Національний технічний університет України
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

Емблема кафедри (за наявності)

Кафедра обчислювальної техніки

ПРОГРАМУВАННЯ-1

Робоча програма навчальної дисципліни (Силабус)

Реквізити навчальної дисципліни

Рівень вищої освіти	***Перший (бакалаврський)***
Галузь знань	*12 Інформаційні технології*
Спеціальність	*123 Комп'ютерна інженерія*
Освітня програма	*Комп'ютерні системи та мережі*
Статус дисципліни	*Нормативна*
Форма навчання	*Очна (денна), заочна*
Рік підготовки, семестр	*1 курс, осінній семестр*
Обсяг дисципліни	*7 кредитів, 210 годин*
Семестровий контроль/ контрольні заходи	*Екзамен*
Розклад занять	*Лекції – 72 години. Лабораторні роботи - 36 годин. Самостійна робота студентів – 72 години*
Мова викладання	*Українська*
Інформація про керівника курсу / викладачів	Лектор(очна): д.т.н, професор, Новотарський Михайло Анатолійович Лектор(заочна) асистент Пономаренко Артем Миколайович novotar@gmail.com Лабораторні: асистент, Пономаренко Артем Миколайович ponomarenkokpi@gmail.com
Розміщення курсу	https://classroom.google.com/c/NDE3ODIzNzU2MDk1?cjc=n2nef3n

Програма навчальної дисципліни

1. Опис навчальної дисципліни, її мета, предмет вивчання та результати навчання

Метою навчальної дисципліни «Програмування-1» є оволодіння основами сучасних мов та технологій програмування, набуття умінь та досвіду зі створення прикладних програм, програмних комплексів з врахуванням особливостей сучасних мов та технологій програмування при вирішенні задач у науковій, інженерно-технічній та економічній сферах діяльності та сприяння розвитку логічного та аналітичного мислення студентів.

Основні завдання навчальної дисципліни полягають:

- **у вивченні** основних етапів процесу проєктування програмного забезпечення з використанням агрегатора "Anaconda", що включає IDE "PyCharm" та "Jupyter notebook"; типових алгоритмічних конструкцій, які є характерними для скриптової мови програмування Python; принципів процедурного, структурного та об'єктно-орієнтованого програмування в Python; особливостей застосування сучасних програмних засобів на базі мови програмування Python, призначених для вирішення економічних задач; базових типів даних, операторів управління програмою, правил роботи з функціями методами типів мови програмування Python; системи вводу-виводу та основних принципів роботи з файлами в Python; правил роботи з регулярними виразами та шаблонами; основних підходів, що використовуються при створенні графічного інтерфейсу користувача для програм на Python;

- **у оволодінні** основами програмування, тобто умінні створювати програми мовою Python, забезпечуючи розв'язування задач математичного характеру; створення і обробку складних структур даних; найпростішу обробку файлів; використання стандартних функцій та функцій користувача; використання методів стандартних типів мови Python; використання раніше складених програм і здійснення супроводу програм, внесення змін в програму, виконання налаштування програм за допомогою вбудованих інструментальних засобів;

- **у набутті** досвіду з володіння методами та технологіями програмування алгоритмічною мовою Python, створення та використання структур даних, які дозволяють ефективно розв'язувати практичні задачі, створення програм з графічним інтерфейсом користувача (GUI).

Таке поєднання теоретичних та практичних знань та вмінь сприяє як достатньому оволодінню технологією роботи з конкретним середовищем розробки, так і полегшенню освоєння нових засобів програмування та переходу в подальшому на нові середовища та операційні системи.

На початку вивчення дисципліни кожен студент має бути ознайомлений з програмою дисципліни і формами організації навчання, а також з усіма видами контролю та методикою оцінювання знань. Тематичний план дисципліни включає змістовний модуль «Основи програмування», який складається з окремих самостійних блоків, що логічно поєднані в кілька навчальних елементів дисципліни за змістом та взаємозв'язками.

Основні компетентності

Здобувачі ступеня бакалавра після засвоєння нормативної дисципліни «Програмування - 1» мають набути таких:

загальних компетентностей:

ЗК1. Здатність до абстрактного мислення, аналізу та синтезу.

ЗК7. Вміння виявляти, ставити та вирішувати проблеми.

ЗК8. Здатність працювати в команді.

