Міністерство освіти і науки України

Національний університет «Полтавська політехніка імені Юрія Кондратюка»

Навчально-науковий інститут інформаційних технологій та робототехніки

Кафедра комп’ютерних та інформаційних технологій і систем

**Пояснювальна записка**

**до курсової роботи з дисципліни «Програмування»**

**на тему «Розробка додатку з використанням мови Python»**

Виконав студент гр. 203-ТК

Гординський Максим Сергійович

Керівник курсової роботи

к.ф-м.н., доцент

Двірна О.А.

Дата захисту:

« » травня 2024р.

Оцінка:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Полтава 2024 р.

Міністерство освіти і науки України

Національний університет “Полтавська політехніка імені Юрія Кондратюка”

Навчально-науковий інститут інформаційних технологій та робототехніки

Кафедра комп’ютерних та інформаційних технологій і систем

**ЗАВДАННЯ**

**до курсової роботи з дисципліни «Програмування»**

**на тему «Розробка додатку з використанням мови Python»**

**Група** 203ТК

**Студент Гординський Максим**

**Завдання:**

Розробити та реалізувати мовою Python програму - телефонний довідник, що містить такі відомості: ПІБ, адреса, електронна пошта, телефон (мобільний, домашній). Дані зберігаються в csv файлі.

Функції роботи довідника:

- додавання нових контактів;

- редагування існуючих контактів. При цьому на одного абонента може припадати кілька номерів телефону;

- перегляд існуючих контактів. Контакти повинні виводитися у відсортованому порядку;

- видалення контакту;

- пошук за параметрами – Прізвище, номер телефону.

Програма повинна містити довідку, де описуються основні можливості програми та інструкція для користувача.

Структура пояснювальної записки:

Титульна сторінка Завдання

Зміст

Вступ

Розділ 1. Теоретична частина.

Розділ 2. Практична частина.

Висновки

Список використаних джерел

Додатки

Дата захисту курсової роботи 24.05.24.

Завдання видала к.ф.-м.н., доц. Двірна О.А

**Зміст**

ВСТУП….………………………………………………………………………….4

РОЗДІЛ 1 ТЕОРЕТИЧНА ЧАСТИНА…………………………………………....6

1.1. Опис мови програмування та її переваг для розробки додатка…….....6

1.2. Опис бібліотеки для розробки додатків та елементів інтерфейсу.……..7

1.3. Опис використаних бібліотек………………….………………………………………...10

РОЗДІЛ 2 ПРАКТИЧНА ЧАСТИНА…………………………………………...12

2.1. Імпорт модулів та бібліотек……………………………………….…12

2.2. Розробка форм додатку, їх опис та реалізація……………………...13

2.3. Опис процедур і функцій…………………………………………….14

2.4. Опис структури програми та інтерфейсу користувача……………..19

2.5. Тестування додатку

ВИСНОВКИ……………………………………………………………………...26

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ………………………………….…….27

Додаток А. Програмний код тестового додатку ………………………………29

**ВСТУП**

Сьогодні Python використовується практично у всіх сферах програмування, від веб-розробки до аналізу даних та розробки продуктів штучного інтелекту. Розробка мобільних додатків на Python також набуває популярності і стає важливою галуззю для цієї мови програмування.Хоча розробка мобільних додатків на Python все ще розвивається, універсальність мови дозволяє створювати програмне забезпечення для різних мобільних операційних систем. Це може повторити успіх JavaScript, який став невід'ємною частиною мобільної розробки.Особливо цінні можливості Python в розробці штучного інтелекту, аналізу даних та програм, що інтегруються з Інтернетом речей. Python — це популярна мова програмування загального призначення з чітким синтаксисом та багатими бібліотеками.Основні переваги Python для мобільних додатків — це велика спільнота розробників, фреймворки для швидкої розробки та широкі можливості використання штучного інтелекту та аналізу даних.Python зараз є другою за поширеністю мовою програмування у світі, і розвиток мобільної розробки на цій мові буде швидко зростати. Розробка на Python актуальна, особливо для проектів, які вимагають обробки великих обсягів даних та роботи з алгоритмами штучного інтелекту. Ця мова програмування вже довела свою ефективність та переваги в багатьох галузях, і мобільна розробка — не виняток.

Мета курсової роботи – розробити додаток “Телефонний довідник” засобами мови програмування Python, який буде містити такі функції:

- додавання нових контактів;

- редагування існуючих контактів. При цьому на одного абонента може припадати кілька номерів телефону;

-перегляд існуючих контактів. Контакти повинні виводитися у відсортованому порядку;

- видалення контакту;

- пошук за параметрами – Прізвище, номер телефону.

**Для досягнення поставленої мети** необхідно виконати такі завдання:

- вибір інструментів розробки;

- розробка дизайну/макету додатку;

- написання коду додатку відповідно до технічного завдання;

- тестування додатку.

**Об’єкт дослідження** процес розробки додатків засобами мови Python.

**Предмет дослідження** – розробка додатку “Телефонний довідник” з використанням мови Python.

Python – мова програмування, яку все ще можна вважати новою з точки зору використання для розробки мобільних додатків, проте вона вже встигла довести свою ефективність та переваги.

Отже, написання додатків на Python є актуальним завдяки своїм перевагам у простоті, універсальності та ефективності, а також великому співтовариству та широкій спеціалізації в різних галузях розробки програмного забезпечення.

**РОЗДІЛ 1 ТЕОРЕТИЧНА ЧАСТИНА**

Python - це високорівнева мова програмування, яка володіє другим за популярністю місцем у світі. Вона застосовується для створення веб-застосунків, програмного забезпечення та машинного навчання. Компанії, такі як Google, Instagram, Facebook, IBM, NASA, Dropbox, Netflix і багато інших, використовують Python для вирішення своїх завдань.Розробники оцінюють Python за його простоту вивчення, ефективність та мультиплатформність. Розробка додатків на Python може відбуватися в різних областях, таких як веб-розробка, наукові обчислення, штучний інтелект, обробка даних та інші.Однією з переваг Python є наявність великої кількості бібліотек та фреймворків, які спрощують розробку різноманітних застосунків. Він дозволяє розробникам швидко та ефективно створювати програмне забезпечення для різних потреб і галузей, забезпечуючи при цьому високу якість та продуктивність.

