МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЧЕРКАСЬКИЙ ДЕРЖАВНИЙ ТЕХНОЛОГІЧНИЙ УНІВЕРСИТЕТ
ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ І СИСТЕМ

МЕТОДИЧНІ РЕКОМЕНДАЦІЇ

до виконання лабораторних робіт з дисципліни
«Алгоритмізація та програмування»
для здобувачів освітнього ступеня «бакалавр»
галузі знань F «Інформаційні технології»
спеціальності F3 «Комп'ютерні науки»
освітньої програми «Комп'ютерні науки та прикладне програмування»
усіх форм навчання

Черкаси
2026

УДК 004.42 (07)
М54

Затверджено вченою радою ФІТІС, протокол №9 від 26.02.2026 р., згідно з рішенням кафедри комп'ютерних наук та системного аналізу, протокол №9 від 05.01.2026 р.

Упорядник: Максимов А.Є., *PhD з комп'ютерних наук, викладач кафедри комп'ютерних наук та системного аналізу*

Рецензент: Заспа Г.О., *кандидат технічних наук, доцент, доцент кафедри програмного забезпечення автоматизованих систем*

М54 Методичні рекомендації до виконання лабораторних робіт з дисципліни «Алгоритмізація та програмування» для здобувачів освітнього ступеня «бакалавр» галузі знань F «Інформаційні технології», спеціальності F3 «Комп'ютерні науки», освітньої програми «Комп'ютерні науки та прикладне програмування» усіх форм навчання [Електронний ресурс] / [упоряд. Максимов А.Є.]; М-во освіти і науки України, Черкас. держ. технол. ун-т. Черкаси: ЧДТУ, 2026. 61 с.

Методичні рекомендації спрямовані на формування у здобувачів освітнього ступеня «бакалавр» галузі знань F «Інформаційні технології», спеціальності F3 «Комп'ютерні науки», освітньої програми «Комп'ютерні науки та прикладне програмування» базових знань і практичних навичок з побудови алгоритмів, використання структур даних, розроблення програмного забезпечення та відлагодження програм, а також сприяє розвитку логічного та алгоритмічного мислення у сфері комп'ютерних наук.

УДК 004.42 (07)

Виробничо-практичне
електронне видання
комбінованого використовування

МЕТОДИЧНІ РЕКОМЕНДАЦІЇ

до виконання лабораторних робіт з дисципліни
«Алгоритмізація та програмування»
для здобувачів освітнього ступеня «бакалавр»
галузі знань F «Інформаційні технології»
спеціальності F3 «Комп'ютерні науки»
освітньої програми «Комп'ютерні науки та прикладне програмування»
усіх форм навчання

Упорядники: **Максимов** Антон Євгенійович
В авторській редакції

ВСТУП

Формування та розвиток практичних умінь студентів, які здобувають освітній рівень бакалавра у процесі вивчення дисципліни «Алгоритмізація та програмування», передбачає оволодіння знаннями з побудови алгоритмів, планування та реалізації програмних рішень, набуття навичок у розв'язанні прикладних та навчально-наукових задач. Для цього необхідна система знань щодо основних понятійних та структурних елементів алгоритмізації й програмування. Підґрунтям виступає розуміння методології створення алгоритмів, принципів роботи з базовими структурами даних, знання сфер застосування сучасних мов програмування, їхніх переваг та обмежень.

Мета викладання навчальної дисципліни «Алгоритмізація та програмування» полягає в ознайомленні здобувачів освітнього ступеня бакалавр з основними поняттями алгоритмізації і техніками застосування у програмуванні базових алгоритмічних структур (організація програм) і базових структур даних (організація даних), формування у здобувачів теоретичної бази та практичних навичок з основ програмування для вирішення прикладних задач із різних предметних областей.

Основними завданнями вивчення дисципліни «Алгоритмізація та програмування» є формування у здобувачів освітнього ступеня бакалавра базової підготовки з алгоритмізації і програмування, забезпечення розуміння підходів до аналізу, проєктування, реалізації, тестування та документування програмного забезпечення, а також розвиток умінь використовувати сучасні засоби розробки програмного забезпечення.

Мета лабораторних робіт полягає у забезпеченні розуміння та засвоєння здобувачами освітнього рівня бакалавр змісту дисципліни «Алгоритмізація та програмування» шляхом свідомого закріплення, поглиблення й систематизації теоретичних знань, а також набуття практичних навичок розробки, налагодження та тестування алгоритмів і програмних рішень для розв'язання прикладних задач у сфері комп'ютерних наук.

Методичні рекомендації до лабораторних робіт здобувачів освітнього рівня бакалавр з навчальної дисципліни «Алгоритмізація та програмування» містять: вступ; дванадцять лабораторних робіт, які охоплюють основні теми курсу; список використаної літератури; додатки. Під час виконання робіт здобувачі вивчають матеріали окремих тем шляхом опрацювання відповідної літератури, здійснюють підготовку до лабораторних робіт, виконують лабораторні завдання та поточного контролю знань.

Цільовим призначенням видання є забезпечення студентів спеціальності F3 «Комп'ютерні науки» навчально-методичною підтримкою у вивченні дисципліни «Алгоритмізація та програмування», здобуття ними базових знань і практичних навичок з побудови алгоритмів, використання структур даних, розроблення програмного забезпечення та відлагодження програм, а також сприяння розвитку логічного та алгоритмічного мислення у сфері комп'ютерних наук.

Лабораторна робота (ЛР) вважається виконаною за умови повного й коректного виконання всіх завдань, передбачених методичними рекомендаціями; подання коректного програмного рішення без критичних помилок; обґрунтованого

використання алгоритмів, мовних конструкцій, інструментів і середовища розробки; належного оформлення роботи, коду та результатів роботи (див. Додаток А); здійснення аналізу й відлагодження програми, а також дотримання принципів академічної доброчесності. Максимальна оцінка за виконану ЛР складає 5 балів. Бали розподіляються наступним чином: за виконання загальних (аудиторних) завдань до ЛР – 2 бали, за виконання індивідуальних завдань до ЛР – 3 бали.

ЛАБОРАТОРНА РОБОТА №1

Тема: «Встановлення середовища програмування Python. Синтаксис Python. Основні принципи та правила створення, компіляції та відлагодження програм»

Мета роботи: Ознайомитись із середовищем програмування Python, процесом його встановлення та налаштування. Сформувати базові навички побудови алгоритмів для розв'язання типових задач комп'ютерних наук, а також засвоїти основи структурного та алгоритмічного мислення.

Завдання до лабораторної роботи

1. Вивчити теоретичний матеріал лекції №1.
2. Долучитись до курсу «Python Essentials 1 by Cisco» за посиланням: https://www.netacad.com/courses/python-essentials-1. Пройти в цьому курсі розділ «PE1: Модуль 1. Вступ до мови Python та комп'ютерного програмування».
3. Встановити Python на комп'ютер та середовище розробки VS Code.
4. Виконати створення змінних:
   - створіть змінну name і присвойте їй своє ім'я;
   - створіть змінні age та height і присвойте їм відповідно свій вік та зріст;
   - створіть змінну з недопустимою назвою (наприклад, 1name) і зафіксуйте, яку помилку отримаєте.
5. Створення кількох змінних одночасно:
   - призначте трьом змінним x, y, z значення 5, 10, 15 в одному рядку коду;
   - виведіть значення всіх трьох змінних на екран у форматі: x=5, y=10, z=15.
6. Виведення змінних:
   - створіть змінну city і призначте їй назву вашого міста через input();
   - виведіть її за допомогою print();
   - виведіть декілька змінних в одному рядку з текстовим поясненням, наприклад: Моє ім'я: <name>, Місто: <city>.
7. Глобальні та локальні змінні:
   - створіть глобальну змінну counter і надайте їй значення 0;
   - скопіюйте приклад з лекції з функцією, яка збільшує counter на 1 і виводить його значення;
   - викликайте функцію кілька разів і спостерігайте, як змінюється глобальна змінна.
8. Використання коментарів:
   - додайте однорядковий коментар до кожної змінної із завдання 4 з поясненням її призначення;
   - створіть багаторядковий коментар, який описує програму із завдання 6.
9. Проведіть відлагодження програми з лекційного матеріалу (Debugging).
   - додайте breakpoints у ключових місцях коду;
   - переконайтеся, що виконання зупиняється на встановлених точках зупину;
   - перевірте значення змінних у вікні «Variables»;

- перевірте стек викликів у вікні «Call Stack»;
- використайте Step Over (F10), Step Into (F11) та Step Out (Shift+F11), щоб послідовно перевірити логіку роботи коду;
- під час паузи виконання спробуйте ввести вирази у консолі дебагу і перевірте значення змінних;
- опишіть, які помилки або логічні нюанси були виявлені під час відлагодження програми та як їх вдалося усунути.

Індивідуальні завдання до лабораторної роботи

1. Вивести на екран Ваше ПІБ та групу, в якій навчаєтесь.
2. Написати програму, яка запитує у користувача його ПІБ, рік народження, групу, в якій навчається студент. Після введення інформації програма виводить на екран цю інформацію. Використати метод format().

Зміст звіту

Оформити звіт з лабораторної роботи за такою структурою (Додаток А):

1. Титульний аркуш;
2. Тема і мета роботи;
3. Протокол виконання завдань;
4. Висновки.

Контрольні питання

1. Що таке Python і для чого його використовують?
2. Які основні компоненти середовища програмування Python?
3. Які особливості Python відрізняють його від інших мов програмування?
4. Які кроки необхідно виконати для встановлення Python на комп’ютер?
5. Які способи запуску Python-програм існують (інтерактивний режим, файл .py)?
6. Що таке інтерпретатор Python і яка його роль у виконанні програм?
7. Як перевірити, що Python встановлено правильно?
8. Що таке IDE і які IDE можна використовувати для Python?
9. Чим відрізняється робота в інтерактивному режимі (REPL) від роботи з файлом скрипта?
10. Що таке алгоритм? Наведіть приклад алгоритму повсякденного завдання.
11. Які етапи побудови алгоритму ви знаєте?
12. Що таке синтаксис Python і чим він відрізняється від інших мов програмування?
13. Як у Python організовується відступ коду і чому він важливий?
14. Що таке змінна у Python і як вона створюється?
15. Які основні типи даних існують у Python? Наведіть приклади.
16. Які є правила іменування змінних у Python?
17. Що таке коментар у Python і як його використовувати?

18. Як працює функція print() і які способи форматування рядків у Python ви знаєте?
19. Чим відрізняються глобальні та локальні змінні, і як використовується ключове слово global?
20. Що таке компіляція та відлагодження Python-програм і які інструменти можна для цього застосовувати?
21. Які основні принципи побудови алгоритмів?
22. Як у Python обробляються помилки під час виконання програм?
23. Поясніть, як вводяться дані від користувача і які функції для цього використовуються у Python 3.x та Python 2.x.
24. Як створити та виконати просту програму на Python, яка виводить повідомлення на екран?
25. Як зчитувати дані з клавіатури та виводити результати?
26. Як правильно розбити задачу на послідовні кроки для програмування?
27. У чому полягає важливість перевірки алгоритму перед реалізацією на Python?

ЛАБОРАТОРНА РОБОТА №2
Тема: «Типи даних і операції з ними. Арифметичні та логічні оператори. Стиль кодування»

Мета роботи: Розглянути основні типи даних у Python, ознайомитись з особливостями використання арифметичних та логічних операторів, навчитися застосовувати їх у розв'язанні практичних задач. Розвинути навички дотримання правил стилю програмування для підвищення читабельності та якості коду.

Завдання до лабораторної роботи

Завдання 1.

1.1. Вивчити теоретичний матеріал лекції №1.

1.2. З курсу «Python Essentials 1 by Cisco» (https://www.netacad.com/courses/python-essentials-1) пройти «PE1: Модуль 2. Типи даних Python, змінні, оператори та основні операції введення-виведення».

1.3. Виконайте завдання з числами:
- створіть змінні наступних типів: int, float, complex;
- виконайте конверсію між типами (int(), float(), complex());
- виконайте прості арифметичні операції між числами (+, -, *, /, %, **, //);
- використовуючи оператори присвоєння, змініть значення змінних (+=, -=, *=, /=, %=).

1.4. Виконайте завдання з рядками:
- створіть рядки за допомогою одинарних, подвійних та трьох лапок;
- використовуйте слайсинг для виділення частини рядка;
- застосуйте методи рядків: upper(), lower(), strip(), replace(), split();
- виконайте конкатенацію рядків та використовуйте форматування з format();
- використайте символи екранування для вставки лапок та спеціальних символів.

1.5. Виконайте завдання з булевими значеннями:
- створіть булеві змінні True та False;
- виконайте логічні операції and, or, not;
- оцініть різні значення за допомогою функції bool();
- напишіть програму, яка повертає булеве значення, і виконайте умовну перевірку за її результатом.

1.6. Виконайте завдання із використанням операторів:
- використайте оператори порівняння (==, !=, >, <, >=, <=);
- виконайте логічні оператори (and, or, not);
- продемонструйте роботу операторів приналежності (in, not in) та ідентифікації (is, is not);
- використайте порозрядні (побітові) оператори (&, |, ^, ~, <<, >>);
- напишіть вирази, що демонструють пріоритет операторів.

Завдання 2. Написати програму відповідно до поставленого завдання.

1. Написати програму, яка приймає два числа, обчислює їх суму, різницю, добуток і частку. Вивести результати, дотримуючись правил форматування коду PEP 8.

2. Написати програму, яка приймає число і визначає, чи воно додатнє, від'ємне або нуль, використовуючи логічні оператори.

3. Написати програму, яка обчислює середнє арифметичне трьох чисел і перевіряє, чи середнє більше 10, використовуючи арифметичні та логічні оператори.

4. Написати програму, яка конвертує введене користувачем число у різні типи даних (int, float, str, bool) та виводить результати.

5. Написати програму, яка обчислює залишок від ділення двох чисел та перевіряє, чи він дорівнює нулю, використовуючи логічний оператор.

6. Написати програму, яка перевіряє, чи обидва введені числа парні, використовуючи логічні оператори.

7. Написати програму, яка приймає число і обчислює його квадрат і куб, дотримуючись правил оформлення змінних і вирівнювання коду.

8. Написати програму, яка приймає три числа і визначає, чи принаймні одне з них більше 0, використовуючи логічні оператори.

9. Написати програму, яка обчислює вираз (a + b) * c / d для введених чисел і перевіряє, чи результат менший за 100.

10. Написати програму, яка перевіряє, чи число одночасно додатнє і кратне 5, використовуючи логічні та арифметичні оператори, і виводить результат у зрозумілому форматі.

Індивідуальні завдання до лабораторної роботи

Завдання 1. Написати програму відповідно до номера варіанту.

Варіанти:

1-3. Написати програму, яка приймає від користувача логічне значення (True/False) і виводить його як int, str і float.

4-6. Написати програму, яка приймає два числа і визначає, який тип даних у кожного числа після операції ділення / та ділення націло //.

7-9. Написати програму, яка приймає рядок і виводить його довжину, перший символ та останній символ.

10-12. Написати програму, яка приймає від користувача рядок і виводить його довжину (int), перший символ (str) і логічне значення (bool) – чи рядок не порожній.

13-15. Написати програму, яка приймає число і перевіряє, чи воно ціле (int) або дійсне (float), використовуючи функцію type().

Завдання 2. Написати програму відповідно до номера варіанту.

Варіанти:

1-3. Написати програму, яка приймає два числа і перевіряє, чи обидва вони додатні. Використати логічний оператор and.

4-6. Створити програму, яка приймає число і перевіряє, чи воно від'ємне або дорівнює нулю. Використати логічний оператор or.

7-9. Написати програму, яка приймає логічне значення is_student і число age, і перевіряє, чи особа є студентом і старше 18 років.

10-12. Написати програму, яка приймає три числа і перевіряє, чи принаймні одне з них додатнє. Використати логічний оператор or.

13-15. Написати програму, яка приймає число і перевіряє, чи воно не дорівнює нулю, використовуючи логічне заперечення not.

Завдання 3. Написати програму відповідно до номера варіанту.

Варіанти:

1-3. Написати програму для обчислення площі прямокутника. Використовувати зрозумілі назви змінних і правильні відступи, додати docstring і коментарі для пояснення логіки програми.

4-6. Написати програму для обчислення периметра трикутника. Параметри – довжини сторін. Додати docstring, коментарі та дотримуватись PEP 8.

7-9. Написати програму, яка перевіряє, чи є число парним. Використати зрозумілі назви змінних, коментарі та docstring для пояснення логіки.

10-12. Написати програму для обчислення середнього арифметичного та геометричного трьох чисел. Додати docstring, коментарі, дотримуватись правил PEP 8 (відступи, пробіли, зрозумілі назви).

13-15. Написати програму для конвертації температури з Цельсія у Фаренгейти. Додати docstring, пояснити параметри і результат, використати зрозумілі назви змінних та стиль кодування PEP 8.

Завдання 4. Написати програму відповідно до номера варіанту.

1-3. Написати програму, яка приймає від користувача рядок і виводить його довжину, перший і останній символ, а також рядок у верхньому та нижньому регістрі.

4-6. Написати програму, яка приймає два рядки і перевіряє, чи перший рядок міститься у другому. Вивести результат у вигляді True/False.

7-9. Написати програму, яка приймає рядок і заміщує всі пробіли на підкреслення (_). Вивести результат.

10-12. Написати програму, яка приймає рядок і виводить кількість голосних та приголосних літер у ньому.

13-15. Написати програму, яка приймає рядок і виводить рядок у зворотному порядку, а також перевіряє, чи є він паліндромом (читається однаково зліва направо і справа наліво).

Контрольні питання

1. Які основні типи даних використовуються в Python? Наведіть приклади.
2. Чим відрізняється ціле число (int) від дійсного числа (float)?
3. Що таке булевий тип (bool) і які значення він може приймати?
4. Як перетворити рядок у число, а число у рядок? Наведіть приклад.
5. Що робить функція type() в Python?
6. Назвіть основні арифметичні оператори в Python та їх призначення.
7. Чим відрізняються оператори / і //?
8. Як обчислити залишок від ділення двох чисел у Python?