фахових компетентностей:

ФК2. Здатність використовувати сучасні методи і мови програмування для розроблення алгоритмічного та програмного забезпечення

ФК16. Здатність проектувати, розробляти, впроваджувати та обслуговувати програмноапаратне забезпечення для високопродуктивних паралельних та розподілених комп'ютерні систем та їх складових на сучасній елементній базі, зокрема, з використанням ПЛІС і систем автоматизованого проектування.

Основні програмні результати навчання

ПРН4. Знати та розуміти вплив технічних рішень в суспільному, економічному, соціальному і екологічному контексті.

ПРН5. Мати знання основ економіки та управління проектами

ПРН7. Вміти розв'язувати задачі аналізу та синтезу засобів, характерних для спеціальності.

ПРН8. Вміти системно мислити та застосовувати творчі здібності до формування нових ідей

ПРН12. Вміти ефективно працювати як індивідуально, так і у складі команди

ПРН16. Вміти оцінювати отримані результати та аргументовано захищати прийняті рішення

ПРН18. Використовувати інформаційні технології та для ефективного спілкування на професійному та соціальному рівнях

2. Пререквізити та постреквізити дисципліни (місце в структурно-логічній схемі навчання за відповідною освітньою програмою)

Матеріал курсу пов'язаній з матеріалами, які вивчаються у навчальних дисциплінах «Комп'ютерна логіка», «Алгоритми та структури даних», «Вища математика».

Матеріал даної навчальної дисципліни буде застосований при вивченні наступних навчальних дисциплін: «Дискретна математика», «Алгоритми та методи обчислень», «Комп'ютерне моделювання», «Програмна інженерія», «Системне програмування», «Технологія розподілених обчислень», «Мережні та інформаційні технології», «Комп'ютерне моделювання», «Основи програмної інженерії», «Системне програмування», «Технологія розподілених обчислень», «Мережеві та інформаційні технології» та інші.

3. Зміст навчальної дисципліни

Навчальна дисципліна «Програмування» включає вивчення наступних тем.

Тема 1. Вступ в курс лекцій «Програмування». Методи та задачі курсу, зв'язок з іншими дисциплінами. Принципи створення архітектури сучасних комп'ютерів. Загальне поняття про програмне забезпечення сучасних комп'ютерів: базовий рівень, системний рівень, службовий рівень, прикладний рівень. Поняття про програмне забезпечення системного рівня. Класифікація службових програмних засобів. Класифікація прикладного програмного забезпечення. Алгоритмічні мови й системи програмування. Рівні алгоритмічних мов програмування. Опис структури середовищ розробки програмного забезпечення.

Тема 2. Загальні відомості про скриптову мову програмування Python та інтегроване середовище розробки програм PyCharm Edu.

Основні характеристики мови Python: імперативне програмування, об'єктно-орієнтоване програмування, подійно-керовані програми (GUI), формати представлення даних. Технологія освоєння мови Python: семантика, синтаксис та прагматика. Історія створення мови Python. Дзен Пайтона. Короткі відомості про інсталяцію Python. Середовище програмування на Python IDLE. Основні правила роботи в середовищі IDLE, меню. Середовище розробки PyCharm, меню. Приклад програми «калькулятор».

Тема 3. Змінні в мові Python

Іменування змінних, список ключових слів та інструкція для їх виводу. Базові типи даних скриптової мови програмування Python. Способи присвоювання значення змінним. Визначення та перевірка типу даних. Способи перетворення типів даних. Видалення змінної.

Тема 4. Оператори в мові програмування Python

Різновидності математичних операторів та їх застосування. Двійкові оператори. Оператори для послідовностей. Оператори присвоювання. Пріоритет виконання операторів. Оператори умови. Оператори порівняння. Оператори розгалуження в мові Python: оператори розгалуження if…else, оператор циклу for та оператор циклу while.

Тема 5. Представлення чисел в мові Python

Базові типи для представлення чисел. Представлення чисел в десятковій, двійковій, вісімковій та шістнадцятковій системах числення. Операції над числами, які задані. Числа, що задані з фіксованою точністю за підтримки модуля decimal. Виконання операцій над дробами за підтримки методів модуля fractions. Формат представлення та операції над комплексними числами. Вбудовані функції для роботи з числами. Стандартні константи модуля math. Основні функції модуля math для роботи з числами. Застосування модуля random для генерації випадкових чисел.