* 1. **Опис мови програмування та її переваг для розробки додатка**

Python - це скриптова мова програмування з простим синтаксисом, яка набула великої популярності. Вона є доступною для новачків, але здатною вирішувати різноманітні завдання для професіоналів. Python має мультиплатформний характер, що дозволяє працювати з ним на різних операційних системах і в різних галузях, від розробки мобільних додатків до штучного інтелекту.Особливістю мови є динамічна типізація, що означає можливість передавати різні типи даних до функцій без попередньої декларації. Інтерпретованість Python дозволяє виявляти помилки у коді на ранніх етапах, що сприяє розробці більш надійного програмного забезпечення. Однак, серед можливих недоліків Python можна виділити меншу швидкість виконання коду порівняно з компільованими мовами програмування. Код на Python спочатку компілюється у внутрішній байт-код, що може зробити програми повільнішими, особливо в порівнянні з мовами, такими як C. Щодо розробки мобільних додатків на Python, наразі цей напрямок тільки починає набирати обертів. Опитування Statista свідчить про те, що лише невеликий відсоток розробників використовує Python для мобільних продуктів. Однак, з огляду на поширеність та універсальність мови, це може змінитися в майбутньому, особливо у зв'язку з розвитком технологій штучного інтелекту та аналізу даних.Загалом, використання Python для розробки додатків має численні переваги, включаючи зручний синтаксис, високу сумісність з різними платформами, доступ до багатьох бібліотек і фреймворків, а також можливість використання сучасних технологій Big Data і Machine Learning.

* 1. **Опис бібліотеки для розробки додатків та елементів інтерфейсу**

Бібліотека у мові програмування - це файл з кодом програми, призначений для зберігання часто використовуваних функцій, класів та сталих. Вона дозволяє зручно організувати код та спрощує його повторне використання.У мові Python існує стандартна бібліотека модулів, яка включає в себе різноманітні функції та інструменти для різних завдань. Ці модулі можуть бути імпортовані у програми для використання їхніх функцій і можливостей.Графічний інтерфейс (GUI) важливий для багатьох програм і веб-сайтів, оскільки він створює сприятливе середовище для користувачів. У Python є кілька бібліотек для розробки GUI:PyQt5: цей фреймворк базується на Qt, який є міжплатформовим інструментом для розробки програм. PyQt5 дозволяє легко створювати GUI, надаючи широкий спектр елементів інтерфейсу та підтримку кросплатформеності.PySimpleGUI: це простий фреймворк для швидкої розробки GUI в Python. Він базується на різних бібліотеках GUI, таких як Qt, Tkinter, wxPython і Remi, і надає простий спосіб створення інтерфейсів користувача для початківців.Libavg: це інша бібліотека для розробки GUI в Python, яка спеціалізується на сенсорних пристроях і використовує апаратне прискорення за допомогою шейдерів OpenGL і GPU.PyForms: це фреймворк, який дозволяє розробникам створювати високоінтерактивні інтерфейси для різних платформ, включаючи Windows, веб та термінал.Ці бібліотеки роблять розробку GUI в Python простою і ефективною, дозволяючи розробникам створювати привабливі та функціональні програми.

* 1. **Опис використаних бібліотек**

У роботі використано дві біліотеки мови Python (tkinter і csv). Додатково, ви використані підмодулі та класи з tkinter, такі як messagebox, simpledialog та ttk, які допомагають у реалізації різних елементів інтерфейсу та вікон повідомлень.

Ще однією найкращою бібліотекою Python для GUI є Tkinter, яка є бібліотекою Python Graphic User Interface з відкритим кодом. Він добре відомий своєю простотою та попередньо встановлений на Python, тобто з вашого боку не потрібно працювати. Ці особливості роблять його чудовим вибором для початківців і середнього рівня, але він не здатний виконувати більш масштабні проекти.

У Tkinter візуальні елементи називаються віджетами, і кожен з віджетів має різний рівень настроюваності. Він також пропонує широкий спектр часто використовуваних елементів, з якими вже знайомі багато розробників, як-от рамка, кнопки, кнопки-перевірки, мітки, діалогові вікна файлів і полотно.[5]

Так званий формат CSV (значення, розділені комами) є найпоширенішим форматом імпорту та експорту для електронних таблиць і баз даних. Формат CSV використовувався протягом багатьох років до того, як його спробували описати стандартизованим способом. Відсутність чітко визначеного стандарту означає, що часто існують тонкі відмінності в даних, які створюються та споживаються різними програмами. Ці відмінності можуть дратувати обробку файлів CSV із кількох джерел. Тим не менш, хоча розділювачі та символи лапок відрізняються, загальний формат досить схожий, щоб можна було написати один модуль, який може ефективно маніпулювати такими даними, приховуючи деталі читання та запису даних від програміста.

**РОЗДІЛ 2 ПРАКТИЧНА ЧАСТИНА**

Python – це високорівнева мова програмування загального призначення. Це означає, що ви можете запрограмувати [все, що завгодно](http://codeguida.com/post/406/), починаючи з простої гри і закінчуючи веб-сайтом, який може підтримувати мільйони користувачів щомісяця.

**2.1. Імпорт модулів та бібліотек**

У курсовій роботі було використано дві бібліотеки – TkInter та CSV. Кожен рядок цього коду встановлює певні можливості для роботи з бібліотекою tkinter у вашій програмі на Python. Проаналізуємо кожен рядок:

“import tkinter as tk”: Цей рядок імпортує модуль “tkinter” та створює псевдонім “tk”, щоб зробити доступ до функцій та класів модуля зручнішим.