9. Який оператор використовується для піднесення числа до степеня?
10. Напишіть приклад обчислення середнього арифметичного трьох чисел.
11. Назвіть логічні оператори в Python та поясніть їх роботу.
12. Що таке коротке замикання (short-circuit) у логічних операторах?
13. Напишіть приклад використання логічного оператора and.
14. Напишіть приклад використання логічного оператора or.
15. Як використовується оператор not і для чого він потрібен?
16. Які основні правила стилю кодування Python (PEP 8)?
17. Чому важливо використовувати зрозумілі імена змінних?
18. Як правильно робити відступи та пробіли у виразах і операторах?
19. Що таке docstring і для чого він використовується?
20. Як правильно структурувати коментарі та логічні блоки у Python-коді?

ЛАБОРАТОРНА РОБОТА №3
Тема: «Умовні конструкції if/elif/else»

Мета роботи: Ознайомитися з принципами роботи умовних конструкцій у Python, навчитися застосовувати оператори if, elif, else для реалізації розгалужень у програмах, що підвищить їх гнучкість і ефективність. Дослідити можливості використання оператора match/case для обробки множинних умов, навчитися комбінувати різні умови.

Завдання до лабораторної роботи

1. Напишіть програму, яка зчитує два числа та виводить, яке з них більше, використовуючи оператор if.
2. Напишіть програму, в якій спеціально пропустіть відступ у блоці if, та перевірте, яке повідомлення про помилку виведе Python.
3. Напишіть програму, яка зчитує два числа і за допомогою elif:
   - виводить «Перше більше», якщо перше число більше;
   - виводить «Друге більше», якщо друге число більше;
   - виводить «Числа рівні», якщо значення однакові.
4. Напишіть програму, яка визначає, чи є введене число додатним, від'ємним або нулем за допомогою else.
5. Використайте скорочений запис if та if ... else, щоб у одному рядку визначити, яке з двох чисел більше, і вивести відповідний результат.
6. Використайте логічні оператори (and, or, not):
   - перевірте, чи введене число потрапляє у діапазон від 10 до 50 (включно);
   - виведіть повідомлення, якщо число не належить цьому діапазону.
7. За допомогою вкладених перевірок напишіть програму, яка зчитує число x і перевіряє:
   - чи воно більше 0;
   - якщо так, то додатково перевіряє, чи більше воно 100.
8. Створіть умовну конструкцію if, яка перевіряє будь-яку умову, але не виконує жодних дій, використовуючи оператор pass.
9. Напишіть програму, яка зчитує з клавіатури слово, що описує настрій користувача, та виводить відповідь за допомогою конструкції match/case:
   - якщо введено happy, glad або great, вивести «Радий чути, що у вас гарний настрій!»;
   - якщо введено sad, tired або upset, вивести «Не засмучуйтесь, все буде добре!»;
   - якщо введено angry або mad, вивести «Спробуйте заспокоїтись і вдихнути глибше.»;
   - у будь-якому іншому випадку → вивести «Я поки що не знаю такого настрою.».
10. Напишіть програму, яка зчитує від користувача номер функції та обчислює значення y в точці x, використовуючи match/case:
    - $y = x^2$;
    - $y = \sqrt{x}$, якщо $x \geq 0$;

– $y = |x|$;
– $y = \sin(x)$;
– $y = \cos(x)$.

Індивідуальні завдання до лабораторної роботи

Завдання 1. Написати, відлагодити та протестувати розгалужену програму для обчислення значення функції $y = f(x, a, b)$ за допомогою умовного оператора при заданих значеннях змінної x та параметрів a і b. Значення параметрів a та b обираються так, щоб функція була визначена на відповідних проміжках.

Варіанти:

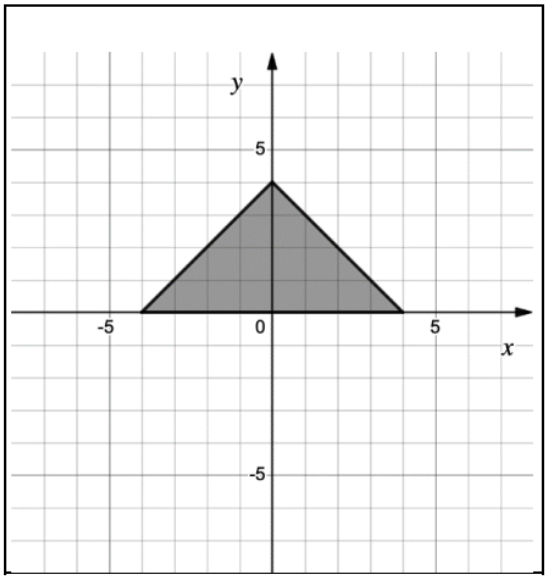
№	Функція	Змінні
1.	$y = \begin{cases} \dfrac{e^{x+4}}{\cos(6x)}, & x \le 0 \\ \log\sqrt{x}, & 0 < x < 3 \\ x^3, & x \ge 3 \end{cases}$	e – стала (2.718); x, y – змінні
2.	$y = \begin{cases} \sqrt[3]{x^2 + \lvert a\rvert}, & x \le 1 \\ \sin^2(ax) * \cos^2(bx), & 1 < x \le 3 \\ \lvert a + x + b\rvert, & x > 3 \end{cases}$	a, b – параметри; x, y – змінні
3.	$y = \begin{cases} e^{-x+6}, & x \le 0 \\ \sqrt[3]{a^2 + 1} + x, & 0 < x < 1 \\ b + \sqrt{x}, & x \ge 1 \end{cases}$	e – стала (2.718); a, b – параметри; x, y – змінні
4.	$y = \begin{cases} \sqrt[4]{\lvert a + bx\rvert}, & x < 1 \\ \ln^2(b^2 + x), & 1 \le x < 2 \\ e^{b+3}, & x \ge 2 \end{cases}$	e – стала (2.718); a, b – параметри; x, y – змінні
5.	$y = \begin{cases} \sin x + \sqrt[4]{\lvert a\rvert}, & x < 1 \\ \log\lvert b^2 + x\rvert, & 2 \le x < 4 \\ \cos^3(ax), & x \ge 4 \end{cases}$	a, b – параметри; x, y – змінні
6.	$y = \begin{cases} \sqrt[3]{a^2 + 3x^2}, & x < b \\ \ln^2\lvert x - b\rvert, & b \le x \le b + 1 \\ e^{ax - x^4}, & x > b + 1 \end{cases}$	e – стала (2.718); a, b – параметри; x, y – змінні
7.	$y = \begin{cases} \operatorname{arctg}(ax) + \sqrt[3]{\lvert x\rvert + 2}, & ax < 1 \\ \sin(bx) + 1.4, & 1 \le ax \le 2 \\ \ln^2(x^2 + \lvert ab\rvert), & ax > 2 \end{cases}$	a, b – параметри; x, y – змінні
8.	$y = \begin{cases} \ln^2 \dfrac{x^2 + \lvert a\rvert}{1 + x^4}, & x \le 3 \\ b^3 + \cos^2(x), & 3 < x \le 5 \\ \sqrt[3]{\lvert ab\rvert x}, & x > 5 \end{cases}$	a, b – параметри; x, y – змінні

9.	$y = \begin{cases} 2\sqrt[3]{\sin(ax+10)}, & x \le 4 \\ \ln^2(\lvert a+bx\rvert+1), & 4 < x < 5 \\ \dfrac{x^3}{a^2+b^2+2}, & x \ge 5 \end{cases}$	a, b – параметри; x, y – змінні
10.	$y = \begin{cases} \ln^3(\lvert ax+b\rvert+1), & ax < 1 \\ e^{-\sqrt{x^2+a^2}}\sin(bx), & 1 \le ax < 2 \\ \operatorname{arctg}(ax-b), & ax \ge 2 \end{cases}$	e – стала (2.718); a, b – параметри; x, y – змінні
11.	$y = \begin{cases} \sqrt{\lvert a+x\rvert}, & x \le -1 \\ \sin(ax)+\ln(x^2+1), & -1 < x < 2 \\ e^{b-x}, & x \ge 2 \end{cases}$	a, b – параметри; x, y – змінні
12.	$y = \begin{cases} \ln^2(\lvert x\rvert+a+1), & x < 0 \\ \sqrt[3]{x^2+b^2}, & 0 \le x < 3 \\ \cos(ax)+\sqrt{x}, & x \ge 3 \end{cases}$	a, b – параметри; x, y – змінні
13.	$y = \begin{cases} e^{x^2-a}, & x \le 1 \\ \operatorname{arctg}(bx)+x, & 1 < x < 2 \\ \ln(\lvert x-a\rvert+1), & x \ge 2 \end{cases}$	e – стала (2.718); a, b – параметри; x, y – змінні
14.	$y = \begin{cases} \sqrt[4]{\lvert ax+b\rvert}, & x < 1 \\ \sin^2(x)+\cos^2(bx), & 1 \le x < 4 \\ \dfrac{x^2}{a^2+b^2+1}, & x \ge 4 \end{cases}$	a, b – параметри; x, y – змінні
15.	$y = \begin{cases} \ln(\lvert x\rvert+1)+a, & x \le 0 \\ \sqrt{x+b}, & 0 < x < 2,\ b \ge 0 \\ e^{\sin(ax)}, & x \ge 2 \end{cases}$	e – стала (2.718); a, b – параметри; x, y – змінні

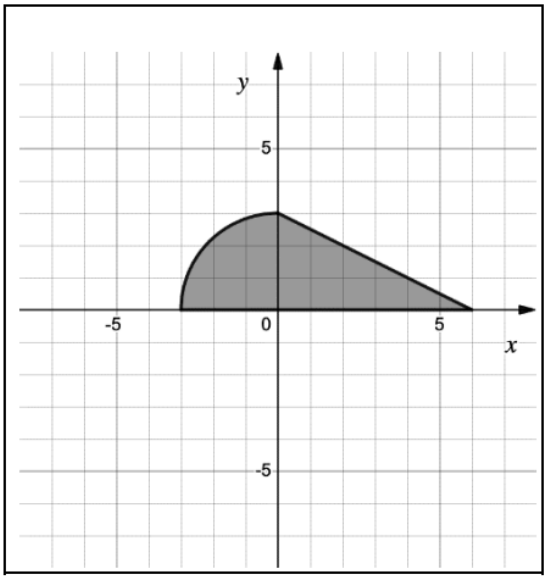
Завдання 2. Відповідно до варіанту напишіть програму, яка виводить True, якщо точка з координатами (x, y) належить зафарбованій області, і False – в іншому випадку.

Варіанти:

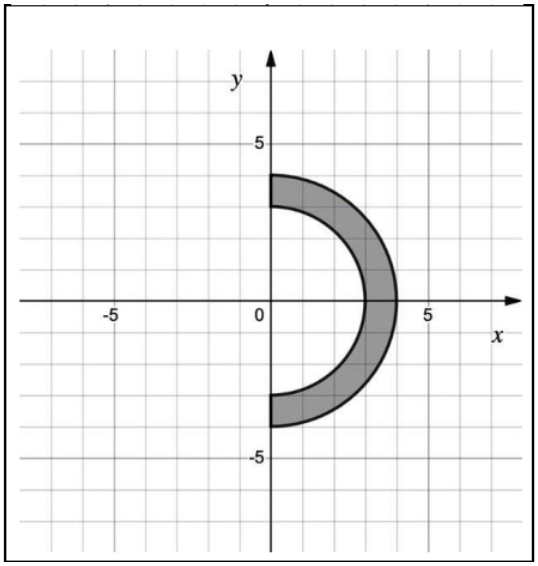
1.



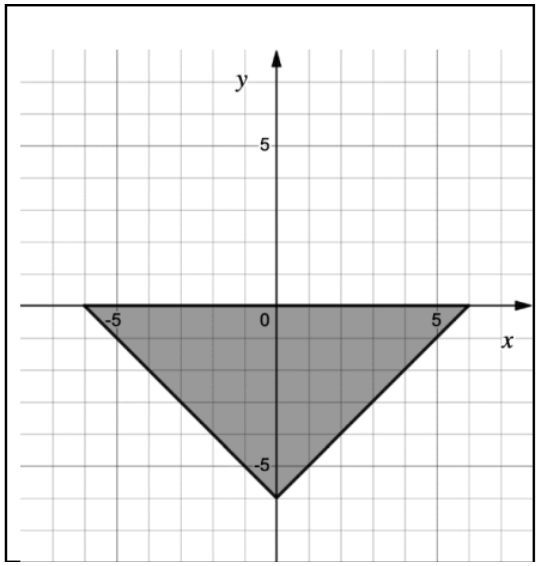
2.



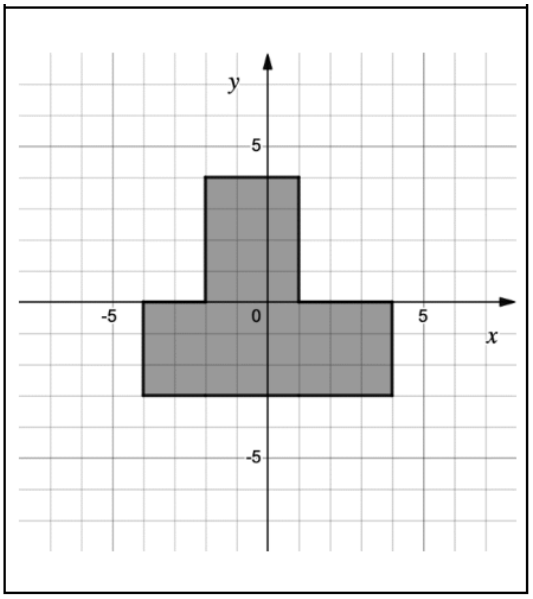
3.



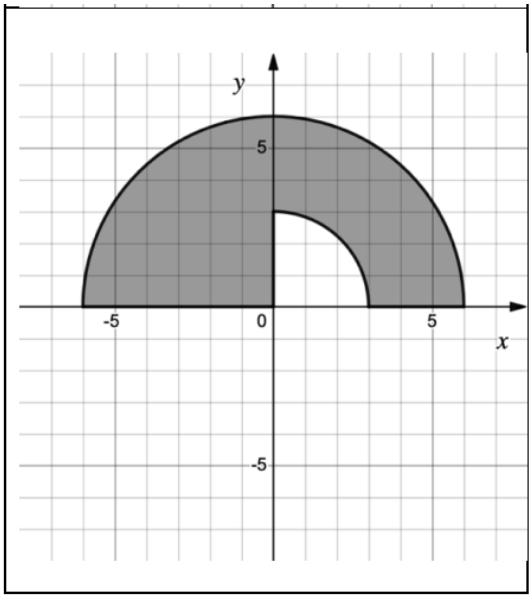
4.



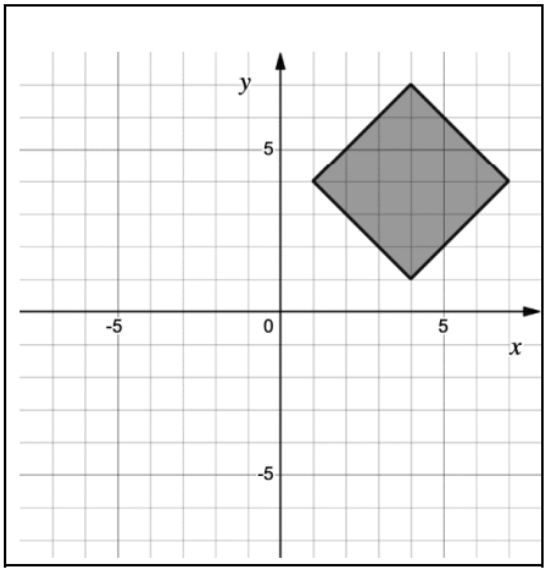
5.



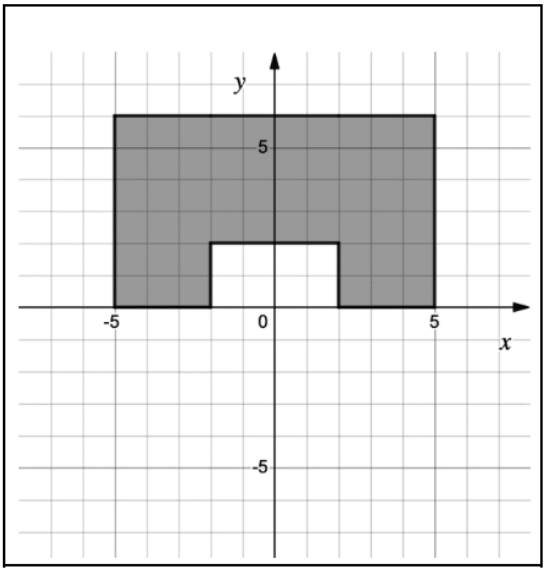
6.



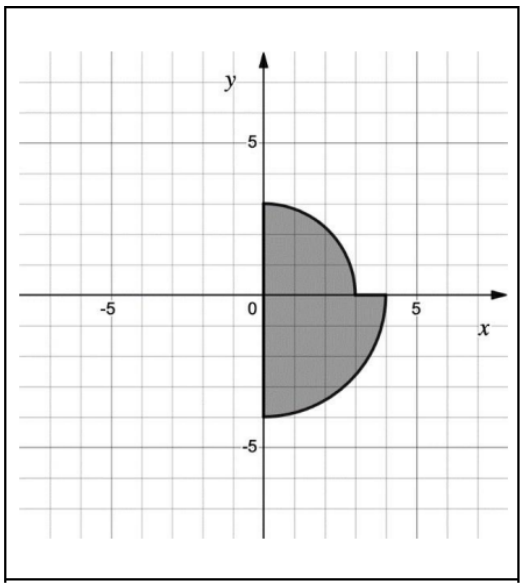
7.