Тема 6. Створення рядків та операції над рядками. Форматування рядків. Функції та методи для роботи з рядками.

Визначення рядка та основні властивості рядка як типу даних в системі програмування Python. Базові строкові типи в Python та сфери їх застосування. Способи створення рядка. Вплив модифікатора r на правила форматування рядка. Операції з рядками. Формат операції виконання зрізу. Базовий синтаксис форматування рядків, рядок спеціального формату та його параметри. Методи форматування рядків. Метод format(), його параметри та способи застосування. Основні функції для роботи з рядками. Основні методи для роботи з рядками. Налаштування локалі. Зміна регістру символів. Функції для роботи з символами. Пошук і заміна в рядку. Перевірка типу вмісту рядка.

Тема 7. Типи даних bytes та bytearray

Способи створення об'єктів типу bytes: за допомогою функції bytes з різними видами параметрів, за допомогою методу рядків encode(), за допомогою префікса b"xxx", за допомогою методу bytes.fromhex(). Способи створення об'єктів типу bytearray: за допомогою функції bytearray з різними видами параметрів, за допомогою методу bytearray.fromhex(). Методи зміни об'єкта типу bytearray. Шифрування рядків за допомогою функцій модуля hashlib.

Тема 8. Означення, властивості та застосування списків у мові Python.

Визначення списку як змінюваної послідовності даних. Способи створення списку. Способи створення копії списку. Операції над списками. Особливості формування та застосування багатовимірних списків.

Перебір елементів списку. Функція rang() та її параметри. Генератори списків та вирази-генератори. Методи додавання та видалення елементів списку.

Пошук елемента списку й одержання відомостей про значення, які входять в список. Властивості та параметри методу index(). Визначення кількості елементів. Визначення параметрів елементів списку. Перевертання та перемішування списку. Вибір елементів списку випадковим чином за допомогою функцій модуля random. Сортування списку. Заповнення списку числами. Перетворення списку на рядок.

Тема 9. Означення, властивості та застосування кортежів множин та діапазонів. Ітератори.

Незмінювані послідовності типу кортеж. Основна властивість типу «множина». Особливості змінюваних і незмінюваних множин. Основні властивості діапазонів.

Методи створення кортежів. Одержання елемента кортежу по індексу.

Створення множини за допомогою функції set(). Перебір елементів множини в циклі for. Означення та застосування операторів на множинах, які виконують логічні перетворення множин. Методи для роботи з множинами. Генератори множин та їх синтаксис. Незмінювані множини типу frozentset. Оператори та методи, що підтримують перетворення даних типу frozenset. Розгляд даних типу «діапазон» як послідовностей. Параметри та формат функції range для задавання діапазону. Функції, що підтримують перетворення даних типу «діапазон». Ітератори, що представлені функціями модуля itertools.

Тема 10. Означення, властивості та застосування словників.

Означення словників як відображень та порівняння їх властивостей з властивостями асоціативних масивів в інших мовах програмування. Способи створення словників. Операції над словниками: доступ до елементів словника, перевірка існування ключа. Визначення кількості ключів. Вилучення елементів словника за функцією del. Перебір елементів словника за допомогою циклу for. Сортування по ключах. Методи для роботи зі словниками. Генератори словників.

Тема 11. Визначення та властивості функцій користувача в Python

Визначення функції як фрагменту коду багаторазового використання. Схема створення функції за допомогою ключового слова def. Інструкція return. Функції зворотного виклику. Атрибути функції, вивід списку атрибутів за допомогою функції dir(). Строгий порядок слідування визначення та виклику функції. Необов'язкові параметри функції та їх зіставлення по ключах. Змінне число параметрів функції. Комбінування параметрів. Анонімні функції lambda. Способи задавання та сфери використання анонімних функцій. Функції-генератори. Застосування методу __next__ до функцій-генераторів. Виклик функції-генератора з функції-генератора за допомогою ключового слова yield from. Декоратори функцій. Рекурсія, обчислення факторіала. Видимість глобальних та локальних змінних. Одержання словників глобальних та локальних ідентифікаторів за допомогою функцій globals() та locals(). Вкладені функції. Анотації функцій. Атрибут об'єкта функції __anotations__

