Чернівецький національний університет імені Юрія Федьковича

Факультет математики та інформатики

Кафедра прикладної математики та інформаційних технологій

«ЗАТВЕРДЖУЮ»
Декан факультету

_____ Ольга МАРТИНЮК
«____» _____ 20___ року

РОБОЧА ПРОГРАМА «РОЗРОБКА ІНТЕРАКТИВНИХ ДОДАТКІВ НА PYTHON»

вибіркова

Освітньо-професійна програма

Технології програмування та комп'ютерне моделювання

Спеціальність

F1 – Прикладна математика

Галузь знань

F – Інформаційні технології

Рівень вищої освіти перший (бакалаврський)

Факультет математики та інформатики

Мова навчання українська

Чернівці 2026 рік

Робоча програма навчальної дисципліни
«РОЗРОБКА ІНТЕРАКТИВНИХ ДОДАТКІВ НА PYTHON» **складена відповідно до** освітньо-професійної програми «Технології програмування та комп'ютерне моделювання».

Розробник:

Ковдриш Володимир Володимирович, асистент, к.ф.-м.н.

Викладач (чі), що забезпечує читання даної навчальної дисципліни:

Ковдриш Володимир Володимирович, асистент, к.ф.-м.н.

Затверджено на засіданні кафедри прикладної математики та інформаційних технологій

Протокол № ___ від «___» _____ 20__ року

Завідувач кафедри _____ **Ярослав БІГУН**

Схвалено методичною радою факультету

Протокол № ___ від «___» _____ 202__ року

Голова методичної ради _____ **Тоня ФРАТАВЧАН**

Мета навчальної дисципліни: Сформувати практичні вміння створювати, перевіряти й пояснювати інтерактивні застосунки на Python, відокремлювати операції над даними від інтерфейсу та повторно використовувати спільну логіку в CLI, GUI і WEB.

Опис навчальної дисципліни

Загальна інформація

Форма навчання	Рік підготовки	Семестр	**Кількість**		**Кількість годин**						Вид підсумкового контролю
			кредитів	годин	лекції	практичні	семінарські	лабораторні	самостійна робота	індивідуальні завдання	
Денна	2	4	3	90	16	-	-	30	44	-	залік
Заочна	-	-	-	-	-	-	-	-	-	-	-

Структура змісту навчальної дисципліни

Назви змістових модулів і тем навчальних занять	**Кількість годин**											
	денна форма						заочна форма					
	усього	у тому числі					усього	у тому числі				
		лекц.	**практ.**	**лаб.**	**інд.**	**срс**		**лекц.**	**практ.**	**лаб.**	**інд.**	**срс**
1	2	3	4	5	6	7	8	9	10	11	12	13
Змістовий модуль 1. (CLI — ІНТЕРФЕЙС КОМАНДНОГО РЯДКА)												
Тема 1. Вступ до інтерактивних програм. Діалоговий цикл і список справ.	12	2		4		6						
Тема 2. Файли, функції, модулі та редагування даних.	12	2		4		6						
Тема 3. Завершення справ, обробники й диспетчер команд.	12	2		4		6						
Тема 4. Аргументи запуску та спільне ядро застосунку.	12	2		4		6						
Тема 5. Службові команди, декоратори, Click і Typer.	12	2		4		6						
Разом за ЗМ 1	**60**	**10**		**20**		**30**						
Змістовий модуль 2. (GUI — ГРАФІЧНИЙ ІНТЕРФЕЙС КОРИСТУВАЧА)												

Продовження таблиці «Структура змісту навчальної дисципліни»