“from tkinter import messagebox, simpledialog”: Цей рядок імпортує підмодулі “messagebox” та “simpledialog” з бібліотеки “tkinter”. “messagebox” дозволяє вам виводити діалогові вікна з повідомленнями, а “simpledialog” дозволяє створювати прості діалогові вікна для введення даних користувачем.

“from tkinter import ttk”: Цей рядок імпортує модуль “ttk”, який містить класи для створення стильних віджетів (включаючи елементи керування, такі як кнопки та поля введення) для використання разом з “tkinter”.

“import csv”: Цей рядок імпортує модуль “csv”, який дозволяє працювати з файлами CSV (Comma-Separated Values), що дуже зручно для читання та запису даних у таблиці.

Цей код підготовлює програму до використання різноманітних інструментів та можливостей для роботи з графічним інтерфейсом користувача (GUI), обробки повідомлень, роботи з віджетами та роботи з файлами CSV.

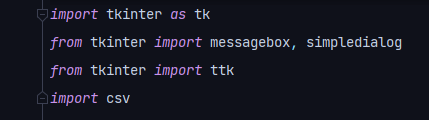


Рисунок 2.1.1 – Імпорт бібліотек

**2.2. Розробка форм додатку, їх опис та реалізація**

Розробляємо головне вікно для додатку. Вносимо основний заголовок. Встановлюємо розміри вікна. І змінюємо колір. Для цього, ми використовуємо:

“window = tk.Tk()”: Цей рядок створює нове вікно програми за допомогою класу “Tk” з модуля “tkinter”. Об'єкт ”window” стає основним вікном вашого програмного додатку.

“ window.title(«Телефонна книга»)”: Цей рядок встановлює заголовок вікна на «Телефонна книга». Це буде відображатися у верхній частині вікна, коли програма запущена.

“ window.geometry(«500x300»)”: Цей рядок встановлює розмір вікна на ширину 500 пікселів та висоту 300 пікселів. Вікно буде мати такі розміри при запуску програми.

“ window [«bg»] = «light grey»”: Цей рядок встановлює колір фону вікна на світло-синій. “bg” - це атрибут, який відповідає за фоновий колір вікна. Шляхом присвоєння рядкового значення «light grey», фон буде встановлено у світло-сірий колір.

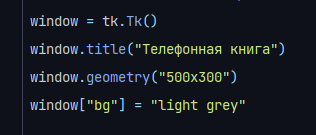


Рисунок 2.2.1.– Розробка головного вікна

Вносимо все, що буде відображатися на головному вікні (текст,кнопки і т.д.), підбираємо кольори. У кінцевому результаті маємо:

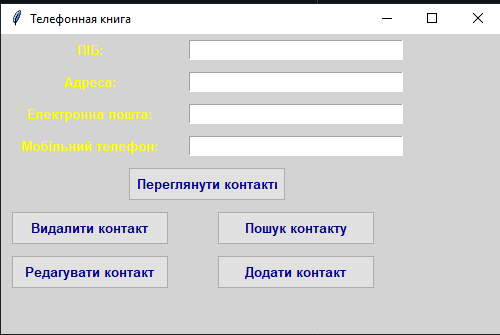


Рисунок 2.2.2. – Головне вікно додатка

**2.3. Опис процедур і функцій**

Код має декілька основних функцій, які забезпечують роботу програми телефонної книги. Ось перелік функцій і їх призначення:

**“add\_contact()”:** збирає дані з полів вводу (ім'я, адреса, електронна пошта, мобільний телефон), додає ці дані у файл “contacts.csv”, виводить повідомлення про успішне додавання контакту.

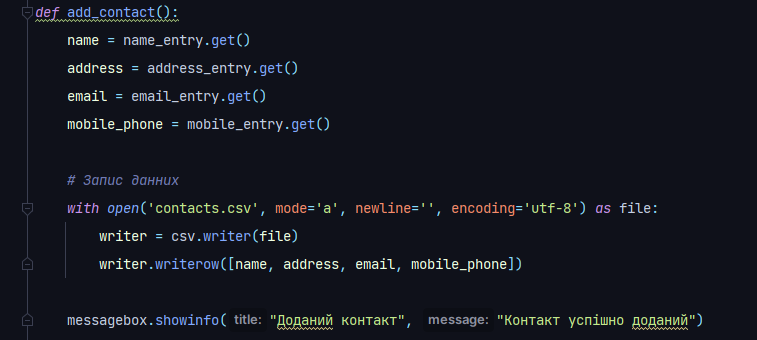


Рисунок 2.3.1. – Функція додавання контактів

**“view\_contacts()”:**  читає контакти з файлу “contacts.csv”, сортує контакти за іменем, відображає контакти у діалоговому вікні або виводить повідомлення, якщо контактів немає.

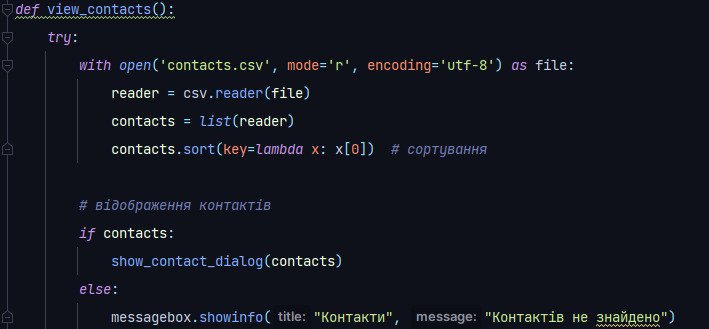


Рисунок 2.3.2. – Функція для перегляду контактів

**“show\_contact\_dialog(contacts)”:** відображає контакти у нових вікнах, має кнопку "Далі", яка дозволяє переглядати наступний контакт у списку.



Рисунок 2.3.3. – Функція для відображення діалогу з контактами

**“delete\_contact()”:** запитує у користувача ім'я контакту, який потрібно видалити, читає контакти з файлу contacts.csv. Видаляє контакт з файлу, записує оновлений список контактів у файл contacts.csv, виводить повідомлення про успішне видалення контакту.