Тема 12. Модулі і пакети

Визначення модуля в Python. Головний модуль "__main__". Атрибут об'єкта модуля __main__: __name__. Інструкція import та варіанти її застосування для імпортування модулів. Імпорт кількох модулів однією інструкцією. Функція getattr() для динамічного формування назви атрибута в ході виконання програми. Перевірка існування атрибута за допомогою функції hasattr(). Способи одержання скомпільованого файла з розширенням *.рус. Шляхи пошуку модулів. Повторне завантаження модулів. Формат функції reload(). Означення пакета. Файл ініціалізації пакета __init__.py. Одержання доступу до ідентифікаторів імпортованого модуля. Імпорт з головного модуля та з файлів пакета.

Тема 13. Об'єктно-орієнтоване програмування

Основні означення, що пов'язані з об'єктно-орієнтованим програмуванням. Особливості інкапсуляції, спадкування та поліморфізму для ООП в Python. Визначення класу й створення екземпляра класу. Синтаксис створення атрибуту класу. Спосіб створення методу класу за допомогою інструкції def. Застосування змінної self для доступу до атрибутів та методів всередині класу. Методи __init__() для задавання початкових значень атрибутам екземпляра класу. Метод __del__(), як деструктор, що викликається при видаленні екземпляра класу. Ідея інкапсуляції та її реалізація в Python за допомогою методів класу. Перезавантаження операторів на прикладі методу __str__. Спадкування. Поняття базового класу (суперкласу) та похідного класу (підкласу). Множинне спадкування. Особливості реалізації поліморфізму в Python. Перезавантаження методів суперкласів їх підкласами. Домішки та їх використання. Застосування функції getattr() до значення атрибута по його назві. Функція setattr () для задавання значення атрибуту за його назвою. Видалення атрибута функцією delattr() та перевірка його значення функцією hasattr(). Принцип відкритості атрибутів в мові Python (public). Спеціальні методи екземпляра класу: виклик екземпляра класу як функції (__call__()), метод __getattr__ викликається при доступі до неіснуючого атрибута класу. Метод __getattribute__(), який викликається при доступі до будь-якого атрибута класу, метод __setattr__(), який викликається при спробі присвоювання значення атрибуту екземпляра класу, метод __delattr__(), який викликається при видаленні атрибута за допомогою інструкції del та ряд інших методів, що викликаються при виконанні арифметичних та логічних операцій.

Тема 14. Налагодження програм в PyCharm

Спектр можливостей з налагодження програм в PyCharm. Основні етапи налагодження. Налаштування параметрів програми Debugger. Створення точок останову (breakpoints) у вхідному коді. Опції точок останову. Видалення точок останову. Запуск сеансу налагодження. Налаштування додатка. Припинення й поновлення сеансу налагоджувача. Дії під час сеансу налаштування: моніторинг інформації про налагодження, вивчення припиненої програми, вивчення Frames, додавання, редагування й видалення watches, обчислення виразів та обмеження при обчисленні виразів, обчислення виразів у редакторі, інспектування спостережуваних елементів, перехід з Debug Tool Window до вхідного коду, знаходження точки поточного виконання. Покрокове виконання

програм.

Тема 15. Помилки в програмі Python та методи обробки виключень.

Опис типів помилок в програмі Python. Інструкція try...except...else...finally. Формати інструкції try. Застосування функції exc_info() для одержання інформації про виключення. Поняття про протокол менеджерів контексту. Інструкція with...as. Формати інструкції with...as. Метод __exit__() та його формат. Конструкція with open() та її використання при обробці виключень. Виключення користувача. Застосування інструкцій raises assert для створення виключень користувача. Основне поняття про клас Exception та його атрибути.

Тема 16. Підходи до зберігання інформації в програмах на мові програмування Python. Робота з файлами.

Основні поняття про файл та типи файлів. Відкриття файлу та формат функції open(). Абсолютний та відносний шлях до файлу. Перетворення відносного шляху у абсолютний. Можливі задавання шляхів до файлу та їх модифікація. Встановлення шляху до поточного каталогу. Одержання шляху до файлу, що виконується, за допомогою атрибута __file__. Одержання повного шляху до файлу. Модифікатори відкриття файлу. Особливості роботи з механізмом буферизації файлів. Поняття про кодування текстових файлів та особливості роботи з різними кодуваннями. Методи для роботи з файлами. Права доступу до файлів і каталогів. Визначення прав доступу. Функції для маніпулювання файлами. Перевірка наявності файлу, розміру файлу, часу останнього доступу, часу створення, часу останньої зміни. Функції для перетворення шляху до файлу. Перенаправлення вводу/виводу. Функція print(). Стандартний ввід і вивід.