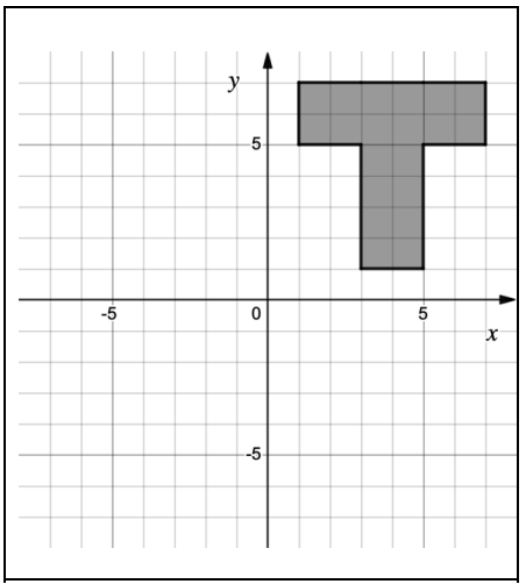
8.



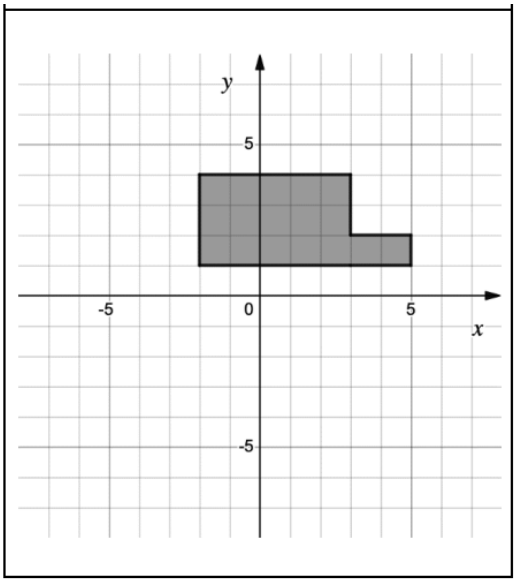
9.



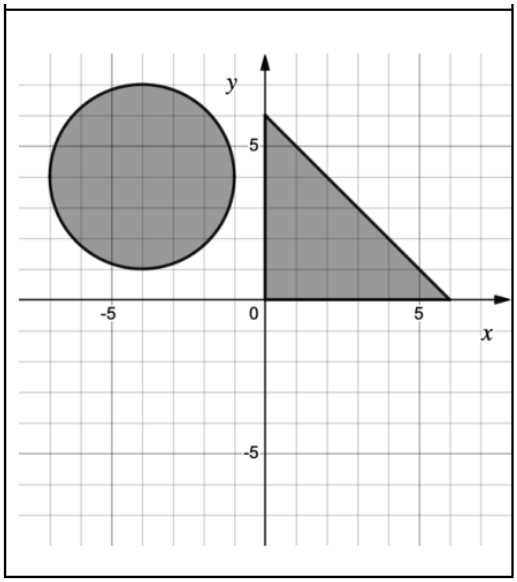
10.



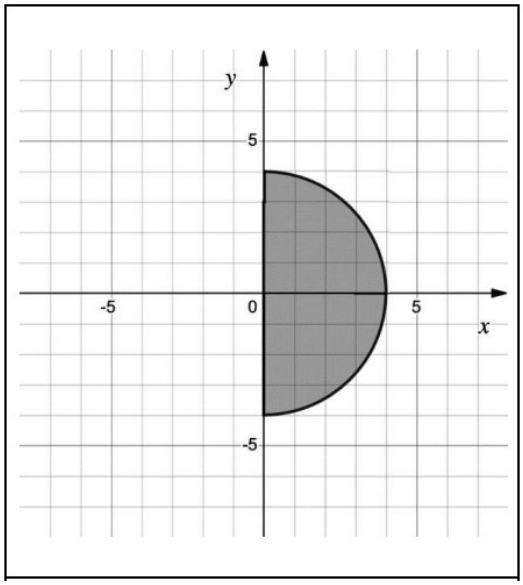
11.



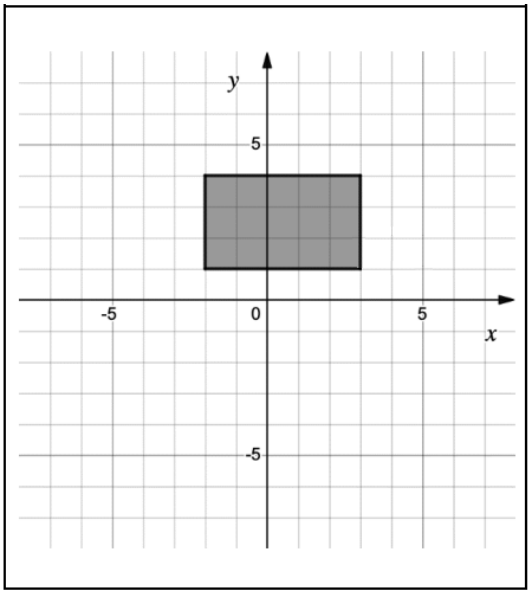
12.

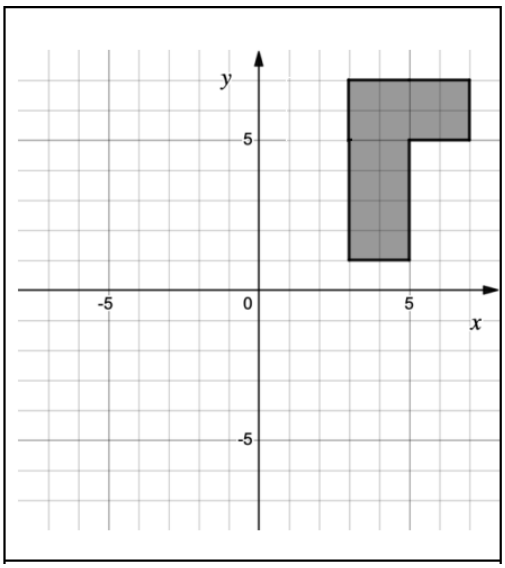


13.



14.



15. 

Завдання 3. Написати програму відповідно до номера варіанту.

Варіанти:

1. Напишіть програму, яка за номером дня тижня (цілому числу від 1 до 7) видає як результат кількість занять у вашій групі цього дня.

2. Напишіть програму, яка дозволяє за останньою цифрою числа визначити останню цифру його квадрата.

3. Напишіть програму, яка за заданим роком та номером місяця m визначає кількість днів цього місяця.

4. Для кожної введеної цифри (0 – 9) виведіть відповідну назву англійською мовою (0 – zero, 1 – one, 2 – two, ...).

5. Напишіть програму, яка за введеним числом (1 – 12) виводить назву відповідного місяця.

6. Напишіть програму, що дозволяє отримати словесне опис оцінок (1 – погано, 2 – незадовільно, 3 – «задовільно», 4 – «добре», 5 – «відмінно»).

7. Напишіть програму, яка за номером елемента запитувала б його відповідне значення та обчислювала б площу кола. Нехай елементами кола є радіус (перший елемент), діаметр (другий елемент) та довжина кола (третій елемент).

8. Напишіть програму, яка за заданим номером і значенням відповідного елемента обчислює значення всіх інших елементів трикутника. Нехай елементами рівнобедреного прямокутного трикутника є: катет a (1), гіпотенуза b (2), висота h (3), яка проведена з вершини прямого кута на гіпотенузу, площа S (4).

9. Напишіть програму, яка за номером місяця видає назву наступного місяця.

*Приклад:*1 (місяць – січень). Відповідь: лютий (наступний за січнем місяць).

10. Напишіть програму, яка б за введеним номером пори року (1 – зима, 2 – весна, 3 – літо, 4 – осінь) видавала відповідні цій порі року місяці та кількість днів у кожному з місяців.

11. Напишіть програму, яка за введеним номером одиниці виміру (1 – дециметр, 2 – кілометр, 3 – метр, 4 – міліметр, 5 – сантиметр) і довжині відрізка L видавала б відповідне значення довжини відрізка в метрах.

12. Напишіть програму, яка за числом, що вводиться, від 1 до 11 (номеру класу) видає відповідне повідомлення «Привіт, k-класник». Наприклад: "Привіт, першокласник", "Привіт, четверокласник".

13. Напишіть програму, яка за введеною кількістю від 1 до 12 (номером місяця) видає всі святкові дні, що припадають на цей місяць (наприклад, якщо введено число 1, то має вийти «1 січня – Новий рік).

14. Напишіть програму, яка за введеним номером дня місяця (1-31) визначає, до якої декади він належить (1-10 – перша декада, 11-20 – друга декада, 21-31 – третя декада).

15. Напишіть програму, яка за введеним числом від 1 до 30 визначає, чи є це число парним або непарним, а також виводить його словесний опис (наприклад: «двадцять чотири – парне число»).

Контрольні питання

1. Для чого призначені умовні конструкції в програмуванні?
2. Як записується базова конструкція if у Python?
3. Чим відрізняється if від elif?
4. Для чого використовується блок else?
5. Чи може існувати конструкція if без else?
6. Які типи виразів можна перевіряти в умовах if?
7. Як працюють логічні оператори and, or, not у поєднанні з умовами?
8. Що станеться, якщо замість if/elif/else написати кілька незалежних if?
9. Для чого використовується оператор pass в умовних конструкціях?
10. Як виглядає скорочений запис умовного виразу в одному рядку?
11. Які основні відмінності між конструкціями if/elif/else та match/case?
12. Як записується базовий синтаксис match/case у Python?
13. Чому оператор match/case вважається зручнішим за багаторівневий if/elif/else?
14. Для чого використовується символ підкреслення _ у конструкції match/case?
15. Чому символ _ потрібно розташовувати останнім у списку варіантів case?
16. Як у match/case перевірити одразу кілька значень в одному випадку?
17. Який символ використовується як оператор or у case?
18. Як у match/case можна додати додаткову умову з використанням if (guard)?
19. Наведіть приклади можливих практичних застосувань match/case.
20. У яких випадках краще використовувати if/elif/else, а у яких – match/case?

ЛАБОРАТОРНА РОБОТА №4
Тема: «Цикли for/while. Побудова блок-схем»

Мета роботи: Ознайомитися з принципами роботи циклічних операторів for та while у мові програмування Python, навчитися застосовувати їх для розв'язання практичних задач. Навчитись відображати логіку виконання програмних конструкцій у вигляді блок-схем.

Завдання до лабораторної роботи

Завдання 1. Написати програму з використанням циклів до наступних задач:

1. Написати програму, яка друкує таблицю множення від 1 до 10 у вигляді відформатованої таблиці. Використати вкладені цикли for.

2. Користувач вводить n чисел. Програма визначає максимальне і мінімальне число без використання вбудованих функцій max() та min(). Потрібно використати цикл for та умовні оператори. Написати програму для пошуку максимального і мінімального числа.

3. Користувач має рахунок із певною сумою. Він може вводити команди: зняти гроші, поповнити рахунок, показати баланс. Програма працює у циклі, поки користувач не введе «вихід». Використати while та if/elif/else. Написати програму «Симулятор банкомату».

4. Програма генерує випадковий пароль із літер, цифр і символів. Користувач вводить бажану довжину паролю. Використати цикл for, модуль random і список символів. Написати програму «Генератор паролів».

5. Користувач вводить список чисел через пробіл. Програма виводить окремо парні та непарні числа. Використати for та умовний оператор %. Написати програму для визначення парності чисел у списку.

Завдання 2. Написати програмний код та блок-схеми до наступних задач:

1. Написати алгоритм переходу пішохідного світлофора, з огляду на наступні правила:
   - дорога має дві смуги, острівок безпеки відсутній;
   - світлофор має лише два кольори: червоний та зелений;
   - кожні 10 секунд необхідно перевіряти колір світлофора;
   - аварія відбувається, якщо переходити дорогу на червоний колір.

2. У магазині 1 літр води коштує 7 грн, 2 літри – 10 грн, а покупець має 10 грн. Алгоритм має визначити, скільки літрів води може придбати покупець залежно від суми грошей, і показати результат кожного можливого сценарію покупки. Необхідно визначити всі можливі комбінації.

3. Програма для аналізу температури в кімнаті. Якщо температура >25°C – вмикається охолодження; якщо <18°C – нагрівання; інакше – кондиціонер вимкнено.

Алгоритм повинен враховувати зміну температури кожні 5 хвилин і виводити поточний режим кондиціонера.

4. У метро є три лінії: червона, синя і зелена. Кожна лінія має різну швидкість та кількість станцій. Алгоритм має визначити, якою лінією добратися швидше до заданої станції, і вивести кількість зупинок.

5. Написати гру «Питання-відповідь». Гравець отримує за правильну відповідь +5 балів, за пропуск 0 балів, за неправильну відповідь -2 бали. Алгоритм повинен обчислити загальний результат після серії питань та визначити, чи гравець пройшов рівень (не менше 20 балів).

Індивідуальні завдання до лабораторної роботи

Завдання 1. Згідно з номером Вашого варіанту необхідно написати програму до завдання та побудувати відповідну блок-схему.

Варіанти:

1. Квиток у кіно на будні коштує 50 грн, на вихідні – 80 грн. Покупець має 200 грн. Програма має показати, скільки квитків і на які дні можна купити, щоб не перевищити бюджет.

2. У меню є суп за 20 грн, основна страва за 35 грн, десерт за 40 грн. Студент має 250 грн. Программа повинна визначити всі можливі комбінації страв, які можна купити, і вивести залишок грошей.

3. Є квитки: плацкарт – 250 грн, купе – 400 грн, люкс – 700 грн. Пасажир має 500 грн. Програма має визначити, який тип квитка доступний і скільки грошей залишиться.

4. Є три парковки: A – 5 місць, B – 2 місця, C – 0 місць. Програма має визначити, де можна залишити авто, і повідомити користувача, якщо паркування немає.

5. Таксі пропонує три маршрути: короткий – 10 хв, середній – 20 хв, довгий – 30 хв. Клієнт має обмежений час – 25 хв. Програма має вибрати доступні маршрути та показати їх тривалість.

6. Автомат продає воду за 7 грн, сік за 12 грн і чай за 10 грн. Користувач має 15 грн. Програма має визначити, які напої можна купити, і вивести залишок грошей.

7. Працівники отримують базову зарплату 10 000 грн. Якщо робітник виконав план, він отримує бонус 20% від зарплати; якщо план виконано частково – 10%; якщо не виконано – бонусу немає. Програма має розрахувати остаточну зарплату для кожного працівника та вивести суму до виплати.

8. Банк приймає заявки на кредит. Якщо заявка менша за 50 000 грн і клієнт має позитивну кредитну історію – кредит видається одразу. Якщо заявка більша або історія негативна – необхідне схвалення менеджера. Програма повинен визначити, чи кредит схвалено, і вивести причину рішення.

9. На складі є 3 види товару: A – 5 шт, B – 3 шт, C – 2 шт. Клієнт робить замовлення, вказуючи кількість кожного товару. Програма має перевірити, чи можна виконати замовлення повністю, або повідомити, яких товарів не вистачає.

10. Ліфт обслуговує 5 поверхів. Якщо виклик з нижніх поверхів – ліфт піднімається; якщо з верхніх – спускається. Якщо ліфт зайнятий або несправний – виводиться відповідне повідомлення. Програма має визначати напрямок руху та стан ліфта.

11. Магазин продає зошити по 12 грн, ручки по 8 грн і олівці по 5 грн. Учень має 60 грн. Програма має визначити всі можливі набори канцелярії, які можна купити, не перевищуючи бюджет, і вивести залишок коштів.

12. У спортзалі доступні абонементи: місячний – 600 грн, на 3 місяці – 1600 грн, на 6 місяців – 3000 грн. Клієнт має 2000 грн. Програма повинна визначити, які абонементи він може придбати, та скільки грошей залишиться.

13. Онлайн-магазин доставляє товари: стандартна доставка – 3 дні, експрес – 1 день, самовивіз – у день замовлення. Клієнт хоче отримати товар не пізніше ніж за 2 дні. Програма має визначити доступні способи доставки та вивести їх.

14. Учень складає тест із 12 завдань. Якщо правильних відповідей 10-12 – оцінка «відмінно», 7-9 – «добре», 4-6 – «задовільно», менше 4 – «незадовільно». Програма має визначити оцінку за кількістю правильних відповідей.

15. В автоматі з продажу квитків доступні напрямки: місто A – 120 грн, місто B – 180 грн, місто C – 250 грн. Пасажир вводить суму грошей. Програма має визначити, у які міста він може купити квиток, або повідомити, якщо коштів недостатньо.

Завдання 2. За допомогою команди *print* та циклу for вивести на екран рисунок 1. Також потрібно створити власний малюнок з використанням циклів.

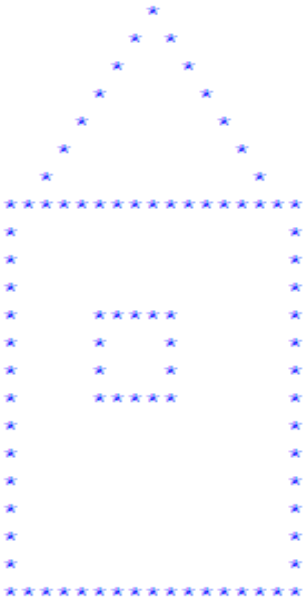


Рисунок 1 – Приклад виведення результату роботи алгоритму

Завдання 3. Згідно з номером Вашого варіанту необхідно написати програму для обчислення відповідного виразу при заданих параметрах.