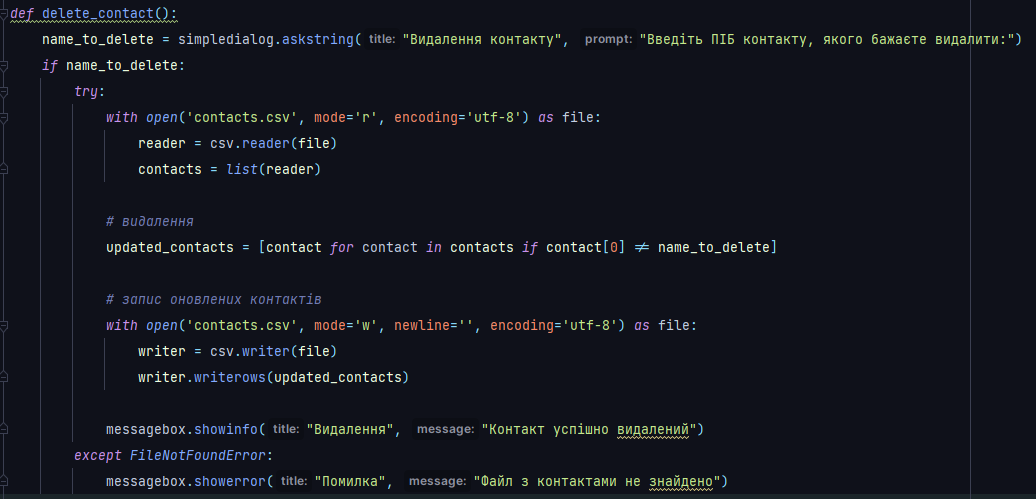


Рисунок 2.3.5. – Функція для видалення контакту

**“search\_kontakt()”:** запитує у користувача критерій пошуку (ім'я або номер телефону), читає контакти з файлу contacts.csv, шукає контакти, що відповідають критеріям пошуку, відображає знайдені контакти або виводить повідомлення, якщо контакти не знайдені.

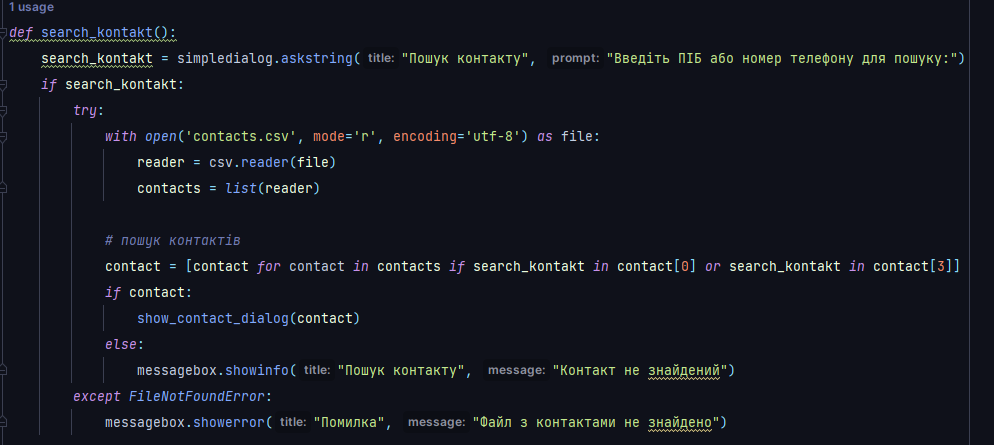


Рисунок 2.3.5. – Функція для пошуку контакту

**“edit\_contact()”:** Запитує у користувача ім'я контакту, який потрібно редагувати, читає контакти з файлу contacts.csv, Заповнює поля вводу даними знайденого контакту для редагування, має внутрішню функцію save\_changes() (яка: збирає оновлені дані з полів вводу, оновлює контакт у файлі contacts.csv, виводить повідомлення про успішне оновлення контакту.)

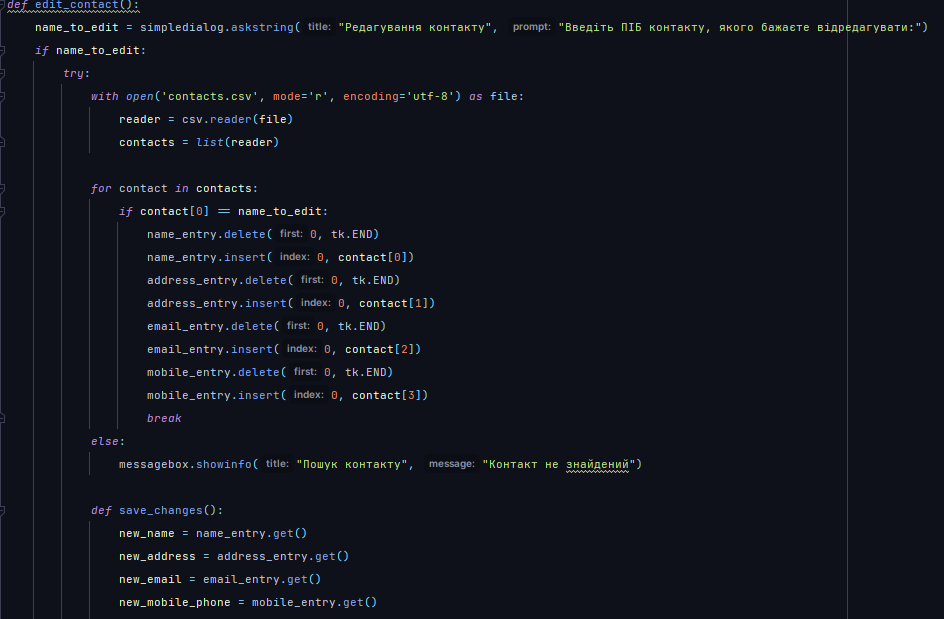
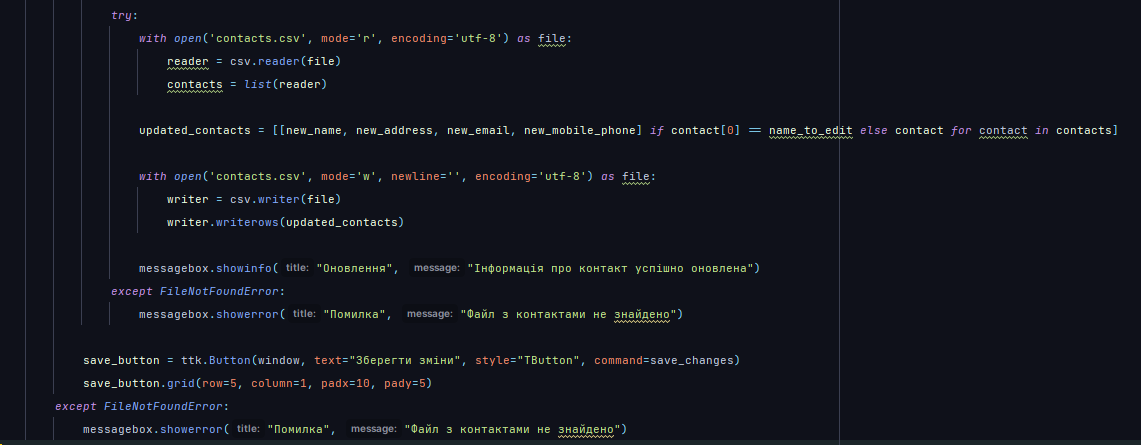
 