Збереження об'єктів у файлі. Функції модуля pickle для роботи з файлами. Використання модуля shelve для зберігання даних в файлі по ключу. Функції для роботи з каталогами. Виключення, які виникають при виконанні операцій з файлами.

Тема 17. Графічний інтерфейс користувача. Бібліотека tkinter (віджети).

Підключення бібліотеки tkinter. Варіанти застосування інструкції import при роботі з tkinter. Створення головного вікна. Створення віджета. Задавання властивостей віджетів. Визначення подій та їх обробників. Розміщення віджета за допомогою методу pack. Відображення головного вікна. Огляд віджетів, які реалізовані в бібліотеці tkinter. Кнопка (Button), лейбл (Label), однорядкове текстове поле (Entry), багаторядкове текстове поле (Text), список (Listbox), фрейм (Frame), галочка (Checkbutton), галочка з одним вибором (Radiobutton), шкала (Scale), скролбар (Scrollbar). Принципи взаємного розміщення віджетів у tkinter. Пакувальник pack(). Пакування кількох віджетів одного над одним. Розміщення кількох віджетів поряд. Опції пакувальника pack():anchor, expand, fill, ipadx та ipady, padx та pady, side. Пакувальник grid(). Аргументи пакувальника grid: row, rowspan, column, columnspan, padx / pady, ipadx / ipady, sticky, in_. Додаткові функції пакувальника grid(). Пакувальник place. Синтаксис пакувальника place та його опції: anchor, bordermode, height, width, relheight, relwidth, relx, rely, x, y. Віджети меню: меню верхнього рівня (toplevel), випливаюче вікно (pop-up), та випадаюче меню (pull-down). Просунуті віджети типу OptionMenu. Синтаксис та властивості меню. Методи доступу на об'єктах меню. Кнопка меню (Menubutton), синтаксис та властивості. Застосування меню типу Toplevel: синтаксис, властивості та методи. Віджет, що задає тло для малювання (Canvas): синтаксис, властивості та методи. Прийоми малювання на Canvas. Віджет-контейнер (PaneWindow): синтаксис, властивості та методи. Модуль messagebox та його застосування для відображення вікон повідомлень у прикладних програмах. Події та зв'язування. Розгляд об'єкта Event та його атрибутів: widget, x, y, x_root, y_root, char, keysym, keycode, num, width, height, type. Зв'язування віджета з подіями за допомогою bind(). Представлення подій та основні види подій. Формати подій. Екземпляри класів та зв'язування класів. Методи зв'язування класів.

Інші callback-подібні методи. Протоколи та їх застосування.

Тема 18. Поняття про регулярні вирази та їх застосування в Python

Синтаксис регулярних виразів. Створення скомпільованого шаблону за допомогою функції compile(). Формат функції compile(). Застосування прапорів компіляції. Прив'язка регулярного виразу. Метасимволи вибору між альтернативними значеннями. Класи та квантифікатори в регулярних виразах. Звернення до іменованих фрагментів всередині шаблону. Вирази в круглих дужках, які задають опції регулярних виразів. Пошук першого збігу з шаблоном. Формати методів обробки регулярних виразів.

Тема 19. Програми на Python для взаємодії через локальні обчислювальні мережі.

Початкові відомості про взаємодію комп'ютерів через локальні обчислювальні мережі. Поняття TCP, UDP, FTP та HTTP протоколи взаємодії. Створення TCP-клієнта. Створення структури даних типу pickle для передачі. Використання контекстного менеджера користувача SocketManager. Створення TCP-сервера на базі модуля високого рівня SocketServer. Розгляд основних принципів роботи TCP-сервера. Функції та методи для прийому, зберігання та контролю даних.