Назви змістових модулів і тем навчальних занять	**Кількість годин**											
	денна форма						заочна форма					
	усього	у тому числі					усього	у тому числі				
		лекц.	**практ.**	**лаб.**	**інд.**	**срс**		**лекц.**	**практ.**	**лаб.**	**інд.**	**срс**
1	2	3	4	5	6	7	8	9	10	11	12	13
Тема 1. Графічний інтерфейс FreeSimpleGUI: події, список і дії над справами.	12	2		4		6						
Разом за ЗМ 2	**12**	**2**		**4**		**6**						
Змістовий модуль 3. (WEB — ВЕБІНТЕРФЕЙС ЗАСТОСУНКУ)												
Тема 1. Вебінтерфейс Streamlit: форми, стан сеансу та спільна логіка.	18	4		6		8						
Разом за ЗМ 3	**18**	**4**		**6**		**8**						
Усього годин	**90**	**16**		**30**		**44**						

Тематика лекційних занять з переліком питань

№	**Назва теми з основними питаннями**
1	Вступ до інтерактивних програм. Діалоговий цикл і список справ. — 1) Python, інтерпретатор, віртуальне середовище та інтерактивна консоль; поняття REPL; 2) Списки й рядки; input, strip, split, if/elif, while, break і continue; 3) Команди add/show/exit, нумерація через enumerate та захист прямого запуску __name__
2	Файли, функції, модулі та редагування даних. — 1) Функції, обов’язкові та необов’язкові параметри; filepath=FILEPATH, return і docstring; 2) Модуль, пакет і бібліотека; імпорт, help та запуск модуля окремо; 3) Читання і запис UTF-8 через with open, підготовка data/, редагування за номером, перевірки та виключення локальних даних із Git
3	Завершення справ, обробники й диспетчер команд. — 1) Команда complete: перетворення номера на індекс, видалення й збереження; 2) Функції-обробники зі спільним способом виклику; словник COMMANDS; 3) Диспетчер, невідомі команди, ValueError та IndexError; True/False як керування діалогом
4	Аргументи запуску та спільне ядро застосунку. — 1) sys.argv, argparse, підкоманди, обов’язкові аргументи, nargs, type=int та автоматична довідка; 2) run_from_args і main: вибір режиму, роль return після діалогового циклу; 3) Відокремлення операцій у core.py; повторне використання функцій читання і запису
5	Службові команди, декоратори, Click і Typer. — 1) count/clear/backup: список у пам’яті, файлове збереження та резервна копія; 2) Декоратор без @ і з @; вкладена функція, замикання, *args/**kwargs, __name__ та time.perf_counter; 3) Click: групи й аргументи; Typer: команди та анотації типів; порівняння запусків через три парсери
6	Графічний інтерфейс FreeSimpleGUI: події, список і дії над справами. — 1) GUI, FreeSimpleGUI, Tkinter, layout, key, Window.read, event і values; 2) Підготовка сховища, Listbox та оновлення елементів після зміни даних; 3) Додавання, вибір за індексом, редагування і завершення справ через core; порожній вибір, однакові тексти й коректне закриття
7	Вебінтерфейс Streamlit: форми, стан сеансу та спільна логіка. — 1) Streamlit, локальний вебсервер, браузер і повторне виконання скрипту; 2) Показ списку, поля, кнопки та форми; валідація введення й повідомлення користувачу; 3) Стан сеансу, оновлення після дії та повторне використання core/functions; порівняння CLI, GUI і WEB та межі локального файлового сховища

Тематика лабораторних занять з переліком питань

№	Назва теми (завдання)
1	Діалоговий список справ (Реалізувати add/show/exit, коректно розбирати команду з текстом із кількох слів, показувати нумерований список і перевіряти порожнє введення.)
2	Файлове збереження й редагування (Винести читання та запис у functions.py; зберігати справи між запусками, редагувати за номером, зберегти опис функцій і продемонструвати інший шлях до файла.)
3	Таблиця команд і диспетчер (Додати завершення справ, відокремити обробники й диспетчер; перевірити невідому команду, нечисловий номер, вихід за межі списку та повернення до діалогу.)
4	Аргументи запуску та спільне ядро (Описати команди argparse і довідку; забезпечити діалог без аргументів і одноразову дію з аргументами. Виділити операції над даними в core.py, зберігши поведінку.)
5	Click, Typer і службові дії (Додати count/clear/backup та два альтернативні способи опису CLI. Окремо продемонструвати декоратор вимірювання часу з позиційними й іменованими аргументами; перевірити спільні дані між способами запуску.)
6	Вікно для списку справ (Побудувати GUI із полем, списком і кнопками додавання, редагування та завершення. Повторно використати core, перевірити порожній вибір, дві однакові справи, закриття вікна та перезапуск.)
7	Вебінтерфейс і представлення проєкту (Створити локальний вебінтерфейс Streamlit для перегляду, додавання, редагування й завершення справ. Пояснити повторне виконання та стан сеансу, показати спільне збереження й підготувати інструкцію запуску всіх інтерфейсів.)