Рисунок 2.3.6. – Функція редагування контакту

Окрім цих функцій, код також містить налаштування основного вікна, створення віджетів для вводу даних та кнопок для виклику цих функцій. Основний цикл програми (“window.mainloop()”) забезпечує безперервну роботу графічного інтерфейсу, дозволяючи користувачам взаємодіяти з додатком.

**2.4. Опис структури програми та інтерфейсу користувача**

Програма складається з кількох основних частин: імпорт модулів, створення головного вікна, функцій для обробки контактів, інтерфейсу користувача та запуску головного циклу програми.

Імпортуються модулі tkinter для створення графічного інтерфейсу та csv для роботи з файлом контактів у форматі CSV. Головне вікно програми (window) ініціалізується з встановленими заголовком, розміром та кольором фону.

**Функції для обробки контактів включають:**

add\_contact(), яка збирає дані з полів вводу (ім'я, адреса, електронна пошта, мобільний телефон), додає ці дані у файл contacts.csv і виводить повідомлення про успішне додавання контакту.

view\_contacts(), яка читає контакти з файлу contacts.csv, сортує їх за іменем і відображає у діалоговому вікні або виводить повідомлення, якщо контактів немає.

show\_contact\_dialog(contacts), яка відображає контакти у нових вікнах із кнопками «Наступний» та «Попередній», що дозволяють переглядати контакти у списку.

delete\_contact(), яка запитує у користувача ім'я контакту, який потрібно видалити, читає контакти з файлу contacts.csv, видаляє контакт із файлу та записує оновлений список контактів, виводячи повідомлення про успішне видалення контакту.

search\_contact(), яка запитує у користувача критерій пошуку (ім'я або номер телефону), читає контакти з файлу contacts.csv, шукає відповідні контакти та відображає їх або виводить повідомлення, якщо контакти не знайдені.

edit\_contact(), яка запитує у користувача ім'я контакту, який потрібно редагувати, читає контакти з файлу contacts.csv, заповнює поля вводу даними знайденого контакту, дозволяє користувачу вносити зміни, а потім оновлює контакт у файлі contacts.csv.

Інтерфейс користувача включає створення міток (label) та полів вводу (entry) для введення інформації про контакт. Мітки мають шрифт Arial, 10, жирний, жовтий колір тексту та світло-сірий фон. Поля вводу мають шрифт Arial, 10, та ширину 30 символів. Створюються кнопки для виконання різних дій (додавання, перегляд, видалення, пошук та редагування контактів). Кнопки мають шрифт Arial, 10, жирний, темно-синій колір тексту, світло-сірий фон, ширину 20 символів, відступи 5 пікселів та втоплений стиль.

Запуск головного циклу програми (window.mainloop()) забезпечує роботу інтерфейсу.

Користувач може додавати контакт, вводячи дані у поля та натискаючи кнопку «Додати контакт». Програма зберігає дані у файл CSV. Користувач може переглядати контакти, натискаючи кнопку «Переглянути контакти». Програма читає дані з файлу CSV і відображає їх у нових вікнах. Для видалення контакту користувач натискає кнопку «Видалити контакт», вводить ім'я контакту, який хоче видалити, і програма видаляє відповідний контакт із файлу CSV. Для пошуку контакту користувач натискає кнопку «Пошук контакту», вводить критерій пошуку (ім'я або номер телефону), і програма знаходить відповідні контакти та відображає їх. Для редагування контакту користувач натискає кнопку «Редагувати контакт», вводить ім'я контакту, який хоче редагувати, і програма заповнює поля вводу даними цього контакту. Користувач вносить зміни і натискає кнопку «Зберегти зміни», і програма оновлює дані у файлі CSV.

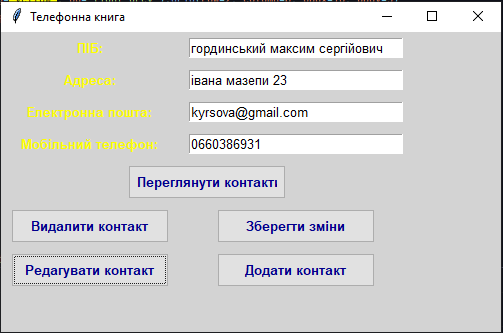


Рисунок 2.4.1. – Інтерфейс при редагуванні контакту

**2.5. Тестування додатку**

Тестування додатку "Телефонна книга" проводиться з метою перевірки коректності роботи всіх функціональних можливостей, а саме: додавання, перегляду, пошуку, редагування та видалення контактів. При роботі додатки, виконуємо такі дії:

Запуск програми: відкрито головне вікно додатку з заголовком "Телефонна книга", розміром 500x300 пікселів та світло-сірим фоном.