4. Навчальні матеріали та ресурси

Базова:

1. Новотарський, М. А. Основи програмування алгоритмічною мовою Python [Електронний ресурс] : навч. посіб. для студ. освітньої програми «Комп'ютерні системи та мережі» спеціальності 123 «Комп'ютерна інженерія» / М. А. Новотарський ; КПІ ім. Ігоря Сікорського. – Електронні текстові дані (1 файл: 17.93 Мбайт). – Київ : КПІ ім. Ігоря Сікорського, 2022. – 701с. .chttps://ela.kpi.ua/handle/123456789/49913
2. Новотарський М. А. Лекції відео лекції, презентації:
https://classroom.google.com/u/0/w/NDE3ODIzNzU2MDk1/tc/NDEyNzkxMjk0ODc4
3. Новотарський М. А. Лабораторні роботи з курсу «Програмування»// Методичні вказівки, https://classroom.google.com/u/0/w/NDE3ODIzNzU2MDk1/tc/NDEyNzkwNTg3NDI3
4. Новотарський М. А. Інструкції з установки програмного забезпечення// https://classroom.google.com/u/0/w/NDE3ODIzNzU2MDk1/tc/NDEyODAwNjI3NTIw.

Додаткова:

1. Phillips D. Python 3 Object-oriented Programming. – Packt Publishing, 2015. – 460 p.
2. Christopher A. Jones, Fred L. Drake Jr. Python & XML. – O'Reilly Media, 2001. – 450 p.
3. Chun W.J. Core Python Programming, Second Edition. – Prentice Hall, 2006. – 1120 p.
4. Rappin N., Dunn R. wxPython in Action. – Manning Publications Co., 2006. – 552 p.
5. Summerfield M. Rapid GUI Programming with Python and Qt. – Prentice Hall, 2007. – 628 p.
6. Alchin M. Pro Python. – Apress, 2010. – 341 p.
7. Foundations of Python Network Programming: The comprehensive guide to building network applications with Python (Books for Professionals by Professionals) 2nd ed. Edition by John Goerzen (Author), Tim Bower (Author), Brandon Rhodes (Author).

Навчальний контент

5. Методика опанування навчальної дисципліни (освітнього компонента)

Структура навчальної дисципліни «Програмування» представлена в таблиці 1 .

Таблиця 1

Структура навчальної дисципліни «Програмування»

Назви розділів і тем	Кількість годин			
	Всього	у тому числі		
		Лекції	Лабораторні (Комп'ютерний практикум)	СРС
Тема 1. Вступ в курс лекцій «Програмування».	7	2	2	3
Тема 2. Загальні відомості про скриптову мову програмування Python та інтегроване середовище розробки програм PyCharm Edu.	7	2	2	3
Тема 3. Змінні в мові Python	7	2	2	3
Тема 4. Оператори в мові програмування Python.	10	4	2	4
Тема 5. Представлення чисел в мові Python	7	2	2	3
Тема 6. Створення рядків та операції над рядками. Форматування рядків. Функції та методи для роботи з рядками.	12	6	2	4
Тема 7. Типи даних bytes та bytearray	7	2	2	3
Тема 8. Означення, властивості та застосування списків в мові Python	11	5	2	4
Тема 9. Означення, властивості та застосування кортежів, множин та діапазонів. Ітератори.	8	3	2	3
Тема 10. Означення, властивості та застосування словників.	10	2	4	4
Тема 11. Визначення та властивості функцій користувача в Python	9	4	2	3
Тема 12. Модулі і пакети	6	4		2
Тема 13. Об'єктно-орієнтоване програмування	14	4	6	4
Тема 14. Налагодження програм в PyCharm	5	2		3
Тема 15. Помилки в програмі Python та методи обробки виключень	5	2		3
Тема 16. Підходи до зберігання інформації в програмах на мові програмування Python. Робота з файлами.	18	8	6	4
Тема 17. Графічний інтерфейс користувача. Бібліотека tkinter (віджети).	14	10		4
Тема 18. Поняття про регулярні вирази та їх застосування в Python	7	4		3
Тема 19. Програми на Python для взаємодії через локальні обчислювальні мережі.	7	4		3
Тестування закритими тестами	3			3
Екзамен	**36**			**36**
Всього в семестрі:	**210**	**72**	**36**	**102**

Тематика лекційних занять, яка сформована відповідно до темам, що розглядаються у рамках навчальної дисципліни, наведена у таблиці 2.