Індивідуальні науково-дослідні завдання (ІНДЗ)

№	Завдання до тем
1	Власний командний застосунок — Адаптувати список справ до обраної предметної області без зміни основного маршруту навчання. Подати код, інструкцію й сценарії коректного та помилкового введення; пояснити власні рішення. Виконується в межах самостійної роботи.
2	Три інтерфейси до спільних даних — Завершити наскрізний проєкт: спільне ядро, CLI, GUI та WEB. Подати інструкцію запуску, опис залежностей, навчальні дані й результати перевірок. Пояснити межі файлового сховища. Підсумковий захист оцінюється в межах 40 балів, окремі години не додаються.

Завдання для самостійної роботи студентів

№	Назва теми	Завдання для самостійної роботи	К-сть год.
1	Вступ до інтерактивних програм. Діалоговий цикл і список справ.	Відтворити діалог у власному середовищі; дослідити порожнє введення, команди з пробілами та завершення циклу. Пояснити різницю між файлом програми й інтерактивною консоллю.	6
2	Файли, функції, модулі та редагування даних.	Перевірити читання й запис українського тексту, значення за замовчуванням, імпорт без запуску діалогу та збереження після перезапуску; пояснити відмінність модуля, пакета й бібліотеки.	6

№	Назва теми	Завдання для самостійної роботи	К-сть год.
3	Завершення справ, обробники й диспетчер команд.	Підготувати сценарії перевірки диспетчера й помилок номера; простежити шлях команди до обробника та пояснити призначення поверненого логічного значення.	6
4	Аргументи запуску та спільне ядро застосунку.	Перевірити довідку та помилки argparse; простежити керування між main, repl і run_from_args. Порівняти поведінку до і після виділення core.	6
5	Службові команди, декоратори, Click і Typer.	Порівняти argparse, Click і Typer на однакових діях; дослідити декоратор із *args/**kwargs, обов'язковість параметрів, точність показу часу й відновлюваність даних із резервної копії.	6
6	Графічний інтерфейс FreeSimpleGUI: події, список і дії над справами.	Перевірити послідовність графічних подій, оновлення списку та роботу з дублями; зафіксувати результати ручного тестування і пояснити, чому GUI не викликає термінальні обробники.	6
7	Вебінтерфейс Streamlit: форми, стан сеансу та спільна логіка.	Дослідити повторне виконання Streamlit, ключі елементів і стан сеансу; перевірити помилкове введення. Оформити README, залежності та результати перевірок індивідуального проєкту.	8

Разом за наведеними завданнями: 44 год.

Завдання навчальної дисципліни

Організувати віртуальне середовище, запуск Python-програми та роботу з інтерактивною консоллю.

Побудувати діалоговий цикл команд, перевірку введення, нумерацію та основні дії зі списком справ.

Реалізувати файлове збереження, функції, модулі й обробку помилок; зберігати навчальну історію в Git.

Описувати команди запуску через argparse, Click і Typer; пояснювати декоратори та анотації типів.

Виділити спільні операції в core.py та використати їх у графічному інтерфейсі FreeSimpleGUI.

Створити вебінтерфейс на Streamlit із введенням, показом та зміною справ; перевірити узгодженість даних між інтерфейсами.

Очікувані результати навчання

Запускати програму в активованому віртуальному середовищі, встановлювати потрібні пакети й відтворювати запуск за інструкцією.