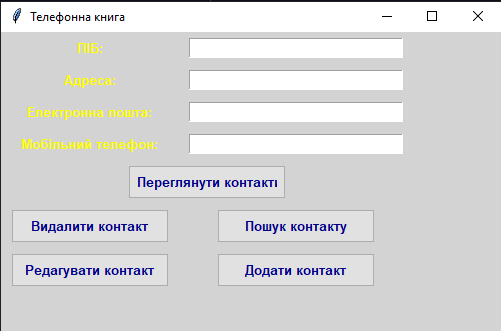


Рисунок 2.5.1. – Головне меню додатка

Додавання контакту: у відповідні поля були введені тестові дані (ім'я, адреса, електронна пошта, мобільний телефон). Натиснуто кнопку «Додати контакт». Перевірено файл contacts.csv на наявність нового контакту. Контакт було успішно додано, а повідомлення "Контакт успішно доданий" з'явилося на екрані.

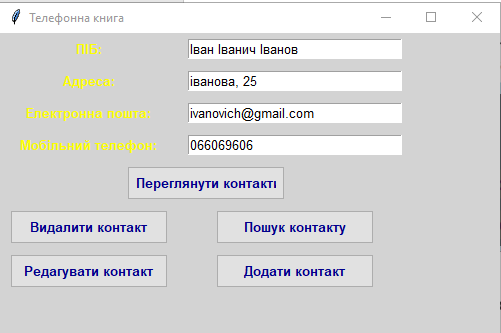
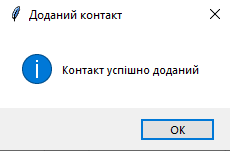
 

Рисунок 2.5.2. – Інтерфейс, при додаванні контакту

**Перегляд контактів:** натиснуто кнопку «Переглянути контакти», у нових вікнах були відображені всі додані контакти, сортування виконувалось за ім'ям контакту.

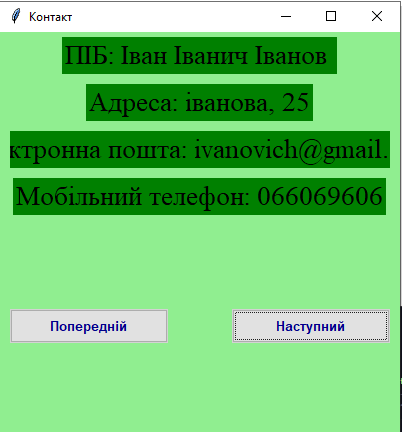


Рисунок 2.5.3. – Інтерфейс, при перегляді контактів

**Пошук контакту:** натиснуто кнопку "Пошук контакту", введено ім'я або номер телефону для пошуку, у випадку наявності відповідних контактів, вони були відображені у новому вікні, у випадку відсутності контактів, з'явилося повідомлення "Контакт не знайдений".

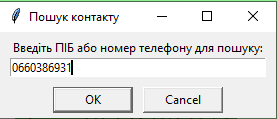
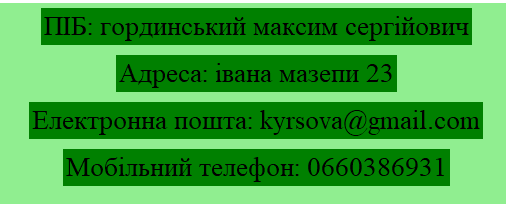
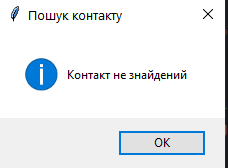
  

Рисунок 2.5.4. – Інтерфейс при пошуку контакту

**Редагування контакту:** натиснуто кнопку «Редагувати контакт», Введено ім'я контакту, який потрібно редагувати, поля вводу заповнилися даними цього контакту, що дозволило змінити інформацію, після внесення змін і натиснення кнопки «Зберегти зміни», дані контакту в файлі “contacts.csv” оновилися. З'явилося повідомлення «Нові данні збереженно».

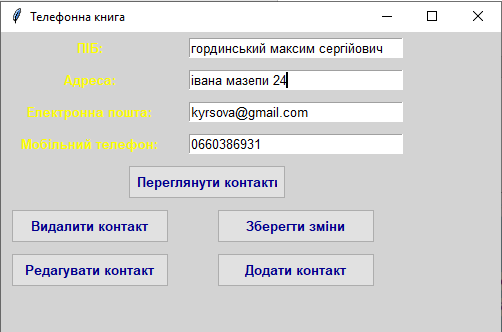
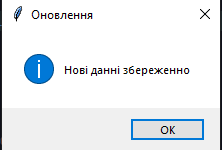
 

Рисунок 2.5.5. – Інтерфейс, при редагуванні контакту

**Видалення контакту:** натиснуто кнопку «Видалити контакт», введено ім'я контакту, який потрібно видалити, після перевірки файл “contacts.cs” не містив видаленого контакту, з'явилося повідомлення «Контакт видалений».

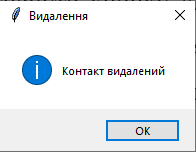


Рисунок 2.5.5. – Інтерфейс, при редагуванні контакту

**ВИСНОВКИ**

У цій курсовій роботі я розробив додаток для телефонної книги, використовуючи мову програмування python та бібліотеку Tkinter, і створив графічний інтерфейс користувача.

Основна мета програми-створити корисний інструмент управління контактами, що надає можливість додавати, переглядати, шукати, редагувати і видаляти контакти в зручному для користувача інтерфейсі, розробити програму, що підтримує різні мови програмування Python, включаючи роботу з файлами, обробку користувальницьких подій, створення графічних інтерфейсів і Обробка винятків.