Таблиця 2

Лекційні заняття

№ з/п	Назва теми лекції та перелік основних питань (перелік дидактичних засобів, посилання на літературу та завдання на СРС)
1	**Вступ в курс лекцій «Програмування».** Методи та задачі курсу, зв'язок з іншими дисциплінами. Основні розділи та питання, які мають бути досліджені. Зміст самостійних та лабораторних робіт. Особливості виконання лабораторних робіт та індивідуальних завдань, вимоги до них. Базові відомості про апаратне забезпечення персональних та професійних комп'ютерів. Принципи створення архітектури сучасних комп'ютерів. Загальне поняття про програмне забезпечення сучасних комп'ютерів: базовий рівень, системний рівень, службовий рівень, прикладний рівень. Поняття про програмне забезпечення системного рівня. Класифікація службових програмних засобів. Класифікація прикладного програмного забезпечення. Алгоритмічні мови й системи програмування. Рівні алгоритмічних мов програмування. Опис структури середовищ розробки програмного забезпечення. **Завдання на СРС.** Розглянути наявне програмне забезпечення на своїх комп'ютерах та розподілити його за рівнями.
2	**Загальні відомості про скриптову мову програмування Python та інтегроване середовище розробки програм PyCharm Edu.** Основні характеристики мови Python: імперативне програмування, об'єктно-орієнтоване програмування, подійно-керовані програми (GUI), формати представлення даних. Технологія освоєння мови Python: семантика, синтаксис та прагматика. Історія створення мови Python. Дзен Пайтона. Короткі відомості про інсталяцію Python. Середовище програмування на Python IDLE. Основні правила роботи в середовищі IDLE, меню, середовище розробки Pycharm, меню. Приклад програми «калькулятор». **Завдання на СРС.** Підготовка середовища для опрацювання фрагментів програм та виконання лабораторних робіт: інсталяція Python. Інсталяція середовища розробки PyCharm.
3	**Змінні в мові Python** Іменування змінних, список ключових слів та інструкція для їх виводу. Базові типи даних скриптової мови програмування Python. Способи присвоювання значення змінним. Визначення та перевірка типу даних. Способи перетворення типів даних. Видалення змінної. **Завдання на СРС.** Підготовка та налаштування фрагментів програм для інтерпретатора та компілятора (Приклади 1–37 за матеріалами лекції 3).
4	**Оператори в мові програмування Python** Різновидності математичних операторів та їх застосування. Двійкові оператори. Оператори для послідовностей. Оператори присвоювання. Пріоритет виконання операторів. Оператори умови. Оператори порівняння. **Завдання на СРС.** Підготовка та налаштування фрагментів програм для інтерпретатора та компілятора (Приклади 1–19 за матеріалами лекції 4).
5	**Оператори в мові програмування Python** (продовження) Оператори розгалуження в мові Python: оператор розгалуження if…else, оператор циклу for, функції range() і enumerate(), оператор циклу while, оператор continue,