Варіанти:

1. Дано дійсне a, натуральне n. Обчислити $a(a+1)\dots(a+n-1)$.
2. Дано дійсне a, натуральне n. Обчислити $\frac{1}{a}+\frac{1}{a+1}+\dots+\frac{1}{a(a+1)\dots(a+n)}$.
3. Дано дійсне a, натуральне n. Обчислити $\frac{1}{a}+\frac{1}{a^2}+\frac{1}{a^4}+\dots+\frac{1}{a^{2^n}}$.
4. Дано дійсне a, натуральне n. Обчислити $a(a-n)(a-2n)\dots(a+n^2)$.
5. Дано натуральне n. Обчислити $\left(1+\frac{1}{1^2}\right)\left(1+\frac{1}{2^2}\right)\dots\left(1+\frac{1}{n^2}\right)$.
6. Дано натуральне n. Обчислити $\frac{1}{\sin 1}+\frac{1}{\sin 1+\sin 2}+\dots+\frac{1}{\sin 1+\sin 2+\dots+\sin n}$.
7. Дано натуральне n. Обчислити $\frac{\cos 1}{\sin 1}\cdot\frac{\cos 1+\cos 2}{\sin 1+\sin 2}\cdot\dots\cdot\frac{\cos 1+\cos 2+\dots+\cos n}{\sin 1+\sin 2+\dots+\sin n}$.
8. Дано дійсне x, натуральне n. Обчислити $\sin x+\sin^2 x+\dots+\sin^n x$.
9. Дано дійсне x, натуральне n. Обчислити $\sin x+\sin x^2+\dots+\sin x^n$.
10. Дано натуральне n. Обчислити $\frac{1}{\cos 1}+\frac{1+2}{\cos(1+2)}+\dots+\frac{1+2+\dots+n}{\cos(1+2+\dots+n)}$.
11. Дано натуральне n. Обчислити $\left(1+\frac{\cos 1}{1^2}\right)\left(1+\frac{\cos 2}{1^2+2^2}\right)\dots\left(1+\frac{\cos n}{1^2+2^2+\dots+n^2}\right)$.
12. Дано натуральне n. Обчислити $\frac{\sin 1}{\sqrt{1}}+\frac{\sin 2}{\sqrt{1}+\sqrt{2}}+\dots+\frac{\sin n}{\sqrt{1}+\sqrt{2}+\dots+\sqrt{n}}$.
13. Дано натуральне n. Обчислити $\frac{1}{\sin n}+\frac{1}{\sin n+\sin(n-1)}+\dots+\frac{1}{\sin n+\dots+\sin 1}$.
14. Дано натуральне n. Обчислити $\frac{\cos n}{\sin 1}+\frac{\cos n\cdot\cos(n-1)}{\sin 1\cdot\sin 2}+\dots+\frac{\cos n\cdot\dots\cdot\cos 1}{\sin 1\cdot\sin 2\cdot\dots\cdot\sin n}$.
15. Дано натуральне n. Обчислити $\frac{a}{\cos a+1}+\frac{a^2}{\cos^2 a+1}+\dots+\frac{a^n}{\cos^n a+1}$.

Завдання 4. Згідно з номером Вашого варіанту необхідно написати програму для обчислення відповідного виразу при заданих параметрах.

Варіанти:

1. Дано натуральне n і дійсні $a_1, a_2, \dots, a_n$. Обчислити a_1, a_1+a_2, …, $a_1+a_2+\dots+a_n$.
2. Дано натуральне n і дійсні $a_1, a_2, \dots, a_n$. Обчислити a_1, a_1a_2, …, $a_1a_2\dots a_n$.
3. Дано натуральне n і дійсні $a_1, a_2, \dots, a_n$. Обчислити $a_1+1!$, $a_2+2!$, …, $a_n+n!$.
4. Дано натуральне n і дійсні $a_1, a_2, \dots, a_n$. Обчислити $\sin a_1$, $\sin(a_1+a_2)$, …, $\sin(a_1+a_2+\dots+a_n)$.
5. Дано натуральне n і дійсні $a_1, a_2, \dots, a_n$. Обчислити $\frac{a_1}{1}+\frac{a_2}{1+2}+\dots+\frac{a_n}{1+2+\dots+n}$.

6. Дано натуральне n і дійсні $a_1, a_2, \dots, a_n$. Обчислити $\frac{a_1+a_2+\cdots+a_n}{a_1 \cdot a_2 \cdot \dots \cdot a_n}$.

7. Дано натуральне n і дійсні $a_1, a_2, \dots, a_n$. Обчислити $\sqrt{|\, a_1 \,|} + \sqrt{|\, a_2 \,|} + \cdots + \sqrt{|\, a_n \,|}$.

8. Дано натуральне n і дійсні $a_1, a_2, \dots, a_n$. Обчислити $(a_1 + 1)(a_2 + 2) \dots (a_n + n)$.

9. Дано натуральне n і дійсні $a_1, a_2, \dots, a_n$. Обчислити $\sqrt{|\, a_1 \,| + 1}\sqrt{|\, a_2 \,| + 2} \dots \sqrt{|\, a_n \,| + n}$.

10. Дано натуральне n і дійсні $a_1, a_2, \dots, a_n$. Обчислити $\sin(a_1 + a_2 + \cdots + a_n)$.

11. Дано натуральне n і дійсні $a_1, a_2, \dots, a_n$. Обчислити $\sqrt{a_1^2 + 1} + \sqrt{a_2^2 + 2} + \cdots + \sqrt{a_n^2 + n}$.

12. Дано натуральне n і дійсні $a_1, a_2, \dots, a_n$. Обчислити $\sqrt{|\, a_1 a_2 \dots a_n \,|}$.

13. Дано натуральне n і дійсні $a_1, a_2, \dots, a_n$. Обчислити $\left(\sqrt{|\, a_1 \,|} - a_1\right)^2 + \left(\sqrt{|\, a_2 \,|} - a_2\right)^2 + \cdots + \left(\sqrt{|\, a_n \,|} - a_n\right)^2$.

14. Дано натуральне n і дійсні $a_1, a_2, \dots, a_n$. Обчислити $a_1 - a_2 + a_3 - \cdots - (-1)^{n+1} a_n$.

15. Дано натуральне n і дійсні $a_1, a_2, \dots, a_n$. Обчислити $\cos(a_1 a_2 \dots a_n)$.

Завдання 5. Методом ітерації обчислити корінь рівняння виду $f(x) = 0$, розміщеної на інтервалі [a, b] з абсолютною похибкою ε (у відповідності до варіанту). Визначити число ітерацій необхідне для знаходження кореня.

Варіанти:

№	Рівняння	a	b	ε
1	$\operatorname{tg} x = \frac{1}{x}$	0	$\frac{\pi}{2}$	10^{-4}
2	$\ln x = \frac{1}{x}$	1	2	10^{-4}
3	$e^{-x} = x$	0	1	10^{-3}
4	$\ln x = \frac{1}{x^2}$	1	2	10^{-4}
5	$e^{-x^2} = x$	0	1	10^{-5}
6	$e^{-x^2} = \ln x$	0	2	10^{-4}
7	$x - x^3 + 1 = 0$	1	2	10^{-3}
8	$x + 3 = x^3$	1	2	10^{-3}
9	$x + x^3 - 5 = 0$	1	2	10^{-3}
10	$1 - 5x + x^4 = 0$	0	1	10^{-4}
11	$1 - 5x + x^3 = 0$	0	1	10^{-4}
12	$1 - 3x + x^5 = 0$	1	2	10^{-5}

13	$e^x = \frac{1}{\sin x}$	0	$\frac{\pi}{2}$	10^{-3}
14	$x + 5 = x^3$	1	2	10^{-4}
15	$\ln\ x + 2 = \frac{1}{x}$	0	1	10^{-4}

Контрольні питання

1. Що таке цикл у програмуванні?
2. Які типи циклів підтримує мова Python?
3. У чому різниця між циклами for і while?
4. Який синтаксис циклу for у Python?
5. Як працює функція range() у циклі for?
6. Який синтаксис циклу while у Python?
7. Яка роль змінної-лічильника у циклі?
8. Що таке нескінченний цикл і як його уникнути?
9. Для чого використовується оператор break?
10. Для чого використовується оператор continue?
11. Чим відрізняється цикл while від циклу for за способом організації повторень?
12. Як можна реалізувати цикл із перевіркою умови на початку та в кінці?
13. Як працює вкладений цикл (цикл у циклі)?
14. Які типові помилки виникають при використанні циклів?
15. Як можна використовувати цикл для обробки списку або рядка?
16. Як відобразити структуру циклу на блок-схемі?
17. Які основні символи використовуються у блок-схемах алгоритмів?
18. Як на блок-схемі позначають умову переходу у циклі?
19. Як зобразити на блок-схемі вкладені цикли або комбіновану структуру (цикл + умова)?
20. Для чого необхідно складати блок-схеми перед написанням коду програми?

ЛАБОРАТОРНА РОБОТА №5
Тема: «Колекційні типи даних: списки, кортежі, масиви»

Мета роботи: Ознайомитися з колекційними типами даних у мові програмування Python, зокрема списками, кортежами та масивами. Навчитися створювати, змінювати та опрацьовувати ці структури, використовувати основні операції та методи для роботи з ними.

Завдання до лабораторної роботи

Завдання 1. Опанувати навички роботи зі списками, кортежами та масивами.

1.1. Виконайте операції зі списками:

- створіть список fruits, що містить: "apple", "banana", "cherry", "orange";
- додайте до списку елемент "kiwi";
- вставте "mango" на третю позицію;
- видаліть елемент "banana" за значенням;
- видаліть останній елемент списку за допомогою pop();
- виведіть кількість елементів у списку;
- виведіть усі елементи списку циклом for;
- використовуючи *List Comprehension,* створіть новий список fruits_char_a, який містить тільки ті фрукти, що містять літеру "a".

1.2. Напишіть код для сортування та копіювання списків:

- створіть список чисел: [23, 12, 45, 67, 34, 89, 10];
- відсортуйте список за зростанням;
- відсортуйте список за спаданням;
- створіть копію списку двома способами: за допомогою copy() та функції list();
- зробіть зворотній порядок елементів у списку.

1.3. Виконайте операції з кортежами:

- створіть кортеж colors зі значень: "red", "green", "blue", "yellow";
- виведіть перший та останній елементи кортежу;
- використовуючи діапазон індексів, виведіть елементи "green" та "blue";
- перетворіть кортеж на список і додайте елемент "purple", потім знову перетворіть на кортеж;
- створіть новий кортеж all_colors, об’єднавши colors із кортежем ("orange", "pink");
- використовуючи розпакування з *asterisk*, присвойте перший елемент змінній primary, останній елемент – secondary, а решту – списку others.

1.4. Застосуйте методи кортежів:

- створіть кортеж чисел [10, 20, 30, 20, 40, 20];
- поверніть індекс першого входження числа 20;

- порахуйте, скільки разів число 20 зустрічається в кортежі.

1.5. Виконайте операції з масивами (списками як масивами):
- створіть список cars = ["Ford", "Volvo", "BMW"];
- додайте елемент "Honda";
- видаліть другий елемент за допомогою pop();
- використайте метод insert() для додавання "Mercedes" на другу позицію;
- виведіть усі елементи циклом while;
- підрахуйте кількість елементів у списку;
- використайте метод sort() для сортування автомобілів за алфавітом.

1.6. Застосуйте List Comprehension в списках:
- створіть список чисел [5, 10, 15, 20, 25, 30];
- створіть новий список, де всі числа помножені на 2, використовуючи *List Comprehension;*
- виведіть новий список у зворотному порядку.

Завдання 2. Напишіть програми відповідно до поставлених завдань.
2.1. Напишіть програму «Список улюблених фруктів»:
- створіть список із 5 улюблених фруктів;
- за допомогою List Comprehension зробіть новий список, де назви фруктів у верхньому регістрі;
- додайте ще 2 фрукти за допомогою append() і insert();
- виведіть результат у відсортованому алфавітному порядку.

2.2. Напишіть програму «Таємна комбінація»:
- створіть список із цифр [0-9];
- за допомогою random.shuffle() перемішайте список;
- візьміть перші 4 цифри та сформуйте «таємну комбінацію»;
- виведіть її у вигляді рядка, наприклад «9351».

2.3. Напишіть програму «Розпакування тортів»:
- створіть кортеж із назв 5 тортів: "Чізкейк", "Тірамісу", "Медовик", "Наполеон", "Шоколадний";
- розпакуйте перший торт у змінну special, останній – у favorite, а решту – у список others;
- виведіть усі змінні у форматі:
  Special: Чізкейк
  Favorite: Шоколадний
  Others: ['Тірамісу', 'Медовик', 'Наполеон'].

2.4. Напишіть програму «Пошук числа»:
- створіть список із 20 випадкових чисел від 1 до 100;
- попросіть користувача ввести число;

- перевірте, чи присутнє це число в списку та повідомте його позиції (індекси);
- використайте цикл та методи списків.

2.5. Напишіть програму «Об’єднання колекцій»:
- створіть два кортежі з назвами тварин: ("кіт", "собака", "попугай") та ("корова", "кінь", "кролик");
- об’єднайте їх у один кортеж та виведіть усі елементи циклом for;
- підрахуйте, скільки елементів у новому кортежі.

2.6. Напишіть програму «Музичний плейлист»:
- створіть список із 5 улюблених пісень;
- додайте 2 нові пісні та вставте одну на другу позицію;
- видаліть пісню, яка вам вже набридла, за назвою;
- виведіть плейлист у зворотному порядку.

2.7. Напишіть програму «Індекси та елементи»:
- створіть список із 10 випадкових слів;
- виведіть усі слова з їхніми індексами у форматі:
  0: слово
  1: слово
  …
- використайте цикл for із range(len(...)).

2.8. Напишіть програму «Калькулятор середнього»:
- створіть список із оцінок студента (будь-які числа від 1 до 12);
- визначте максимальну та мінімальну оцінку;
- обчисліть середнє значення;
- виведіть повідомлення в такому форматі:
  максимум: 12
  мінімум: 6
  середнє: 9.2

2.9. Напишіть програму «Генератор випадкових кольорів»:
- створіть список кольорів: ["червоний", "зелений", "синій", "жовтий", "рожевий"];
- створіть новий список із випадковими кольорами, які повторюються 10 разів;
- порахуй, скільки разів кожен колір зустрічається;
- використайте методи count() та цикл for.

Індивідуальні завдання до лабораторної роботи

Завдання 1. Згідно з номером Вашого варіанту необхідно написати програму для формування масиву *K* за відповідним правилом.

Варіанти:

1. $K(i) = \frac{\sin i - i}{i+1}$, де $i = 1,2,\dots,10$.
2. $K(i) = \frac{3+i^3-(i-1)^2}{i+1}$, де $i = 0,\dots,16$.
3. $K(i) = \frac{3+i}{i^2+1}$, де $i = 1,\dots,25$.
4. $K(i) = \frac{(i-1)^3-3i}{i+1}$, де $i = 1,\dots,21$.
5. $K(i) = \frac{i^3-7}{2i+5}$, де $i = 1,\dots,25$.
6. $K(i) = \frac{3+i^3-(-1)^i}{i}$, де $i = 1,\dots,23$.
7. $K(i) = \frac{(-1)^i+3i}{i+1}$, де $i = 0,\dots,17$.
8. $K(i) = \frac{(-1)^i+(1-i)^2}{i}$, де $i = 1,\dots,9$.
9. $K(i) = \frac{i^2+1+3i}{i+1}$, де $i = 1,\dots,20$.
10. $K(i) = \frac{i^2+3i-7}{i+1}$, де $i = 0,\dots,21$.
11. $K(i) = \frac{i^3+3i-1}{i+1}$, де $i = 0,\dots,15$.
12. $K(i) = \frac{(-1)^i+i^2}{i}$, де $i = 1,\dots,9$.
13. $K(i) = \frac{\cos i+1}{i+1}$, де $i = 1,\dots,15$.
14. $K(i) = \frac{(-1)^i+(1+i)^3}{i+1}$, де $i = 1,\dots,10$.
15. $K(i) = \frac{i^2+2i-1}{i+5}$, де $i = 1,\dots,20$.

Завдання 2. Скласти програму, що виконує задану операцію над одновимірним масивом з 10 елементів. Тип даних, що використовується для задання елементів масиву визначається користувачем.

Варіанти:

1. Обчислити суму парних елементів.
2. Обчислити суму непарних елементів.
3. Підрахувати кількість додатних елементів.
4. Підрахувати кількість від'ємних елементів.
5. Обчислити суму додатних елементів.
6. Обчислити суму від'ємних елементів.
7. Обчислити добуток від'ємних елементів.
8. Обчислити суму перших п'яти додатних елементів.
9. Обчислити суму останніх п'яти додатних елементів.
10. Обчислити добуток перших п'яти додатних елементів.
11. Обчислити добуток останніх п'яти додатних елементів.
12. Обчислити суму елементів, що перевищують задане число A.
13. Обчислити добуток елементів, що не перевищують задане число A.
14. Обчислити суму елементів, що не перевищують задане число A.
15. Обчислити добуток елементів, що перевищують задане число A.

Завдання 3. Згідно з номером Вашого варіанту необхідно написати програму для формування масиву M за заданим правилом та виконати над ним відповідні операції.

Варіанти:

1. Дано масив M(15) невід'ємних дійсних чисел. Виконати сортування за спаданням.
2. Дано масив M(15) невід'ємних дійсних чисел. Виконати сортування за зростанням.
3. Дано масив M(15) дійсних чисел. Виконати сортування невід'ємних елементів за зростанням.
4. Дано масив M(15) дійсних чисел. Виконати сортування додатних елементів за зростанням.
5. Дано масив M(15) дійсних чисел. Упорядкувати за зростанням: спочатку від'ємні, а потім додатні числа.
6. Дано масив M(15) дійсних чисел. Елементи на непарних місцях розташувати у порядку зростання, а на парних в порядку спадання.
7. Дано масив M(15) дійсних чисел. Упорядкувати за спаданням: спочатку парні, а потім непарні числа.
8. Дано масив M(15) дійсних чисел. Упорядкувати за спаданням: спочатку нульові, потім від'ємні, додатні числа.
9. Дано масив M(15) дійсних чисел. Упорядкувати за спаданням: спочатку додатні, потім нульові, від'ємні числа.
10. Дано масив M(15) дійсних чисел. Упорядкувати масив їх абсолютних величин за зростанням.
11. Дано масив M(15) дійсних чисел. Упорядкувати за спаданням: спочатку непарні, а потім парні числа.
12. Дано масив M(15) дійсних чисел. Упорядкувати за зростанням: спочатку від'ємні, потім нульові, додатні числа.
13. Дано масив M(15) дійсних чисел. Виконати сортування додатних елементів за зростанням.
14. Дано масив M(15) дійсних чисел. Упорядкувати масив їх абсолютних величин за спаданням.
15. Дано масив M(15) дійсних чисел. Упорядкувати за спаданням: спочатку парні, а потім непарні числа.

Завдання 4. Згідно з номером Вашого варіанту необхідно написати програму для обробки одновимірних та двовимірних масивів в Python.