За допомогою бібліотеки Tkinter тепер ви можете створити простий, інтуїтивно зрозумілий і зручний інтерфейс. Тестування програми показує, що вона працює стабільно і відповідає вимогам таких функцій, як додавання, перегляд, пошук, Редагування та видалення контактів. Всі функції працювали нормально, демонструючи надійну і ефективну роботу всієї програми. Таким чином, додаток "телефонна книга" є успішним результатом курсової роботи, яке дозволяє зручно управляти контактами і задовольняє потреби користувачів в управлінні особистими даними.

**СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ**

1. Які особливості застосування мови програмування Python. URL: <https://freehost.com.ua/ukr/faq/wiki/chto-takoe-jazik-programmirovanija-python/> (дата звернення – 2.05.2024р.).
2. Які переваги Python для розробки додатків. URL: <https://wezom.com.ua/ua/blog/sozdanie-mobilnogo-prilozhenija-na-python> (дата звернення – 2.05.2024р.).
3. Роз’яснення про бібліотеку на Python. URL: <https://www.kievoit.ippo.kubg.edu.ua/kievoit/2016/66_Python/index.html> (дата звернення – 2.05.2024р., 4.05.2024р.)
4. Найпоширеніші бібліотеки Python. URL: <https://www.unite.ai/uk/10-best-python-libraries-for-gui/> (дата звернення – 2.05.2024р.).
5. Роз’яснення про CSV бібліотеку. URL: <https://docs.python.org/uk/3.10/library/csv.html> (дата звернення – 3.05.2024р., 4.05.2024р.)
6. Базурін В. М., Омелечко Є. А., Ковтун А. В. Порівняльний аналіз середовищ програмування мовою Python. *New computer technology*. 2018. Т. 16. С. 281–292. URL: <https://ccjournals.eu/ojs/index.php/nocote/article/view/851>  (дата звернення – 3.05.2024р., 5.05.2024р.).
7. Урок №1 | Відео-курс Tkinter (Python 3) URL: <https://www.youtube.com/watch?v=CB6f6wL_Huc> (дата звернення – 3.05.2024р.).
8. Як працювати з файлами CSV. URL: <https://pythonguide.rozh2sch.org.ua/> (дата звернення – 2.05.2024р., 3.05.2024р., 5.05.2024).
9. Воробйова О. М. Розробка вдосконаленої методики викладання теми «Мова програмування» шкільного курсу з використанням мобільних технологій : магістерська робота. 2020. URL: <https://dspace.znu.edu.ua/jspui/handle/12345/2262>  (дата звернення: 3.05.2024р.).
10. Робота з функціями на Python. URL: <https://sites.google.com/comp-sc.if.ua/pythoneasy/%D0%BE%D1%81%D0%BD%D0%BE%D0%B2%D0%B8-python/%D1%84%D1%83%D0%BD%D0%BA%D1%86%D1%96%D1%97> (дата звернення: 3.05.2024р., 5.05.2024р., 6.05.2024р.).

Посилання на проєкт:

**https://github.com/Gusa4o/kyrsova.git**

Додаток

import tkinter as tk

from tkinter import messagebox, simpledialog

from tkinter import ttk

import csv

# головне вікно

window = tk.Tk()

window.title("Телефонна книга")

window.geometry("500x300")

window["bg"] = "light grey"

# додати контакт

def add\_contact():

name = name\_entry.get()

address = address\_entry.get()

email = email\_entry.get()

mobile\_phone = mobile\_entry.get()

# Запис данних

with open('contacts.csv', mode='a', newline='', encoding='utf-8') as file:

writer = csv.writer(file)

writer.writerow([name, address, email, mobile\_phone])

messagebox.showinfo("Доданий контакт", "Контакт успішно доданий")

# огляд контактів

def view\_contacts():

try:

with open('contacts.csv', mode='r', encoding='utf-8') as file:

reader = csv.reader(file)

contacts = list(reader)

contacts.sort(key=lambda x: x[0]) # сортування

# відображення контактів

if contacts:

show\_contact\_dialog(contacts)

else:

messagebox.showinfo("Контакти", "Контактів не знайдено")

except FileNotFoundError:

messagebox.showerror("Помилка", "Файл з контактами не знайдено")

# маленьке вікно

def show\_contact\_dialog(contacts):

current\_contact = 0

def display\_contact(index):

if 0 <= index < len(contacts):

contact = contacts[index]

name\_label.config(text=f"ПІБ: {contact[0]}")

address\_label.config(text=f"Адреса: {contact[1]}")

email\_label.config(text=f"Електронна пошта: {contact[2]}")

mobile\_label.config(text=f"Мобільний телефон: {contact[3]}")

else:

name\_label.config(text="")

address\_label.config(text="")

email\_label.config(text="")

mobile\_label.config(text="")

def show\_next\_contact():

nonlocal current\_contact

if current\_contact < len(contacts) - 1:

current\_contact += 1

display\_contact(current\_contact)

def show\_previous\_contact():

nonlocal current\_contact

if current\_contact > 0:

current\_contact -= 1

display\_contact(current\_contact)

mini\_window = tk.Toplevel(window)

mini\_window.title("Контакт")

mini\_window.geometry("400x400")

mini\_window["bg"] = "light green"

name\_label = tk.Label(mini\_window, text="", font=("Times New Roman", 20), bg="green")

name\_label.pack(padx=10, pady=5)

address\_label = tk.Label(mini\_window, text="", font=("Times New Roman", 20), bg="green")

address\_label.pack(padx=10, pady=5)

email\_label = tk.Label(mini\_window, text="", font=("Times New Roman", 20), bg="green")

email\_label.pack(padx=10, pady=5)

mobile\_label = tk.Label(mini\_window, text="", font=("Times New Roman", 20), bg="green")

mobile\_label.pack(padx=10, pady=5)

prev\_button = ttk.Button(mini\_window, text="Попередній", command=show\_previous\_contact)

prev\_button.pack(side=tk.LEFT, padx=10, pady=10)

next\_button = ttk.Button(mini\_window, text="Наступний", command=show\_next\_contact)

next\_button.pack(side=tk.RIGHT, padx=10, pady=10)

display\_contact(current\_contact)