Пояснювати роботу списків, рядків, умов, циклів, функцій, параметрів і повернених значень на прикладі інтерактивної програми.

Читати та записувати дані через with open, застосовувати необов’язкові параметри, відрізняти імпорт модуля від прямого запуску.

Виконувати додавання, перегляд, редагування та завершення справ із перевіркою порожнього введення й недопустимих номерів.

Організовувати таблицю команд і диспетчер; відокремлювати введення та виведення від операцій над даними.

Створювати довідку й розбір аргументів через argparse, Click і Typer; пояснювати передавання *args/**kwargs та вимірювання часу в декораторі.

Реалізовувати підрахунок, очищення та резервне копіювання списку справ.

Побудувати графічне вікно зі списком, полем і кнопками; обробляти події й коректно працювати з вибраними позиціями, зокрема однаковими текстами.

Побудувати навчальну вебсторінку Streamlit із формою, діями над справами та поясненням повторного виконання й стану сеансу.

Перевіряти збереження даних, помилкові сценарії та повторний запуск; пояснювати межі файлового сховища при одночасній роботі кількох процесів.

Методи навчання

Лекції з поетапною демонстрацією коду та поведінки програми, пояснення термінів при першій появі, лабораторні роботи, інтерактивні досліди, контрольні питання, розбір помилок і наскрізний індивідуальний проєкт.

Система контролю та оцінювання

Методи контролю

1. перевірка вихідного Python-коду та структури модулів
2. демонстрація команд, вікна або вебсторінки на навчальних даних
3. перевірка помилкового введення, повторного запуску й файлового збереження
4. усне пояснення функцій, декораторів, подій та потоку даних
5. захист індивідуального застосунку з інструкцією запуску

Форми контролю

1. поточний контроль: лабораторні завдання, перевірки й пояснення рішень — до 60 балів
2. підсумковий контроль — залік; захист інтерактивного застосунку — до 40 балів

Критерії оцінювання поточного та підсумкового контролю

Критерії підсумкового захисту: правильність основних дій і збереження — до 14; узгодженість трьох інтерфейсів та спільної логіки — до 10; відтворюваний запуск, структура й перевірки — до 8; самостійне пояснення та відповіді — до 8.

Лабораторні та індивідуальні завдання є доказами досягнення результатів відповідних тем; їх бали не додаються повторно понад 100. Індивідуальні

завдання виконуються в межах самостійної роботи.

Розподіл балів, які отримують студенти

Поточне оцінювання (аудиторна та самостійна робота)							Кількість балів (залікова/екзаменаційна робота)	Сумарна к-ть балів
Змістовий модуль № 1					**Мод. № 2**	**Мод. № 3**		
Т1	Т2	Т3	Т4	Т5	Т1	Т1	40	100
4	4	4	4	4	20	20		

Шкала оцінювання: національна та ЄКТС

Оцінка за національною шкалою	**Оцінка за шкалою ЄКТС**	
	Оцінка (бали)	**Пояснення за розширеною шкалою**
зараховано	A (90-100)	відмінно
	B (80-89)	дуже добре
	C (70-79)	добре
	D (60-69)	задовільно
	E (50-59)	достатньо
не зараховано	FX (35-49)	(незадовільно) з можливістю повторного складання
	F (1-34)	(незадовільно) з обов’язковим самостійним опрацюванням освітнього компоненту до перескладання