1. Написати програму для обробки одновимірного масиву N.
   - визначте, чи він монотонно зростає;
   - додайте можливість виявлення строгої монотонності (тобто коли всі сусідні елементи відрізняються).
2. Написати програму для обробки двовимірного масиву $N \times M$;
   - визначити найменший елемент кожного рядка матриці $N \times M$;
   - знайти найбільший елемент кожного стовпця матриці $N \times M$;

- обчислити суму елементів головної діагоналі матриці $N \times M$. Якщо матриця не є квадратною, програма повинна вивести відповідне повідомлення. Зробити матрицю квадратною щоб відбувся обрахунок;
- обчислити добуток елементів побічної діагоналі матриці $N \times M$. Якщо матриця не є квадратною, програма повинна вивести відповідне повідомлення. Зробити матрицю квадратною щоб відбувся обрахунок;
- порахувати кількість парних елементів у матриці $N \times M$;
- знайти середнє арифметичне значення елементів кожного рядка у матриці $N \times M$;
- транспонуйте матрицю $N \times M$;
- знайти індекси максимального елемента матриці $N \times M$;
- для кожного стовпчика заданої матриці знайти суму елементів, що розташовані нижче головної діагоналі;
- підрахувати кількість рядків, що містять від'ємні елементи;
- поділити елементи кожного стовпчика заданої матриці на останній її елемент;
- визначте, чи є в ньому два протилежні (тобто ті, що дають у сумі 0) числа. Якщо такі числа є у масиві, виведіть їх індекси. Якщо таких чисел у масиві немає, нічого не виводьте. Гарантується, що таких пар не більше однієї;
- спробуйте реалізувати фунцію так, щоб вона працювала ефективніше за O(N×M^2), якщо можливо;
- запропонуйте додаткові критерії аналізу масивів та реалізуйте відповідні функції.

Варіанти:

1. $N = 6, M = 3$	4. $N = 6, M = 3$	7. $N = 5, M = 9$	10. $N = 6, M = 8$	13. $N = 5, M = 6$
2. $N = 7, M = 7$	5. $N = 7, M = 7$	8. $N = 10, M = 4$	11. $N = 9, M = 7$	14. $N = 8, M = 8$
3. $N = 8, M = 5$	6. $N = 8, M = 5$	9. $N = 6, M = 8$	12. $N = 3, M = 10$	15. $N = 6, M = 9$

Завдання 5. Згідно з номером варіанту необхідно написати програму для обчислення матриці C, за умови коли задано матриці $A, B, E = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$.

1. $A = \begin{pmatrix} 1 & 7 & 1 \\ 0 & 4 & 0 \\ 1 & -3 & 1 \end{pmatrix}, B = \begin{pmatrix} 1 & 0 & -1 \\ 0 & 2 & 1 \\ 5 & 0 & 6 \end{pmatrix}, C = A^2 - B.$
2. $A = \begin{pmatrix} 2 & 2 & 3 \\ 1 & 4 & 0 \\ 1 & -1 & 2 \end{pmatrix}, B = \begin{pmatrix} 3 & 0 & -2 \\ -1 & 2 & -1 \\ 0 & 1 & 2 \end{pmatrix}, C = A(A + B).$
3. $A = \begin{pmatrix} 0 & 7 & 3 \\ 3 & 4 & 1 \\ 1 & -3 & 1 \end{pmatrix}, B = \begin{pmatrix} 4 & 0 & -1 \\ 5 & 2 & -1 \\ 3 & 0 & 5 \end{pmatrix}, C = B(A - B).$
4. $A = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 4 & 0 \\ 7 & -3 & 8 \end{pmatrix}, B = \begin{pmatrix} 1 & 4 & -3 \\ 2 & 0 & 5 \\ 3 & 6 & 1 \end{pmatrix}, C = A^2 + B.$

5. $A = \begin{pmatrix} 2 & 7 & 3 \\ 5 & 4 & 1 \\ 1 & -1 & 2 \end{pmatrix}, B = \begin{pmatrix} 2 & 0 & -3 \\ 5 & 0 & -1 \\ 3 & 4 & 5 \end{pmatrix}, C = (A - E)B.$

6. $A = \begin{pmatrix} 3 & 5 & 3 \\ 3 & 0 & 1 \\ 1 & -3 & 0 \end{pmatrix}, B = \begin{pmatrix} 3 & 2 & -1 \\ 1 & 3 & -1 \\ 7 & 0 & 2 \end{pmatrix}, C = E + A^2 + B.$

7. $A = \begin{pmatrix} 1 & 2 & 1 \\ 4 & 1 & 0 \\ 2 & -2 & 1 \end{pmatrix}, B = \begin{pmatrix} 0 & 6 & -1 \\ 2 & 1 & 5 \\ 4 & 3 & 1 \end{pmatrix}, C = A(E + B^2).$

8. $A = \begin{pmatrix} 2 & 2 & 1 \\ -1 & 1 & 0 \\ 4 & -3 & 5 \end{pmatrix}, B = \begin{pmatrix} 1 & 4 & -3 \\ 2 & 0 & 1 \\ 3 & 2 & 1 \end{pmatrix}, C = A \cdot B + B.$

9. $A = \begin{pmatrix} 1 & 2 & 1 \\ 4 & 1 & 0 \\ 2 & -2 & 6 \end{pmatrix}, B = \begin{pmatrix} 7 & 0 & 2 \\ 2 & 1 & 5 \\ 1 & 4 & -3 \end{pmatrix}, C = A(B^2 - A).$

10. $A = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 1 & 2 \\ 7 & -3 & 1 \end{pmatrix}, B = \begin{pmatrix} 2 & 1 & -1 \\ 2 & 1 & 5 \\ 1 & 4 & -3 \end{pmatrix}, C = A(E - B^2).$

11. $A = \begin{pmatrix} 1 & 2 & 1 \\ 3 & 4 & 5 \\ 2 & -4 & 1 \end{pmatrix}, B = \begin{pmatrix} 2 & 1 & 5 \\ 3 & 5 & 3 \\ 4 & 3 & 1 \end{pmatrix}, C = (E - A^2)B.$

12. $A = \begin{pmatrix} 3 & 1 & 2 \\ 1 & 4 & 1 \\ 2 & 3 & 4 \end{pmatrix}, B = \begin{pmatrix} 1 & 4 & 3 \\ 2 & 3 & 2 \\ 3 & 5 & 1 \end{pmatrix}, C = (E - B^2)A.$

13. $A = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 3 & 0 \\ 7 & -3 & 8 \end{pmatrix}, B = \begin{pmatrix} 1 & 4 & -3 \\ 2 & 0 & 5 \\ 3 & 6 & 1 \end{pmatrix}, C = A^2 + B.$

14. $A = \begin{pmatrix} 1 & 2 & 1 \\ 1 & 3 & 1 \\ 2 & -1 & 5 \end{pmatrix}, B = \begin{pmatrix} 0 & 6 & -5 \\ 2 & 1 & 5 \\ 4 & 3 & 1 \end{pmatrix}, C = A(E + B^2).$

15. $A = \begin{pmatrix} 1 & 2 & 1 \\ 1 & 3 & 1 \\ 2 & -3 & 4 \end{pmatrix}, B = \begin{pmatrix} 0 & 5 & -3 \\ 2 & 1 & 4 \\ 1 & 3 & 1 \end{pmatrix}, C = B(E + A^2).$

Контрольні питання

1. Що таке колекційні типи даних у Python і які їх основні види?
2. Чим відрізняється список (list) від кортежу (tuple)?
3. Які властивості має список у Python?
4. Як створити порожній список?
5. Як звернутися до елемента списку за індексом?
6. Як за допомогою негативної індексації отримати останній елемент списку?
7. Яка різниця між методами append() і extend()?
8. Для чого використовується метод insert()?
9. Як видалити елемент зі списку за значенням?
10. Що робить метод pop()?
11. Як можна об’єднати два списки між собою?

12. Як створити кортеж у Python? Наведи приклад.
13. Що означає, що кортеж є незмінним (immutable)?
14. Як створити кортеж лише з одного елемента?
15. Як отримати доступ до елемента кортежу за індексом?
16. Як можна «оновити» значення кортежу, якщо він незмінний?
17. Поясни, що означає розпакування кортежу (tuple unpacking).
18. Як працює оператор * (зірочка) під час розпакування кортежів?
19. Які два основні методи існують у кортежів?
20. Чому Python не має вбудованого типу масив, і як можна його імітувати?

ЛАБОРАТОРНА РОБОТА №6
Тема: «Колекційні типи даних: словники, множини»

Мета роботи: Ознайомитися з колекційними типами даних у мові програмування Python, зокрема словниками та множинами. Навчитися створювати, змінювати та опрацьовувати ці структури, використовувати основні операції, методи та вбудовані функції для ефективної роботи з ними.

Завдання до лабораторної роботи

Завдання 1. Написати програму для роботи зі словниками.

1.1. Створіть словник student, який містить такі пари ключ–значення:
- "name" – ім’я студента;
- "age" – вік;
- "group" – номер групи.

Виведіть словник на екран.

1.2. За допомогою ключа виведіть значення "name" зі словника student.

1.3. Змініть вік студента у словнику student на новий.

1.4. Додайте новий елемент "grade" зі значенням 95 у словник.

1.5. Видаліть елемент "group" зі словника за допомогою методу pop().

1.6. Виведіть усі ключі, значення та пари ключ–значення окремо, використовуючи методи keys(), values() та items().

1.7. Перевірте, чи міститься у словнику ключ "grade", використовуючи оператор in.

1.8. Створіть копію словника student двома способами: за допомогою copy() та dict().

1.9. Створіть словник students, який містить інформацію про трьох студентів (вкладені словники). Виведіть ім’я другого студента.

1.10. Оновіть всі дані студента, використовуючи метод update().

Завдання 2. Написати програму для роботи з множинами.

2.1. Створіть множину fruits, що містить елементи "apple", "banana", "cherry". Виведіть її на екран.

2.2. Додайте новий елемент "orange" до множини за допомогою методу add().

2.3. Додайте до множини fruits елементи іншої множини tropical = {"mango", "papaya"} за допомогою update().

2.4. Видаліть елемент "banana" за допомогою discard() та поясніть різницю між discard() і remove().

2.5. Створіть дві множини чисел і об’єднайте їх за допомогою union().

2.6. Знайдіть спільні елементи двох множин за допомогою intersection().

2.7. Визначте, які елементи присутні в одній множині, але відсутні в іншій, за допомогою difference().

2.8. Знайдіть елементи, що є лише в одній з двох множин, використовуючи symmetric_difference().

2.9. Створіть дві множини та перевірте:

– чи є одна підмножиною іншої (issubset(), <=);
– чи є одна надмножиною іншої (issuperset(), >=).

2.10. Створіть множину з кількома елементами та очистіть її за допомогою clear().

Індивідуальні завдання до лабораторної роботи

Завдання 1. Написати програму для проведення математичних операцій зі словником чисел:

– створіть словник numbers, у якому ключі – числа з проміжку згідно з варіантом, а значення – їх куби;
– додайте до словника ще три числа (11, 12, 13) та обчисліть їх куби;
– знайдіть суму всіх значень словника;
– знайдіть добуток всіх значень словника;
– створіть новий словник, де значення – квадратний корінь значень старого словника;
– виведіть ключі, для яких значення діляться на 2 без залишку.

Варіанти:

1. 1–10	4. 1–12	7. 2–16	10. 5–19	13. 3–12
2. 2–11	5. 4–13	8. 3–17	11. 1–10	14. 4–13
3. 3–12	6. 1–15	9. 1–14	12. 2–11	15. 5–14

Завдання 2. Написати програму з використанням словника для проведення обчислень математичної фукнції:

– створіть словник func_values, де ключі – числа з проміжку згідно з варіантом, а значення – результат функції $f(x) = 3x^2 - 2x + 1$;
– обчисліть суму значень функції для парних ключів;
– обчисліть добуток значень функції для непарних ключів;
– виведіть ключі, значення яких більше середнього значення функції.

Варіанти:

1. 0–9	4. 0–11	7. 1–15	10. 4–18	13. 2–11
2. 1–10	5. 3–12	8. 2–16	11. 0–9	14. 3–12
3. 2–11	6. 0–14	9. 0–13	12. 1–10	15. 4–13

Завдання 3. Написати програму з використанням множини для проведення зазначених обчислень:

– створіть множини A та B, що містять числа у проміжку згідно з варіантом;
– знайдіть елементи, які є квадратами чисел і належать одночасно множинам A і B;
– знайдіть числа, що діляться на 3, і є у множині A, але не належать B;
– знайдіть суму елементів об’єднання A та B;
– знайдіть добуток елементів перетину множин A та B.

Варіанти:

1. A=1–20, B=10–30 2. A=2–21, B=11–31 3. A=3–22, B=12–32	4. A=1–25, B=5–29 5. A=4–24, B=14–33 6. A=1–20, B=10–29	7. A=2–21, B=11–30 8. A=3–22, B=12–31 9. A=1–23, B=9–28
10. A=5–25, B=15–35 11. A=1–20, B=10–30 12. A=2–21, B=11–31	13. A = 3–22, B = 12–32 14. A = 4–24, B = 14–33 15. A = 5–25, B = 15–35	

Завдання 4. Написати програму з використанням словників та множин для проведення зазначених обчислень.

1. Створіть словник students_scores згідно з варіантом, у якому ключами є імена студентів, а значеннями – множини їх оцінок.
2. Знайдіть студента з найбільшою середньою оцінкою.
3. Знайдіть усіх студентів, які мають хоча б одну оцінку 12.
4. Створіть множину всіх оцінок, які зустрічаються у всіх студентів.
5. Видаліть з кожного студента оцінки, менші за 6, і виведіть новий словник.

Варіанти:

1. {"Іван":{8,9,10}, "Марія":{7,12}, "Олег":{5,6,9}};
2. {"Петро":{6,8,9}, "Ірина":{10,11}, "Марія":{9,12}};
3. {"Олег":{7,8,10}, "Іван":{6,9,12}, "Петро":{5,7}};
4. {"Марія":{8,10,11}, "Ірина":{7,9,12}, "Іван":{5,6,8}};
5. {"Олег":{6,7,12}, "Петро":{5,9,11}, "Марія":{8,10}};
6. {"Іван":{7,9,12}, "Марія":{6,8,10}, "Ірина":{9,11}};
7. {"Петро":{5,7,10}, "Олег":{6,9,12}, "Марія":{8,11}};
8. {"Ірина":{7,10,12}, "Іван":{5,8,9}, "Олег":{6,11}};
9. {"Марія":{9,10,12}, "Петро":{5,7,8}, "Іван":{6,11}};
10. {"Олег":{8,10,12}, "Ірина":{6,9,11}, "Марія":{7,12}};
11. {"Іван":{6,8,9}, "Марія":{7,11}, "Петро":{5,10}};
12. {"Олег":{6,7,12}, "Ірина":{5,9,10}, "Марія":{8,12}};
13. {"Іван":{7,9,11}, "Петро":{6,8,10}, "Олег":{5,9}};
14. {"Марія":{8,10,12}, "Ірина":{7,11}, "Іван":{6,9}};
15. {"Петро":{6,9,11}, "Олег":{7,10,12}, "Марія":{5,8,9}}.

Завдання 5. Написати програму з використанням колекцій та множин для проведення зазначених математичних обчислень.

1. Створіть словник sequence, де ключі – числа від 1 до 15, значення – $K(i)$ згідно з варіантом.
2. Знайдіть суму значень словника, що перевищують 10.
3. Створіть новий словник, де значення піднесені до квадрату.
4. Створіть множину зі значень нового словника, видаліть всі парні числа.
5. Знайдіть максимальне та мінімальне значення у множині.

Варіанти:

1. $K(i) = \frac{i^3-2i+1}{i+1}$.

2. $K(i) = \frac{i^2+3i-4}{i+2}$.
3. $K(i) = \frac{2i^3-i+1}{i+1}$.
4. $K(i) = \frac{3i^2-5i+2}{i+3}$.
5. $K(i) = \frac{i^3+i^2-1}{i+1}$.
6. $K(i) = \frac{i^2-3i+4}{i+2}$.
7. $K(i) = \frac{i^3-i^2+2}{i+1}$.
8. $K(i) = \frac{2i^2+i-1}{i+1}$.
9. $K(i) = \frac{3i^3-2i}{i+2}$.
10. $K(i) = \frac{i^2+2i+3}{i+1}$.
11. $K(i) = \frac{i^3-7}{2i+3}$.
12. $K(i) = \frac{(-1)^i+i^2}{i+1}$.
13. $K(i) = \frac{i^2+3i+1}{i+2}$.
14. $K(i) = \frac{i^3+2i-1}{i+1}$.
15. $K(i) = \frac{i^2-i+2}{i+3}$.

Завдання 6. Написати програму з використанням словників і множин для проведення динамічних операції над ними.

1. Створіть словник mat_dict для матриці розмірністю згідно з варіантом, де ключ – кортеж (i, j) індексів, значення формули згідно з варіантом.

2. Знайдіть суму значень всіх елементів головної діагоналі.

3. Знайдіть добуток значень побічної діагоналі.

4. Створіть множину всіх значень, що діляться на 3.

5. Створіть словник row_sums, де ключ – номер рядка, значення – сума елементів цього рядка.