# видалення контантів

def delete\_contact():

name\_to\_delete = simpledialog.askstring("Видалення контакту", "Введіть ПІБ контакту, якого бажаєте видалити:")

if name\_to\_delete:

try:

with open('contacts.csv', mode='r', encoding='utf-8') as file:

reader = csv.reader(file)

contacts = list(reader)

# видалення

updated\_contacts = [contact for contact in contacts if contact[0] != name\_to\_delete]

# запис оновлених контактів

with open('contacts.csv', mode='w', newline='', encoding='utf-8') as file:

writer = csv.writer(file)

writer.writerows(updated\_contacts)

messagebox.showinfo("Видалення", "Контакт видалений")

except FileNotFoundError:

messagebox.showerror("Помилка", "Файл з контактами не знайдено")

# пошук

def search\_kontakt():

search\_kontakt = simpledialog.askstring("Пошук контакту", "Введіть ПІБ або номер телефону для пошуку:")

if search\_kontakt:

try:

with open('contacts.csv', mode='r', encoding='utf-8') as file:

reader = csv.reader(file)

contacts = list(reader)

# пошук контактів

contact = [contact for contact in contacts if search\_kontakt in contact[0] or search\_kontakt in contact[3]]

if contact:

show\_contact\_dialog(contact)

else:

messagebox.showinfo("Пошук контакту", "Контакт не знайдений")

except FileNotFoundError:

messagebox.showerror("Помилка", "Файл з контактами не знайдено")

# редагування

def edit\_contact():

name\_to\_edit = simpledialog.askstring("Редагування контакту", "Введіть ПІБ контакту, якого бажаєте відредагувати:")

if name\_to\_edit:

try:

with open('contacts.csv', mode='r', encoding='utf-8') as file:

reader = csv.reader(file)

contacts = list(reader)

for contact in contacts:

if contact[0] == name\_to\_edit:

name\_entry.delete(0, tk.END)

name\_entry.insert(0, contact[0])

address\_entry.delete(0, tk.END)

address\_entry.insert(0, contact[1])

email\_entry.delete(0, tk.END)

email\_entry.insert(0, contact[2])

mobile\_entry.delete(0, tk.END)

mobile\_entry.insert(0, contact[3])

break

else:

messagebox.showinfo("Пошук контакту", "Контакт не знайдений")

def save\_changes():

new\_name = name\_entry.get()

new\_address = address\_entry.get()

new\_email = email\_entry.get()

new\_mobile\_phone = mobile\_entry.get()

try:

with open('contacts.csv', mode='r', encoding='utf-8') as file:

reader = csv.reader(file)

contacts = list(reader)

updated\_contacts = [[new\_name, new\_address, new\_email, new\_mobile\_phone] if contact[0] == name\_to\_edit else contact for contact in contacts]

with open('contacts.csv', mode='w', newline='', encoding='utf-8') as file:

writer = csv.writer(file)

writer.writerows(updated\_contacts)

messagebox.showinfo("Оновлення", "Нові данні збереженно")

except FileNotFoundError:

messagebox.showerror("Помилка", "Файл з контактами не знайдено")

save\_button = ttk.Button(window, text="Зберегти зміни", style="TButton", command=save\_changes)

save\_button.grid(row=5, column=1, padx=10, pady=5)

except FileNotFoundError:

messagebox.showerror("Помилка", "Файл з контактами не знайдено")

tk.Label(window, text="ПІБ:", font=("Arial", 10, "bold"), fg="yellow", bg="light grey").grid(row=0, column=0, padx=10, pady=5)

tk.Label(window, text="Адреса:", font=("Arial", 10, "bold"), fg="yellow", bg="light grey").grid(row=1, column=0, padx=10, pady=5)

tk.Label(window, text="Електронна пошта:", font=("Arial", 10, "bold"), fg="yellow", bg="light grey").grid(row=2, column=0, padx=10, pady=5)

tk.Label(window, text="Мобільний телефон:", font=("Arial", 10, "bold"), fg="yellow", bg="light grey").grid(row=3, column=0, padx=10, pady=5)

name\_entry = tk.Entry(window, font=("Arial", 10), width=30)

name\_entry.grid(row=0, column=1, padx=10, pady=5)

address\_entry = tk.Entry(window, font=("Arial", 10), width=30)

address\_entry.grid(row=1, column=1, padx=10, pady=5)

email\_entry = tk.Entry(window, font=("Arial", 10), width=30)

email\_entry.grid(row=2, column=1, padx=10, pady=5)

mobile\_entry = tk.Entry(window, font=("Arial", 10), width=30)

mobile\_entry.grid(row=3, column=1, padx=10, pady=5)

# створення керуючих кнопок

view\_button = ttk.Button(window, text="Переглянути контакти", style="TButton", command=view\_contacts)

view\_button.grid(row=4, column=0, columnspan=2, padx=10, pady=5)

delete\_button = ttk.Button(window, text="Видалити контакт", style="TButton", command=delete\_contact)

delete\_button.grid(row=5, column=0, padx=10, pady=5)

search\_button = ttk.Button(window, text="Пошук контакту", style="TButton", command=search\_kontakt)

search\_button.grid(row=5, column=1, padx=10, pady=5)

edit\_button = ttk.Button(window, text="Редагувати контакт", style="TButton", command=edit\_contact)

edit\_button.grid(row=6, column=0, padx=10, pady=5)

add\_button = ttk.Button(window, text="Додати контакт", style="TButton", command=add\_contact)

add\_button.grid(row=6, column=1, padx=10, pady=5)

style = ttk.Style()

style.configure("TButton",

foreground="dark blue", # Цвет текста кнопки

background="light grey", # Фоновый цвет кнопки

font=("Arial", 10, "bold"), # Шрифт кнопки

width=20, # Ширина кнопки

padding=5, # Отступы кнопки

relief=tk.RAISED, # Стиль кнопки

)

style.map("TButton",

foreground=[('pressed', 'red'), ('active', 'blue')],

background=[('pressed', '!disabled', 'black'), ('active', 'white')]

)

window.mainloop()