha.dp.ua/library/graphics\_in\_WindowsForms/graphics in WF.rar](http://khizha.dp.ua/library/graphics_in_WindowsForms/graphics%20in%20WF.rar)
3. Козак Л. І., Костюк І. В., Стасевич С. П. Основи програмування: навчальний посібник – Львів:«Новий Світ-2000», 2020. – 328с.

**Додаткові**

1. Юрій Грицюк, Тарас Рак. ПРОГРАМУВАННЯ МОВОЮ С++, Львів Вид-во ЛДУ БЖД 2011,- 288 с
2. Трофименко О.Г., Прокоп Ю.В., Швайко І.Г., Буката Л.М., Косирева Л.А.. Основи програмування. Теорія, практика: підручник. Одеса, Фенікс 2010, 534 с.
3. Ковалюк Т. В.К 56 Алгоритмізація та програмування: Підручник. — Львів: «Магнолія 2006», 2013. — 400 с.
4. Нестеренко В., Краєвська О., Чермних І., Адашевська І., Сілічев А. Основи інженерної графіки з елементами професійного конструювання, Київ: Кондор. -2020. -220с.

**Інформаційні ресурси**

Інформаційне забезпечення ґрунтується на використанні ресурсів: загально університетських та кафедральних бібліотек, мережі Internet з вільним доступом та колекцій цифрового репозиторію університету (<http://repository.dnu.dp.ua:1100/?page=inner_dep&id=4> ).

1. Початок до програмування графіки <http://msdn.microsoft.com/ru-ru/library/da0f23z7.aspx>
2. Клас. Graphics <http://msdn.microsoft.com/ru-ru/library/system.drawing.graphics.aspx>
3. Робота в графіці у платформі .NET Framework на мові C++ <http://msdn.microsoft.com/ru-ru/library/9be4455y%28v=vs.90%29.aspx>
4. Швидко розбираємося з C++ CLI (пояснення про посилання) <http://habrahabr.ru/post/111071/>
5. Візуальне проектування доданків на мові C# [www.frolov-lib.ru/books/msnet/c-sharp2/ch03.html](http://www.frolov-lib.ru/books/msnet/c-sharp2/ch03.html)
6. Робота з графікою у бібліотеці sfml на мові С++ <https://www.sfml-dev.org/tutorials/2.6/>