Варіанти:

№	Розмір матриці	Формула для (i,j)	№	Розмір матриці	Формула для (i,j)	№	Розмір матриці	Формула для (i,j)
1	5×5	$i^2 + j^2$	6	5×5	$i^3 + j$	11	5×5	$i \cdot j + i + j$
2	4×4	$i + j$	7	4×4	$i^2 + 2j$	12	4×4	$i^3 - j^2$
3	5×5	$i^2 - j^2$	8	5×5	$i + j^3$	13	5×5	$3i + 2j$
4	6×6	$i \cdot j$	9	6×6	$2i - j^2$	14	5×5	$i^2 - i \cdot j + j^2$
5	5×5	$i + j^2$	10	5×5	$i^2 + j^2 + 1$	15	6×6	$i^2 + j^2 - i \cdot j$

Контрольні питання

1. Що таке словник у Python і як він відрізняється від списку чи кортежу?
2. Який синтаксис створення словника за допомогою фігурних дужок {}? Наведіть приклад.
3. Як створити словник за допомогою конструктора dict()?
4. Як отримати значення елемента словника за ключем? Поясніть різницю між використанням квадратних дужок [] та методом get().
5. Які методи словника повертають ключі, значення та пари ключ-значення?
6. Що відбувається, якщо створити словник з дубльованими ключами?
7. Як змінити значення існуючого елемента словника? Наведіть приклад.
8. Які способи додавання нових елементів до словника існують?
9. Поясніть різницю між методами pop(), popitem(), clear() та ключовим словом del.
10. Як перебрати словник у циклі for, щоб отримати тільки ключі, тільки значення або пари ключ-значення?
11. Як створити копію словника, щоб зміни в новому словнику не впливали на оригінал?
12. Що таке вкладений словник і як отримати доступ до елемента у вкладеному словнику?
13. Що таке множина в Python і чим вона відрізняється від списку?
14. Які особливості елементів множини? Поясніть поняття невпорядкованості, незмінності та унікальності.
15. Як створити множину за допомогою фігурних дужок {} та функції set()?
16. Як додати один або кілька елементів до множини? Наведіть приклади методів add() та update().
17. Чим відрізняються методи remove() і discard() для видалення елементів множини?
18. Який результат виконання методу pop() для множини і чому він непередбачуваний?
19. Як перевірити, чи містить множина певний елемент?
20. Які основні операції над множинами існують у Python: об’єднання, перетин, симетрична різниця, різниця? Наведіть приклади відповідних методів.

ЛАБОРАТОРНА РОБОТА №7
Тема: «Функції та лямбда-функції. Область дії (scope). Обробка помилок»

Мета роботи: Ознайомитися з функціями та лямбда-функціями у мові програмування Python. Навчитися оголошувати та викликати функції, розуміти область дії змінних (scope) і правила їх використання. Засвоїти принципи обробки помилок за допомогою конструкцій try, except, finally для підвищення надійності програм.

Завдання до лабораторної роботи

1. Долучитись до курсу «Python Essentials 2 by Cisco» за посиланням: https://www.netacad.com/courses/python-essentials-2.
2. **Написати програму з використанням функцій:**
   - напишіть просту функцію, яка виводить повідомлення на екран;
   - створіть функцію, що приймає два аргументи та повертає їх суму;
   - реалізуйте функцію з параметром за замовчуванням (наприклад, виведення країни походження користувача).
3. **Написати функції з передаванням аргументів:**
   - напишіть функцію, яка приймає довільну кількість аргументів (*args) і виводить усі передані значення;
   - створіть функцію, яка приймає довільну кількість ключових аргументів (**kwargs) і виводить їх у вигляді словника;
   - реалізуйте функцію, що приймає список як аргумент і виводить усі його елементи у циклі.
4. **Написати програму з використанням лямбда-функцій:**
   - створіть лямбда-функцію, яка додає до числа 10;
   - створіть лямбда-функцію, що множить два числа;
   - реалізуйте функцію, яка повертає лямбда-вираз (наприклад, множення на задане число), і продемонструйте її роботу для подвоєння та потроєння значень.
5. **Дослідити область дії змінних у програмі (Scope):**
   - створіть програму з глобальною та локальною змінними, щоб продемонструвати різницю між ними;
   - реалізуйте приклад з вкладеною функцією, використовуючи ключове слово nonlocal;
   - поясніть, у якому випадку необхідно застосовувати ключове слово global.
6. **Застосувати обробку помилок із використанням конструкції try-except:**
   - напишіть програму, яка обробляє помилку введення користувача (наприклад, спробу поділу на нуль або введення нечислового значення);

– використайте блоки try, except, else та finally у прикладі, що демонструє роботу кожного з них;
– реалізуйте виклик власного винятку за допомогою raise (наприклад, коли введене число є від'ємним або некоректного типу).

Індивідуальні завдання до лабораторної роботи

Завдання 1. Відповідно до варіанту побудувати блок-схему та скласти програму з оформленням виведення результатів за допомогою функцій. Додатково використати лямбда-функції та реалізувати обробку можливих помилок за допомогою конструкції try-except.

Варіанти:

1. Трикутник задано координатами вершин. Знайти периметр трикутника.
2. Трикутник задано координатами вершин. Знайти площу трикутника.
3. Зростаючу геометричну прогресію задано першим членом a, коефіцієнтом q та загальною кількістю членів n. Знайти суму членів геометричної прогресії.
4. Зростаючу геометричну прогресію задано першим членом a, коефіцієнтом q та загальною кількістю членів n. Знайти k-член геометричної прогресії.
5. Визначити час падіння каменя на поверхню Землі з висоти H.
6. Дано гіпотенузу і катет прямокутного трикутника. Знайти другий катет та радіус вписаного кола.
7. Відома довжина кола. Знайти площу круга, який обмежений цим колом
8. Знайти площу рівнобічної трапеції з основами a і b і кутом при більшій основі α.
9. Трикутник заданий довжинами сторін. Знайти радіуси вписаного та описаного кіл.
10. Знайти площу кільця з внутрішнім радіусом R1 і зовнішнім – R2.
11. Відома площа круга. Знайти довжину кола.
12. Арифметичну прогресію задано першим членом a, кроком h та загальною кількістю членів n. Знайти суму членів арифметичної прогресії.
13. Арифметичну прогресію задано першим членом a, кроком h та загальною кількістю членів n. Знайти k-член арифметичної прогресії.
14. Дано довжину ребра куба. Знайти його об'єм та площу.
15. Дано два дійсних числа. Знайти їхнє середнє арифметичне та середнє геометричне.

Завдання 2. Відповідно до варіанту побудувати блок-схему та скласти програму обчислення значення функції *f(x)* при заданому значенні аргументу. Додатково використати лямбда-функції та реалізувати обробку можливих помилок за допомогою конструкції try-except.

Варіанти:

1. $f(x)=\sqrt{e^{2,2x}}-\left\|\sin\frac{\pi x}{x+2/3}\right\|+1.7$	9. $f(x)=1.1e^{x}+\|\cos\sqrt{\pi x}\|-\frac{4}{9}$
2. $f(x)=\left(\frac{1}{7}+\ln\sqrt{x}\right)e^{\sqrt{x-2}}$	10. $f(x)=\frac{\ln(\sqrt{\|x-2\|}+1.2)}{(2+e^{x})}+\sqrt[3]{\frac{2}{x}}$
3. $f(x)=\frac{\sqrt{x}\sin\frac{x^2}{2}-1.3}{\sqrt[5]{x}+e^{3x}+\|\cos x\|}$	11. $f(x)=\frac{1}{3}\sqrt{\|\sin x\|}\cdot\sqrt[3]{e^{0.12x}}$
4. $f(x)=\sqrt{e^{\|\sin x\|}}+2\ln 3x-\frac{1}{9}$	12. $f(x)=\frac{\sqrt[5]{e^{1/3-x}}}{\sqrt{x^2+x^4+\ln\|x-3.4\|}}$
5. $f(x)=\left(\sqrt{1+x^2}+\frac{\|\ln x^3\|}{1.6+x^4}\right)\sin\ 7x$	13. $f(x)=\sqrt{\sin\frac{x^3}{2}+\sqrt[3]{e^{1.3x}}+e^{-1.3x}}$
6. $f(x)=\frac{\sqrt{\frac{1}{5}+\sqrt[5]{e^{x}}}}{\ln(x^2-1.3)}$	14. $f(x)=\frac{\|x\ln x-\frac{4}{7}\|\sqrt{x}}{\sqrt[5]{e^{4x-1.1}}}$
7. $f(x)=1.8+\ln\left(4\frac{2}{7}-\text{tgsin}\frac{5x}{3}\right)$	15. $f(x)=\sqrt{e^{2x}\sqrt{x}-\frac{x+1/3}{x}}\|\cos 2.5x\|$
8. $f(x)=\frac{\|\sin\sqrt{10.5x}\|}{\sqrt[3]{x^2-0.143}}+2\pi x$	

Завдання 3. Відповідно до варіанту скласти програму з оформленням виведення результатів за допомогою функцій. Додатково використати лямбда-функції та реалізувати обробку можливих помилок за допомогою конструкції try-except.

Варіанти:

1. Реалізуйте функцію factorial(n), яка обчислює факторіал числа n рекурсивно. Передбачте обробку помилок: якщо введено від'ємне число або неціле значення, виведіть повідомлення про помилку. Порада: використати try, except і рекурсивний виклик.

2. Напишіть функцію, яка знаходить найбільший простий дільник заданого числа n. Реалізуйте варіант із лямбда-функцією для перевірки простоти числа. При введенні некоректних даних реалізуйте виняток із повідомленням "Введіть додатне ціле число!".

3. Створіть функцію prime_factors(n), яка повертає список усіх простих множників числа. Для зручності реалізуйте допоміжну анонімну лямбда-функцію для перевірки простоти. Забезпечте обробку помилок (наприклад, якщо введено 0 або від'ємне число).

4. Напишіть функцію, що обчислює суму ряду: $S=\sum_{k=1}^{n}\frac{(-1)^{k+1}}{k^2}$. Використайте лямбда-функцію для обчислення окремого члена ряду. Перевірте, щоб n було натуральним числом.

5. Напишіть функцію solve_quadratic(a, b, c), яка знаходить корені квадратного рівняння: $ax^2+bx+c=0$. Використайте обробку помилок, щоб запобігти діленню на нуль (якщо a = 0). Застосуйте лямбда-функцію для обчислення дискримінанта.

6. Число називається числом Армстронга, якщо сума його цифр, піднесених до степеня кількості цифр, дорівнює самому числу. Напишіть функцію is_armstrong(n) з використанням лямбда-виразу та обробки винятків.

7. Напишіть програму, яка наближено обчислює число π за допомогою ряду Лейбніца: $\pi=4\sum_{k=0}^{n}\frac{(-1)^{k}}{2k+1}$. Використайте лямбда-функцію для обчислення члена ряду та функцію з параметром за замовчуванням (наприклад, n=1000). Передбачте обробку помилок для некоректних аргументів.

8. Напишіть функцію lcm(a, b) для знаходження НСК двох чисел через їхній НСД (найбільший спільний дільник). Використайте лямбда-функцію для знаходження НСД за алгоритмом Евкліда. Перевірте введення на цілі та додатні значення.

9. Створіть функцію power_table(n, m), яка будує таблицю степенів від 1 до n для степенів 1...m. Використайте вкладену функцію або лямбда-вираз для обчислення степенів. Реалізуйте обробку помилок для нечислових значень.

10. Напишіть функцію, яка приймає список чисел і повертає гармонічне та геометричне середнє. Застосуйте лямбда-функції для обчислень та обробку помилок у разі порожнього списку або наявності від'ємних чисел.

11. Створіть програму, яка для довільного числа n: перевіряє, чи воно є простим. Якщо ні – розкладає його на множники; знаходить суму його цифр та визначає, чи ця сума є простим числом; обробляє можливі помилки введення (наприклад, рядкові або від'ємні значення). Для реалізації використайте кілька функцій, лямбда-вирази та обробку винятків.

12. Створіть функцію digit_statistics(n), яка повертає кількість цифр числа, їх суму та добуток. Для обчислення суми та добутку цифр застосуйте лямбда-функції. Реалізуйте обробку помилок для випадків, коли введено неціле або від'ємне значення.

13. Напишіть функцію average_even(numbers), яка знаходить середнє арифметичне парних чисел у списку. Використайте лямбда-функцію для фільтрації парних елементів. Врахуйте обробку винятків у разі порожнього списку або відсутності парних чисел.

14. Реалізуйте функцію triangle_area(a, b, c), яка обчислює площу трикутника за формулою Герона. Використайте лямбда-функцію для обчислення півпериметра. Забезпечте обробку помилок, якщо сторони не утворюють трикутник або введено некоректні дані.

15. Напишіть функцію is_perfect(n), яка перевіряє, чи є число досконалим (число дорівнює сумі своїх власних дільників). Використайте лямбда-функцію для перевірки, чи є число дільником. Реалізуйте обробку помилок для випадків, коли введено недодатне або неціле значення.

Контрольні питання

1. Що таке функція в Python, як вона визначається та викликається?
2. У чому різниця між параметрами та аргументами, і що відбувається при їхній невідповідності?
3. Як працюють *args та **kwargs, які типи даних вони формують та коли їх варто використовувати?
4. Що таке іменовані аргументи та параметри із значенням за замовчуванням і які їхні переваги?
5. Для чого у функціях використовують оператори return та pass?
6. Чи можна передавати у функції складні типи даних (наприклад, списки) та як це працює?
7. Що таке рекурсія, як вона реалізується та які її особливості?
8. Які існують типи областей видимості (scope) у Python та для чого потрібні ключові слова global і nonlocal?
9. Що таке лямбда-функція, які її обмеження та в яких ситуаціях вона є корисною?
10. Як працює механізм обробки помилок у Python (try, except, else, finally) та для чого використовується raise?

ЛАБОРАТОРНА РОБОТА №8
Тема: «Ітератори, генератори та декоратори»

Мета роботи: Ознайомитися з механізмами роботи ітераторів, генераторів та декораторів у мові програмування Python. Навчитися створювати власні ітератори та генератори для ефективної обробки послідовностей даних, а також застосовувати декоратори для розширення можливостей функцій без зміни їхнього коду.

Завдання до лабораторної роботи

Завдання 1. Створити ітератор **CustomRange**, який приймає два числа *start* і *end*. Об'єкт ітератора повинен підтримувати протокол ітерації, тобто містити методи __iter__() та __next__().

Вимоги:

- ітератор має повертати всі числа між *start* та *end* включно;
- якщо значення вичерпано – викликається StopIteration;
- перевірити роботу у циклі for.

Завдання 2. Створити генератор **reverse_string(s)**, який приймає рядок *s* та повертає його символи у зворотному порядку (один за одним). Перевірити роботу генератора вручну через next() і через цикл for.

Завдання 3. Створити та перевірити декоратор **debug** для мінімум двох різних функцій, який:

- виводить повідомлення: «Викликаю функцію <ім'я функції>»;
- викликає оригінальну функцію;
- після виконання друкує: «Функцію виконано».

Індивідуальні завдання до лабораторної роботи

Завдання 1. Створити ітератор згідно варіанту.

Варіанти:

1. EvenNumbers(n) – генерує всі парні числа від 0 до n.
2. OddNumbers(n) – генерує всі непарні числа від 1 до n.
3. Squares(n) – повертає квадрати чисел від 1 до n.
4. Letters(s) – ітерується по рядку, повертаючи тільки літери (ігноруючи цифри).
5. StepRange(start, end, step) – аналог range, але у вигляді власного ітератора.
6. DivisibleBy(n, k) – числа від 1 до n, що діляться на k.
7. Countdown(n) – відліковує від n до 0.
8. UniqueList(lst) – повертає унікальні елементи списку, зберігаючи порядок.
9. FloatRange(start, end, step) – ітератор для дробових значень.
10. PairIterator(lst) – повертає пари елементів: (lst[0], lst[1]), (lst[2], lst[3]) …
11. PrimeNumbers(n) – генерує всі прості числа від 2 до n.
12. Cubes(n) – повертає куби чисел від 1 до n.
13. NegativeNumbers(n) – генерує всі від'ємні числа від -1 до -n.
14. Vowels(s) – ітерується по рядку, повертаючи тільки голосні літери (a, e, i, o).

15. SquareRoots(n) – повертає квадратні корені чисел від 1 до n.

Завдання 2. Написати функцію-генератор відповідно до варіанту.

Варіанти:

1. gen_factorials(n) – факторіали чисел 1…n.
2. gen_primes(n) – прості числа до n.
3. gen_fibonacci(n) – n чисел Фібоначчі.
4. gen_chunks(s, k) – повертає рядок *s* порціями по k символів.
5. gen_pairs(lst) – сусідні пари елементів списку.
6. gen_reverse_list(lst) – перебирає список у зворотному порядку.
7. gen_words(text) – повертає слова тексту одне за одним.
8. gen_unique(lst) – унікальні елементи послідовності.
9. gen_power(n, p) – числа 1…n у степені p.
10. gen_cycle(lst) – циклічно перебирає елементи (нескінченний генератор).
11. gen_even(n) – парні числа від 0 до n.
12. gen_odd(n) – непарні числа від 1 до n.
13. gen_vowels(s) – голосні літери з рядка s.
14. gen_squares(n) – квадрати чисел від 1 до n.
15. gen_countdown(n) – рахує послідовно від n до 0.

Завдання 3. Створити декоратор відповідно до варіанту та застосувати його до 2 функцій.

Варіанти:

1. time_logger – вимірює час виконання функції.
2. uppercase – перетворює результат функції на верхній регістр.
3. repeat(n) – повторює виконання функції n разів.
4. count_calls – рахує кількість викликів функції.
5. cache – кешує результати функції за її аргументами.
6. validate_positive – не дозволяє передавати від’ємні числа.
7. print_args – виводить усі передані аргументи.
8. check_types(type_) – перевіряє, що всі аргументи мають заданий тип.
9. limit_calls(n) – блокує функцію після n викликів.
10. log_result – записує результат функції у файл.
11. double_result – множить результат функції на 2.
12. negate_result – змінює знак числового результату функції на протилежний.
13. time_limit(seconds) – блокує виконання функції, якщо вона працює довше заданого часу.
14. retry(n) – повторює виконання функції до n разів у разі виникнення виключення.
15. ensure_nonempty – перевіряє, що результат функції не є порожнім (None, '', []) інакше викликає помилку.

Контрольні питання

1. Що таке ітератор в Python і які методи він повинен реалізувати?
2. Яку роль відіграє метод __next__() і коли він викликає StopIteration?
3. Чим відрізняється ітератор від звичайного Iterable-об’єкта (наприклад, списку)?
4. Що таке генератор і як працює оператор yield?
5. Чим генераторна функція відрізняється від звичайної функції?
6. У чому переваги генераторів порівняно зі списковими структурами?
7. Що таке декоратор у Python і як він працює?
8. Для чого використовують вкладену функцію у декораторі?
9. Чому декоратор зазвичай повертає внутрішню функцію, а не результат виконання?
10. Наведіть приклади практичних задач, де корисно застосовувати декоратори (наприклад, логування, кешування, обмеження викликів).