Перелік питань для самоконтролю та підсумкового контролю

Питання для підсумкового контролю

1. Що таке інтерпретатор, віртуальне середовище та інтерактивна консоль Python?
2. Як розшифровується REPL і де, крім Python, трапляється такий цикл?
3. Як strip і split(maxsplit=1) допомагають відокремити команду від тексту?
4. Чим відрізняються break, continue і return?
5. Чому номери справ від 1 відрізняються від індексів списку від 0?
6. Що таке функція, параметр, аргумент і значення, що повертається?
7. Коли параметр filepath=FILEPATH можна не передавати?
8. Чим відрізняються модуль, пакет і бібліотека?
9. Що означає if __name__ == ”__main__”і чим імпорт відрізняється від прямого запуску?
10. Навіщо використовують docstring і як переглянути його через help?
11. Як with open керує закриттям файла і чим небезпечний режим w без перевірки?
12. Як відрізняються дані в пам’яті від збережених даних у файлі?
13. Як обробляти порожній текст, нечисловий номер і номер поза списком?
14. Як працюють таблиця COMMANDS, диспетчер та обробники зі спільною сигнатурою?
15. Що означає True/False, яке повертають наші термінальні обробники?
16. Які дані містить sys.argv і навіщо передавати явний список аргументів при перевірці парсера?

17. Як argparse описує підкоманди, nargs, обов’язковість і тип аргументу?
18. Чому main має повернути керування після завершення діалогового режиму?
19. Які операції належать до core, а які мають залишатися в інтерфейсі?
20. Чому todos.clear() і присвоєння todos = [] можуть по-різному впливати на спільний список?
21. Що зберігає резервна копія і як перевірити її після очищення робочого файла?
22. Що таке декоратор і як запис із @ подати без символу @?
23. Коли виконується декоратор, а коли — тіло його обгортки?
24. Що означають *args і **kwargs при визначенні функції та при її виклику?
25. Чому декоратор має повертати результат початкової функції?
26. Що містить func.__name__ і як декорування впливає на ім’я обгортки?
27. Для чого потрібен time.perf_counter і що означає формат :.6f?
28. Як у Click реєструються група, команди та аргументи?
29. Як Typer використовує анотації типів і чи перевіряє їх звичайний Python самостійно?
30. Чим відрізняються повідомлення про помилку й код завершення процесу?
31. Що таке GUI, елемент інтерфейсу, layout і key?
32. Що повертає window.read і чим event відрізняється від values?
33. Чому закриття вікна перевіряють до читання значень елементів?
34. Навіщо оновлювати список у вікні після зміни даних через core?
35. Чому для однакових текстів справ потрібно працювати з позицією вибраного рядка?
36. Як обробляти порожній вибір і переводити індекс GUI у номер для core?
37. Чим локальний вебсервер відрізняється від браузера?
38. Як Streamlit повторно виконує скрипт після взаємодії з елементом?
39. Для чого потрібні форма, ключі елементів і стан сеансу?
40. Чим стан сеансу відрізняється від даних, збережених у файлі?
41. Як використати core і functions у вебінтерфейсі без дублювання операцій?
42. Які проблеми можливі при одночасному записі кількома процесами в той самий файл?
43. Які файли, залежності та інструкції потрібні для запуску проєкту на іншому комп’ютері?

Зарахування результатів неформальної освіти

Можливість зарахування результатів неформальної освіти визначається чинними правилами університету.

Для розгляду можуть подаватися сертифікати та самостійні Python-проєкти з вихідним кодом, демонстрацією й інструкцією запуску.

Рекомендована література

Основна

1. Python — офіційний навчальний посібник
2. FreeSimpleGUI — офіційна документація

Додаткова

1. Click — офіційна документація
2. Typer — офіційна документація
3. Streamlit — модель виконання, стан сеансу й форми

Інформаційні ресурси

1. Розробка інтерактивних додатків на Python

Політика академічної доброчесності

Дотримання політики академічної доброчесності регламентовано чинними документами університету.

- Кодекс академічної доброчесності та етики ЧНУ
- Положення про виявлення та запобігання академічному плагіату ЧНУ
- Політика використання штучного інтелекту в ЧНУ

Використання здобувачами освіти інструментів штучного інтелекту (ШІ)

1. Дозволяється за умови дотримання академічної доброчесності, позначення факту та способу використання ШІ і відповідальності здобувача за результат.
2. Не допускається використання ШІ як заміни самостійного виконання завдань або для порушення академічної доброчесності.
3. Викладач може визначати додаткові вимоги до використання ШІ для конкретної дисципліни.