ЛАБОРАТОРНА РОБОТА №9
Тема: «Робота з файлами та структурованими даними»

Мета роботи: Ознайомитися з CRUD-операціями з файлами у Python. Навчитися працювати зі структурованими даними (списки, словники, CSV, JSON, бінарні файли). Отримати навички обробки даних та збереження їх у різних форматах.

Завдання до лабораторної роботи

Завдання 1. Написати програму для відкриття та читання txt-файлів за допомогою якої можна буде:

- створити текстовий файл demo.txt з кількома рядками тексту;
- відкрити файл у режимі читання та вивести весь вміст на екран;
- зчитати перший рядок файлу окремо, використовуючи метод readline();
- зчитати файл рядок за рядком у циклі та вивести кожен рядок на екран.

Завдання 2. Написати програму для запису та редагування txt-файлів за допомогою якої можна буде:

- відкрити файл demo.txt у режимі дописування (a) та додати ще один рядок тексту;
- відкрити файл у режимі перезапису (w) та записати новий вміст, повністю замінивши старий;
- створити новий файл newfile.txt у режимі x;
- закрити усі файли та перевірити їх вміст, відкривши у режимі читання.

Завдання 3. Написати програму для видалення файлів та папок за допомогою якої можна буде:

- перевірити, чи існує файл demo.txt і видалити його, використовуючи модуль os;
- створити папку myfolder та видалити її за допомогою os.rmdir();
- спробувати видалити файл, який не існує, та обробити помилку за допомогою перевірки os.path.exists().

Індивідуальні завдання до лабораторної роботи

Завдання 1. Створити CSV-файл із даними студентів (ім’я, вік, оцінка) та відповідно до індивідуального варіанту виконати наступні операції:

- зчитати файл та вивести усі дані;
- відфільтрувати студентів з оцінкою > 80;
- додати нового студента у кінець файлу;
- змінити оцінку певного студента та зберегти результат.

Варіант	Дані для CSV (Name, Age, Grade)
1	Anton, 21, 95; Maria, 22, 88; Oleg, 20, 76
2	Lina, 23, 90; Ivan, 21, 82; Olga, 22, 78
3	Peter, 20, 85; Anna, 21, 92; Mark, 22, 70

4	Sofia, 23, 88; Alex, 20, 77; Nina, 21, 91
5	John, 22, 84; Kate, 23, 95; Leo, 21, 68
6	Mike, 20, 89; Emma, 21, 76; Tom, 22, 80
7	Alice, 23, 92; Bob, 22, 81; Carol, 21, 79
8	Dave, 20, 87; Eve, 21, 90; Frank, 22, 73
9	Grace, 23, 85; Henry, 22, 88; Ivy, 21, 91
10	Jack, 20, 78; Jill, 21, 83; Ken, 22, 96
11	Liam, 22, 91; Emma, 23, 87; Noah, 21, 79
12	Olivia, 20, 94; Mason, 22, 82; Sophia, 21, 88
13	Ethan, 23, 85; Ava, 21, 90; Logan, 22, 76
14	Isabella, 22, 92; Lucas, 20, 78; Mia, 21, 84
15	James, 21, 89; Amelia, 22, 95; Benjamin, 23, 80

Завдання 2. Створити JSON-файл з інформацією про користувачів (ім'я, вік, список курсів) та відповідно до індивідуального варіанту виконати наступні операції:

- зчитати JSON-файл у Python як словник;
- вивести усіх користувачів, які вивчають певний курс (наприклад, CS);
- додати нового користувача до структури та зберегти файл;
- вивести усі унікальні курси, які відвідують користувачі.

Варіант	**Дані для JSON (name, age, courses)**
1	Anton, 21, ["Math", "CS"]; Maria, 22, ["Physics", "CS"]; Oleg, 20, ["Math", "Biology"]
2	Lina, 23, ["Art", "Math"]; Ivan, 21, ["CS", "Physics"]; Olga, 22, ["Biology", "Math"]
3	Peter, 20, ["Math", "CS"]; Anna, 21, ["Art", "CS"]; Mark, 22, ["Physics", "Math"]
4	Sofia, 23, ["CS", "Math"]; Alex, 20, ["Biology", "Art"]; Nina, 21, ["Math", "Physics"]
5	John, 22, ["CS", "Physics"]; Kate, 23, ["Math", "Art"]; Leo, 21, ["Biology", "CS"]
6	Mike, 20, ["Math", "CS"]; Emma, 21, ["Art", "Biology"]; Tom, 22, ["Physics", "CS"]
7	Alice, 23, ["CS", "Math"]; Bob, 22, ["Art", "Physics"]; Carol, 21, ["Math", "Biology"]
8	Dave, 20, ["Math", "CS"]; Eve, 21, ["Physics", "Art"]; Frank, 22, ["CS", "Math"]
9	Grace, 23, ["Math", "Biology"]; Henry, 22, ["CS", "Physics"]; Ivy, 21, ["Art", "Math"]
10	Jack, 20, ["CS", "Art"]; Jill, 21, ["Math", "Physics"]; Ken, 22, ["Biology", "CS"]
11	Liam, 22, ["Math", "Physics"]; Emma, 23, ["CS", "Art"]; Noah, 21, ["Biology", "Math"]
12	Olivia, 20, ["CS", "Math"]; Mason, 22, ["Physics", "Art"]; Sophia, 21, ["Math", "CS"]
13	Ethan, 23, ["Art", "CS"]; Ava, 21, ["Math", "Biology"]; Logan, 22, ["Physics", "CS"]
14	Isabella, 22, ["CS", "Math"]; Lucas, 20, ["Biology", "Physics"]; Mia, 21, ["Art", "Math"]
15	James, 21, ["Math", "CS"]; Amelia, 22, ["Physics", "Art"]; Benjamin, 23, ["Biology", "CS"]

Завдання 3. Створити бінарний файл із тестовими даними та відповідно до індивідуального варіанту виконати наступні операції:

- записати бінарні дані у файл (wb);
- зчитати дані з файлу у бінарному режимі (rb) та вивести на екран;
- змінити певну частину бінарних даних та перезаписати файл;
- підрахувати кількість символів/байтів у файлі після редагування.

Варіант	**Бінарні дані**
1	b"Data for test 1"
2	b"Binary content example 2"
3	b"Test file data 3"

4	b"Sample binary info 4"
5	b"Python binary test 5"
6	b"Example data 6"
7	b"Binary test 7"
8	b"File data example 8"
9	b"Content for binary 9"
10	b"Test binary file 10"
11	b"Binary example 11"
12	b"Test data 12"
13	b"Sample binary 13"
14	b"Python test 14"
15	b"Binary content 15"

Контрольні питання

1. Яка функція в Python використовується для відкриття файлів? Поясніть її основні параметри та призначення кожного з них.
2. Які існують режими відкриття файлів у Python? Поясніть різницю між режимами "r", "a", "w" та "x".
3. Що таке текстовий ("t") та бінарний ("b") режими роботи з файлами? Наведіть приклади, коли використовується кожен із режимів.
4. Як відкрити файл у Python та прочитати його вміст рядок за рядком? Наведіть приклад коду з використанням циклу.
5. Для чого використовується оператор with при роботі з файлами? Яка його перевага порівняно з класичним open() та close()?
6. Як додати новий рядок до існуючого файлу без втрати старих даних? Наведіть приклад з поясненням параметру відкриття файлу.
7. Як повністю перезаписати файл новим вмістом? Поясніть, чим це відрізняється від дописування.
8. Які формати структурованих даних найчастіше використовуються у файлах? Поясніть, для яких задач підходять формати CSV, JSON та бінарні файли.
9. Як прочитати та обробити JSON-файл у Python? Наведіть приклад фільтрації даних з JSON.
10. Як видалити файл або папку у Python та як перевірити, чи існує файл перед видаленням? Наведіть приклад з використанням модуля os.

ЛАБОРАТОРНА РОБОТА №10
Тема: «Використання модулів та бібліотек. Створення власного модуля»

Мета роботи: Ознайомитися з концепцією модулів та бібліотек у мові програмування Python. Навчитися створювати власні модулі, імпортувати їх у програми, використовувати функції та змінні модулів, а також застосовувати вбудовані та сторонні бібліотеки для підвищення ефективності та структурування коду.

Завдання до лабораторної роботи

Завдання 1. Використати вбудований модуль math в програмі.

1. Імпортуйте модуль *math*.
2. Обчисліть:
   - квадратний корінь з введеного користувачем числа;
   - значення числа π (*math.pi*);
   - косинус кута (в радіанах).
3. Виведіть результати на екран.

Завдання 2. Напишіть власний програмний модуль.

1. Створіть файл *mymodule.py*, у якому опишіть:
   - функцію greet_user(name);
   - словник user_data з інформацією про користувача.
2. Імпортуйте модуль:
   - один раз через import mymodule;
   - другий раз через from mymodule import user_data;
   - третій раз через import mymodule as mm.
3. Виведіть дані зі словника та викличте функцію привітання.

Завдання 3. Використайте функції з власного модуля.

1. Створіть файл *data_utils.py*, який містить функції:
   - calculate_sum(numbers);
   - calculate_average(numbers).

2. У головній програмі імпортуйте функції: from data_utils import calculate_sum, calculate_average.

3. Для списку nums = [10, 20, 30, 40] виведіть суму та середнє значення.

4. За допомогою dir() відобразіть усі доступні елементи модуля data_utils.

Індивідуальні завдання до лабораторної роботи

Завдання 1. Використовуючи *один стандартний модуль Python (math, random, datetime, os, statistics, string, platform, sys, time, json)*, виконайте завдання відповідно до свого варіанта.

Варіанти:

1. Обчислити площу кола за допомогою math.
2. Згенерувати 10 випадкових чисел та знайти максимум (random).

3. Вивести поточну дату у форматі YYYY-MM-DD (datetime).
4. Вивести список файлів у поточній директорії (os).
5. Обчислити середнє та медіану набору чисел (statistics).
6. Вивести всі ASCII-символи з string.ascii_letters.
7. Вивести інформацію про операційну систему через platform.
8. Вивести шлях до Python-інтерпретатора (sys.executable).
9. Вивести час виконання циклу з 1 млн ітерацій (time).
10. Створити словник, перетворити його у JSON-рядок та назад у словник (json).
11. Обчислити квадратний корінь та степінь числа за допомогою math.
12. Згенерувати випадковий список елементів і перемішати його (random.shuffle).
13. Вивести поточний час у форматі HH:MM:SS (datetime).
14. Створити нову папку та перевірити її існування (os).
15. Обчислити стандартне відхилення та дисперсію набору чисел (statistics).

Завдання 2. Створіть модуль *module_varX.py* (де *X – номер варіанта*), який міститиме: функцію, змінну та (за потреби) структуру даних.

У головному файлі необхідно:

- імпортувати модуль;
- викликати функцію;
- вивести значення змінної.

Варіанти:

1. Функція для обчислення факторіалу; змінна – ваше ім’я.
2. Функція для перевірки простого числа; змінна – список цілих чисел.
3. Функція для пошуку максимального елемента; змінна – кортеж.
4. Функція для обчислення площі трикутника; змінна – координати точок.
5. Функція для переводу °C → °F; змінна – температура.
6. Функція для шифрування рядка (простий шифр Цезаря); змінна – ключ.
7. Функція для обчислення суми цифр числа; змінна – саме число.
8. Функція для форматування ПІБ; змінна – словник з полями name, surname.
9. Функція для пошуку слова в тексті; змінна – рядок-текст.
10. Функція для перевірки пароля (довжина та наявність цифр); змінна – пароль.
11. Функція для обчислення середнього арифметичного списку; змінна – список чисел.
12. Функція для знаходження всіх парних чисел у списку; змінна – список цілих чисел.
13. Функція для переведення градусів Фаренгейта → Цельсій; змінна – температура.
14. Функція для реверсу рядка; змінна – рядок.
15. Функція для перевірки, чи є число паліндромом; змінна – саме число.

Завдання 3. Встановіть відповідну бібліотеку через pip у віртуальне середовище venv, імпортуйте її в програму, продемонструйте приклад її використання.

Варіанти бібліотек:

1. numpy – створити масив і обчислити його суму.
2. pandas – створити DataFrame та вивести перші рядки.
3. matplotlib – побудувати графік лінії.
4. requests – виконати HTTP-запит до сайту та вивести статус-код.
5. beautifulsoup4 – розпарсити HTML-рядок і знайти елемент.
6. faker – згенерувати випадкове ім’я та адресу.
7. pyttsx3 – виконати синтез голосу.
8. qrcode – створити QR-код із тексту.
9. opencv-python – прочитати та відобразити зображення.
10. Pillow – відкрити зображення та змінити його розмір.
11. sympy – створити символічне рівняння та обчислити його розв’язок.
12. pygame – створити вікно та намалювати просту фігуру.
13. plotly – побудувати інтерактивний графік.
14. sqlalchemy – створити просту базу даних в пам’яті та виконати запит.
15. openpyxl – створити Excel-файл та записати дані у комірки.

Контрольні питання

1. Що таке модуль у Python і які об’єкти він може містити?
2. У чому полягає різниця між модулем та бібліотекою? Наведіть приклади.
3. Як імпортувати весь модуль, окрему функцію або змінну з модуля у програму? Наведіть приклади.
4. Що таке імпорт з присвоєнням аліаса і коли його доцільно використовувати?
5. Як створити власний модуль у Python? Які правила назви файлу модуля?
6. Як використовувати функції та змінні власного модуля у головній програмі? Наведіть приклад.
7. Яке призначення функції dir() при роботі з модулями? Наведіть приклад використання.
8. Що таке pip і які основні команди для роботи з бібліотеками він надає?
9. Як створити і активувати віртуальне середовище у Python та для чого це потрібно?
10. Які переваги використання модулів, бібліотек та pip для розробки програм? Наведіть мінімум три.

ЛАБОРАТОРНА РОБОТА №11
Тема: «Тестування та профілювання коду»

Мета роботи: Ознайомитися з методами тестування та профілювання коду у мові програмування Python. Навчитися створювати та запускати модульні та автоматизовані тести для перевірки коректності роботи функцій, застосовувати підхід TDD для розробки програмного забезпечення, а також аналізувати продуктивність програм за допомогою інструментів профілювання (time, timeit, cProfile, memory_profiler) для виявлення та оптимізації вузьких місць у коді.

Завдання до лабораторної роботи

1. Провести модульне тестування за допомогою бібліотеки unittest:
   - створити файл calculator.py з функціями: add(a, b), divide(a, b), is_even(n);
   - написати тести з використанням unittest для перевірки всіх функцій (усі можливі випадки: позитивні числа, нуль, від'ємні числа, поділ на нуль);
   - запустити тести і проаналізувати результати.
2. Провести автоматизоване тестування за допомогою бібліотеки pytest:
   - створити файл test_calculator_pytest.py;
   - написати тести з використанням pytest, включно з параметризованими тестами для функції is_even;
   - запустити pytest і відобразити детальні результати (pytest -v).
3. Провести профілювання простого коду:
   - створити функцію сортування (наприклад, bubble_sort і built_in_sort);
   - використати модулі time, timeit, cProfile для вимірювання часу виконання;
   - проаналізувати, яка функція швидше, та знайти вузькі місця у виконанні.

Індивідуальні завдання до лабораторної роботи

Завдання 1. Провести модульне тестування функцій з використанням unittest. Написати тести для відповідної функції.

Завдання 2. Провести автоматизоване тестування з pytest. Написати параметризовані тести для відповідної функції.

Завдання 3. Провести профілювання коду. Для відповідної функції використати наступні інструменти:

- time або timeit для вимірювання часу;
- cProfile для аналізу викликів функцій;
- опційно memory_profiler для вимірювання пам'яті.

Рекомендації:

- перед профілюванням переконатися, що код проходить усі тести;
- використовувати зрозумілі та короткі вхідні дані для наочності результатів;
- після оптимізації коду повторно запустити тести, щоб переконатися в коректності роботи;
- для кращого аналізу результатів профілювання сортувати їх за часом виконання (cumtime) або кількістю викликів (ncalls).

Варіанти:

№	Функція для тестування	Що повинна виконувати функція
1.	add(a, b)	Повертає суму двох чисел a та b
2.	divide(a, b)	Повертає результат ділення a на b; викликає помилку при діленні на нуль
3.	factorial(n)	Обчислює факторіал числа n (n!)
4.	is_prime(n)	Перевіряє, чи є число n простим
5.	fibonacci(n)	Повертає n-е число Фібоначчі
6.	power(a, b)	Повертає a у степені b
7.	is_even(n)	Повертає True, якщо число n парне, і False, якщо непарне
8.	gcd(a, b)	Знаходить найбільший спільний дільник чисел a та b
9.	lcm(a, b)	Знаходить найменше спільне кратне чисел a та b
10.	sum_of_list(lst)	Повертає суму всіх елементів списку lst
11.	max_in_list(lst)	Знаходить максимальний елемент у списку lst
12.	min_in_list(lst)	Знаходить мінімальний елемент у списку lst.
13.	factorial_recursive(n)	Обчислює факторіал числа n рекурсивно.
14.	reverse_string(s)	Повертає рядок s у зворотному порядку.
15.	count_vowels(s)	Рахує кількість голосних у рядку s.

Контрольні питання

1. Що таке тестування програмного забезпечення і яка його основна мета?
2. Назвіть основні завдання тестування коду.
3. Які види тестування розрізняють за рівнем виконання? Наведіть приклади.
4. У чому полягає різниця між ручним та автоматизованим тестуванням?
5. Що таке тест-кейс і які його основні компоненти?
6. Які основні методи модуля unittest використовуються для перевірки коректності роботи функцій?
7. Які переваги має бібліотека pytest порівняно з unittest?
8. Що таке TDD (Test-Driven Development)?
9. Що таке профілювання коду і які основні параметри програми можна при цьому аналізувати?
10. Назвіть основні інструменти профілювання в Python і поясніть, для чого кожен з них використовується (time, timeit, cProfile, memory_profiler).

ЛАБОРАТОРНА РОБОТА №12
Тема: «Алгоритмічна складність. Пошук і сортування. Рекурсивні обчислення»

Мета роботи: Ознайомитися з алгоритмічною складністю та методами пошуку, сортування і рекурсивних обчислень у Python. Навчитися реалізовувати лінійний пошук, бінарний пошук (ітеративний та рекурсивний), сортування вибором, бульбашкове сортування, сортування вставками, швидке сортування (Quick Sort) та сортування злиттям (Merge Sort). Засвоїти поняття рекурсії, будувати рекурсивні функції для обчислення факторіала, чисел Фібоначчі та інших рекурсивних задач, а також порівнювати ефективність рекурсивного та ітеративного підходів.

Завдання до лабораторної роботи

Завдання 1. Провести аналіз часової складності алгоритму відповідно до індивідуального варіанту:

- реалізувати алгоритм пошуку максимального елемента в кожному рядку квадратної матриці A[n][n];
- додати лічильник базових операцій (порівнянь);
- провести експерименти для різних значень n;
- пояснити, чому алгоритм має часову складність $O(n^2)$.

В кожному варіанті провести експерименти з указаними n (мінімум 3-5 запусків на кожне значення).

Варіанти:

1. n = 50, 100, 150, 200.
2. n = 64, 128, 256, 512.
3. n = 25, 50, 75, 100.
4. n = 30, 60, 120, 240.
5. n = 10, 100, 500, 1000.
6. n = 40, 80, 160, 320.
7. n = 70, 140, 210, 280.
8. n = 90, 180, 360, 720.
9. n = 15, 45, 90, 180.
10. n = 32, 96, 192, 384.
11. n = 20, 40, 80, 160.
12. n = 35, 70, 140, 280.
13. n = 12, 24, 48, 96.
14. n = 18, 36, 72, 144.
15. n = 50, 150, 300, 600.

Завдання 2. Провести тестування алгоритмів пошуку:

- реалізувати лінійний пошук;
- реалізувати бінарний пошук (ітеративний);
- підготувати тестові набори даних:
  - відсортований масив;
  - невідсортований масив;

- випадок відсутності шуканого елемента.

– перевірити коректність роботи алгоритмів і порівняти час виконання.

Варіанти:

1. Масив із 50 елементів, числа в діапазоні [0; 100], шуканий = 37.
2. Масив із 100 елементів, діапазон [−50; 50], шуканий = −12.
3. Масив із 200 елементів, діапазон [10; 500], шуканий = 250.
4. Масив із 32 елементів, діапазон [0; 32], шуканий = 17.
5. Масив із 75 елементів, діапазон [0; 300], шуканий = 999.
6. Масив із 120 елементів, діапазон [−100; 100], шуканий = 0.
7. Масив із 64 елементів, діапазон [1; 1000], шуканий = 1.
8. Масив із 90 елементів, діапазон [5; 5000], шуканий = 5000.
9. Масив із 150 елементів, діапазон [2; 2000], шуканий = 2.
10. Масив із 300 елементів, діапазон [0; 10], шуканий = 7.
11. Масив із 40 елементів, числа в діапазоні [0; 500], шуканий = 250.
12. Масив із 60 елементів, числа в діапазоні [−200; 200], шуканий = −50.
13. Масив із 80 елементів, числа в діапазоні [10; 1000], шуканий = 750.
14. Масив із 100 елементів, числа в діапазоні [0; 300], шуканий = 150.
15. Масив із 150 елементів, числа в діапазоні [1; 100], шуканий = 100.

Завдання 3. Провести аналіз ефективності сортування.

1. Написати програми для таких алгоритми сортування:
   - сортування вибором;
   - бульбашкове сортування;
   - сортування вставками.

2. Провести профілювання часу виконання сортування для типів масивів згідно з індивідуальним номером варіанту.

3. Зробити висновки щодо впливу асимптотичної складності та щодо доцільності використання кожного алгоритму сортування в різних умовах.

Варіанти:

№	Параметри
1.	– 100 елементів, випадкові числа [0; 1000]; – 6000 елементів, випадкові числа [0; 10^6]; – 200 елементів, вже відсортований масив; – 200 елементів, відсортований за спаданням.
2.	– 250 елементів, випадкові числа з діапазону [-10; 10]; – 8000 елементів, випадкові числа [0; 10^5]; – 300 елементів – 1/3 значень однакові; – 300 елементів – перша половина впорядкована за зростанням, друга – за спаданням.
3.	– 120 елементів, випадкові числа [0; 500]; – 7000 елементів, випадкові числа [0; 10^4]; – 150 елементів – частково відсортований (20% випадкові); – 150 елементів – арифметична прогресія.
4.	– 80 елементів, випадкові числа [-100; 100];

	– 9000 елементів, випадкові числа [0; 10^6]; – 100 елементів – всі числа різні; – 100 елементів – багато дубльованих значень.
5.	– 300 елементів, випадкові [0; 500]; – 10 000 елементів, випадкові [0; 10^7]; – 400 елементів – від'ємні та додатні; – 400 елементів – відсортований за спаданням.
6.	– 200 елементів, випадкові [0; 50]; – 6000 елементів, випадкові [-10^5; 10^5]; – 220 елементів – частково відсортований; – 220 елементів – 1/3 рівних значень.
7.	– 450 елементів, випадкові [0; 900]; – 5500 елементів, випадкові [0; 10^3]; – 500 елементів – зростання/спадання; – 500 елементів – арифметична прогресія.
8.	– 130 елементів, випадкові [−20; 20]; – 7000 елементів, випадкові [0; 10^2]; – 200 елементів – відсортований за зростанням; – 200 елементів – відсортований за спаданням.
9.	– 400 елементів, випадкові [0; 2000]; – 6500 елементів, випадкові [0; 10^6]; – 350 елементів – 1/3 однакових значень; – 350 елементів – частково відсортований.
10.	– 500 елементів, випадкові [0; 1000]; – 8000 елементів, випадкові [0; 10^5]; – 300 елементів – всі значення різні; – 300 елементів – багато повторів.
11.	– 150 елементів, випадкові числа [0; 1000]; – 9000 елементів, випадкові числа [0; 10^6]; – 200 елементів – частково відсортований (половина випадкові); – 200 елементів – відсортований за спаданням.
12.	– 180 елементів, випадкові числа [-500; 500]; – 10 000 елементів, випадкові числа [0; 10^5]; – 250 елементів – 1/4 однакових значень; – 250 елементів – перша половина впорядкована за зростанням, друга – випадкова.
13.	– 100 елементів, випадкові числа [0; 200]; – 7500 елементів, випадкові числа [0; 10^3]; – 150 елементів – арифметична прогресія; – 150 елементів – всі числа різні.
14.	– 220 елементів, випадкові числа [−100; 100]; – 8500 елементів, випадкові числа [0; 10^5]; – 300 елементів – частково відсортований (30% випадкові); – 300 елементів – багато дубльованих значень.

15.	– 350 елементів, випадкові числа [0; 5000]; – 12 000 елементів, випадкові числа [0; 10^7]; – 400 елементів – від'ємні та додатні; – 400 елементів – відсортований за спаданням.

Завдання 4. Реалізувати обчислення рекурсивно, відповідно до індивідуального варіанту.

1. Обчислити факторіал числа n (де n вводиться користувачем).
2. Знайти добуток усіх непарних чисел до n.
3. Обчислити подвійний факторіал $n!!$ $(n \cdot (n-2) \cdot (n-4)\ldots)$.
4. Рекурсивно обчислити $n! \,/\, (n-k)!$
5. Рекурсивно обчислити добуток чисел від a до b.
6. Створити таблицю значень факторіалу для $n = 1\ldots10$.
7. Порівняти час виконання рекурсивного та ітеративного варіанту на великих значеннях n.
8. Рекурсивно підрахувати кількість цифр у $n!$ (не саме число).
9. Рекурсивно вивести всі множники факторіалу (наприклад, $5! : 5 \cdot 4 \cdot 3 \cdot 2 \cdot 1$).
10. Рекурсивно реалізувати обчислення факторіалу з мемоізацією.

Варіанти:

1. n = 10, k = 3, a = 2, b = 8.
2. n = 12, k = 5, a = 1, b = 6.
3. n = 15, k = 4, a = 3, b = 10.
4. n = 20, k = 7, a = 5, b = 12.
5. n = 25, k = 6, a = 4, b = 18.
6. n = 30, k = 8, a = 6, b = 20.
7. n = 35, k = 10, a = 2, b = 15.
8. n = 40, k = 12, a = 7, b = 21.
9. n = 50, k = 15, a = 10, b = 25.
10. n = 60, k = 20, a = 8, b = 30.
11. n = 18, k = 6, a = 3, b = 12.
12. n = 22, k = 9, a = 5, b = 15.
13. n = 28, k = 11, a = 7, b = 21.
14. n = 32, k = 14, a = 8, b = 24.
15. n = 45, k = 18, a = 10, b = 30.

Контрольні питання

1. Що таке алгоритмічна складність і які ресурси вона оцінює?
2. У чому відмінність між часовою та просторовою складністю алгоритму?
3. Що означає твердження, що алгоритм є поліноміальним? Наведіть приклад.
4. Чим поліноміальні алгоритми відрізняються від експоненційних з погляду виконання на великих даних?
5. Для чого використовується нотація «велике О» (O)? Як вона інтерпретується?
6. Що означає запис складності $O(n^2)$ для двовимірного масиву розміру?
7. Що таке нотація «мале о» (o) і як вона співвідноситься з функцією $g(N)$?

8. Які особливості та переваги лінійного пошуку? Яка його часова складність у найгіршому випадку?
9. Чому бінарний пошук можна використовувати лише на відсортованому масиві?
10. У чому полягає принцип роботи бінарного пошуку? Яка його асимптотична складність?
11. Порівняйте лінійний та бінарний пошук: у яких випадках який з них доцільніше застосовувати?
12. У чому полягає ідея алгоритму бульбашкового сортування? Яка його складність у середньому випадку?
13. Чому бульбашкове сортування вважається неефективним для великих масивів?
14. Як працює сортування вставками та в яких ситуаціях воно є ефективним?
15. Яка середня складність швидкого сортування і в яких випадках його складність погіршується до $O(n^2)$?
16. Який принцип роботи сортування злиттям і чому воно потребує додаткової пам’яті?
17. Порівняйте Quick Sort та Merge Sort за швидкістю, стабільністю та використовуваною пам’яттю.
18. Що таке рекурсія та в яких задачах її доцільно використовувати?
19. Чим рекурсивний алгоритм відрізняється від ітеративного? Наведіть приклад задачі, яку зручно реалізувати рекурсивно.
20. Які потенційні недоліки рекурсивних обчислень?

ВИКОРИСТАНА ЛІТЕРАТУРА

1. Mark Lutz. Learning Python 6th Edition, O'Reilly, 2025. 1169 p.
2. Lubanovic B. Introducing Python: Modern Computing in Simple Packages 2nd ed. O'Reilly, 2019. 627 p.
3. Пол Беррі. Head First. Python. Видавництво "Фабула", 2021. 624 с.
4. Eric Matthes. Python Crash Course, 3rd Edition. No Starch Press, 2023. 552 p.
5. Luciano Ramalho. Fluent Python. 2nd Edition. O'Reilly Media, 2022. 1014 p.
6. Allen B. Downey. Think Python: How to Think Like a Computer Scientist. 2nd Edition. O'Reilly Media, 2015. 292 p.
7. Brett Slatkin. Effective Python: 90 Specific Ways to Write Better Python. 2nd Edition. Addison-Wesley, 2019. 480 p.
8. Офіційний сайт Python. URL: https://www.python.org/ (дата звернення: 07.01.2026).
9. The Best Way to Learn Python. URL: https://code.tutsplus.com/the-best-way-to-learn-python--net-26288a (дата звернення: 07.01.2026).
10. Learn Python 3. URL: https://www.codecademy.com/learn/learn-python-3 (дата звернення: 07.01.2026).
11. Система підтримки дистанційного навчання ФІТІС. URL: https://moodle.chdtu.edu.ua (дата звернення: 07.01.2026).
12. Python Essentials 1 by Cisco. URL: https://www.netacad.com/courses/python-essentials-1 (дата звернення: 07.01.2026).
13. W3Schools Python Tutorial https://www.w3schools.com/python/default.asp
14. Python Essentials 2 by Cisco. URL: https://www.netacad.com/courses/python-essentials-2 (дата звернення: 07.01.2026).
15. Sorting Algorithms Animations. URL: https://www.toptal.com/developers/sorting-algorithms (дата звернення: 07.01.2026).
16. Sorting Algorithms Explained Visually. URL: https://www.youtube.com/watch?v=RfXt_qHDEPw (дата звернення: 07.01.2026).
17. Урок 38. Java Програмування - Сортування масивів (Українською) URL: https://www.youtube.com/watch?v=Gp53WF5tSsw (дата звернення: 07.01.2026).
18. TimSort – GeeksforGeeks. URL: https://www.geeksforgeeks.org/dsa/timsort/ (дата звернення: 07.01.2026).
19. Sorting Algorithm Cheat Sheet. URL: https://www.interviewcake.com/sorting-algorithm-cheat-sheet (дата звернення: 07.01.2026).
20. Sorting algorithm. URL: https://en.wikipedia.org/wiki/Sorting_algorithm (дата звернення: 07.01.2026).
21. Timsort. URL: https://en.wikipedia.org/wiki/Timsort (дата звернення: 07.01.2026).
22. Python Now Uses Powersort. URL: https://www.i-programmer.info/news/216-python/15954-python-now-uses-powersort.html (дата звернення: 07.01.2026).
23. ДСТУ 3008:2015. Звіти у сфері науки і техніки. Структура та правила оформлювання / Нац. Стандарт України. Вид. офіц. [На заміну ДСТУ 3008-95; чинний від 2017-07-01]. Київ : ДП «УкрНДНЦ», 2016. 31 с. (Інформація та документація).

ДОДАТОК А

Міністерство освіти і науки України
Черкаський державний технологічний університет

Факультет інформаційних технологій і систем

Кафедра комп'ютерних наук та системного аналізу

Дисципліна:
«Алгоритмізація та програмування»

З В І Т

з лабораторної роботи № 1
Тема: «Встановлення середовища програмування Python. Синтаксис Python. Основні принципи та правила створення, компіляції та відлагодження програм»

студента групи КН-2401
спеціальності F3 «Комп'ютерні науки»

Петренка Дмитра Олександровича

_____ _____
(Дата) (Підпис студента)

Оцінка _____

Перевірено _____
(Дата)
Викладач _____ / Максимов А.Є. /
(Підпис) (Прізвище та ініціали)

Черкаси – 2026 р.

З В І Т

про виконання завдань до лабораторної роботи 1

Тема: «Встановлення середовища програмування Python. Синтаксис Python. Основні принципи та правила створення, компіляції та відлагодження програм»

Мета роботи: Ознайомитись із середовищем програмування Python, процесом його встановлення та налаштування. Сформувати базові навички побудови алгоритмів для розв'язання типових задач комп'ютерних наук, а також засвоїти основи структурного та алгоритмічного мислення.

Протокол розв'язування завдань

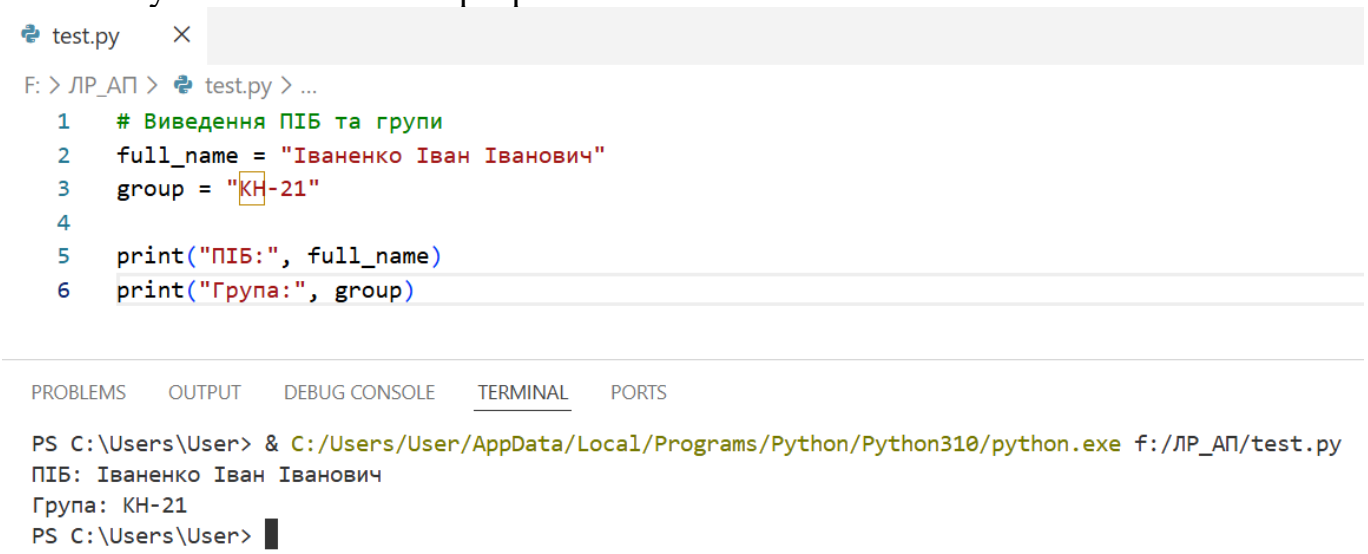
Завдання 1. Вивести на екран Ваше ПІБ та групу, в якій навчаєтесь.

Код програми:

```
# Виведення ПІБ та групи
full_name = "Іваненко Іван Іванович"
group = "КН-25"

print("ПІБ:", full_name)
print("Група:", group)
```

Результат виконання програми:



Висновки

В процесі виконання лабораторної роботи проведено ознайомлення із середовищем програмування Python, процесом його встановлення та налаштування.

В результаті виконання роботи мною успішно реалізовані поставлені завдання, що дозволило набути базових навичок побудови алгоритмів для розв'язання типових задач комп'ютерних наук. Це сприяло закріпленню основ структурного та алгоритмічного мислення.

ЗМІСТ