№ з/п	Назва теми лекції та перелік основних питань (перелік дидактичних засобів, посилання на літературу та завдання на СРС)
	оператор break. **Завдання на СРС.** Підготовка та налаштування фрагментів програм для інтерпретатора та компілятора (Приклади 1–24 за матеріалами лекції 5).
6	**Представлення чисел в мові Python** Базові типи для представлення чисел: int, float, complex. Представлення чисел в десятковій, двійковій, вісімковій та шістнадцятковій системах числення. Операції над числами, які задані. Числа, що задані з фіксованою точністю за підтримки модуля decimal. Виконання операцій над дробами за підтримки методів модуля fractions. Формат представлення та операції над комплексними числами. Вбудовані функції для роботи з числами: int, bin, oct, hex, float, round, abs, pow, max, min, sum, divmod. Стандартні константи модуля math. Основні функції модуля math для роботи з числами. Застосування модуля random для генерації випадкових чисел. **Завдання на СРС.** Підготовка та налаштування фрагментів програм для інтерпретатора та компілятора (Приклади 1–14 за матеріалами лекції 6).
7	**Створення рядків та операції над рядками** Визначення рядка та основні властивості рядка як типу даних в системі програмування Python. Базові строкові типи в Python та сфери їх застосування. Способи створення рядка: за допомогою функцій str, bytes та bytearray; шляхом задавання послідовності символів між подвійними лапками, апострофами або потроєними лапками. Вплив модифікатора r на правила форматування рядка. Застосування зворотного слеша та спеціальних символів для форматування рядка. Операції з рядками: доступ до елемента по індексу, одержання зрізу, конкатенація, повторення та перевірка на входження. **Завдання на СРС.** Підготовка та налаштування фрагментів програм для інтерпретатора та компілятора (Приклади 1–42 за матеріалами лекції 7).
8	**Форматування рядків** Формат операції виконання зрізу. Базовий синтаксис форматування рядків, рядок спеціального формату та його параметри. Методи форматування рядків: expandtabs, center, ljust, rjust, zfill. Метод format(), його параметри та способи застосування. **Завдання на СРС.** Підготовка та налаштування фрагментів програм для інтерпретатора та компілятора (Приклади 1–57 за матеріалами лекції 8).
9	**Функції та методи для роботи з рядками** Основні функції для роботи з рядками: str(), repr(), asci(), len(). Основні методи для роботи з рядками: strip(), lstrip(), rstrip(), split(), rsplit(), splitlines(), partition(), rpartition(), join(). Налаштування локалі. Зміна регістру символів. Функції для роботи з символами. Пошук і заміна в рядку за допомогою методів find(), index(), rfind(), rindex(), count(). Перевірка типу вмісту рядка. **Завдання на СРС.** Підготовка та налаштування фрагментів програм для інтерпретатора та компілятора (Приклади 1–42 за матеріалами лекції 9).
10	**Типи даних bytes та bytearray** Способи створення об’єктів типу bytes: за допомогою функції bytes з різними видами параметрів, за допомогою методу рядків encode(), за допомогою префікса b”xxx“, за допомогою методу bytes.fromhex(). Способи створення об’єктів типу bytearray: за допомогою функції bytearray з різними видами параметрів, за допомогою методу

№ з/п	Назва теми лекції та перелік основних питань (перелік дидактичних засобів, посилання на літературу та завдання на СРС)
	bytearray.fromhex(). Методи зміни об'єкта типу bytearray: append(), extend(), +, +=, присвоювання зрізу, insert(), pop(), remove(), reverse(), decode(). Шифрування рядків за допомогою функцій модуля hashlib. **Завдання на СРС.** Підготовка та налаштування фрагментів програм для інтерпретатора та компілятора (Приклади 1–37 за матеріалами лекції 10).
11	**Означення, властивості та застосування списків у мові Python** **Визначення списку** як змінюваної послідовності даних. Способи створення списку: за допомогою функції list, явно (перелічуванням елементів), за допомогою методу append(). Способи створення копії списку: за допомогою операції добування зрізу, з використанням функції list(), за допомогою методу copy(). **Операції над списками**: доступ до елементів списку, позиційне присвоювання значень, одержання кількості елементів списку за допомогою функції len(), операція добування зрізу для одержання фрагменту списку. Особливості формування та застосування багатовимірних списків. **Завдання на СРС.** Підготовка та налаштування фрагментів програм для інтерпретатора та компілятора (Приклади 1–32 за матеріалами лекції 11).
12	**Означення, властивості та застосування списків в мові Python** (продовження) **Перебір елементів списку**. Функція rang() та її параметри. Генератори списків та вирази-генератори. Функції map(), zip(), filter() і reduce(). Методи додавання та видалення елементів списку: append(), extend(), insert(), pop(), remove(), clear(). Оператор +=. **Завдання на СРС.** Підготовка та налаштування фрагментів програм для інтерпретатора та компілятора (Приклади 1–33 за матеріалами лекції 12).
13	**Означення, властивості та застосування списків в мові Python** (продовження). **Пошук** елемента списку й одержання відомостей про значення, які входять в список. Властивості та параметри методу index(). Визначення кількості елементів за методом count(). Визначення параметрів елементів списку за функціями max(), min(), any(), all(). Перевертання та перемішування списку за методом reverse() та функціями reversed() і shuffle(). Вибір елементів списку випадковим чином за допомогою функцій модуля random. Сортування списку за методом sort() і функцією sorted(). Заповнення списку числами. Перетворення списку в рядок. **Означення, властивості та застосування кортежів, множин та діапазонів.** Незмінювані послідовності типу кортеж. Основна властивість типу «множина». Особливості змінюваних і незмінюваних множин. Основні властивості діапазонів. Методи створення кортежів: за допомогою функціїї tuple(), явно (шляхом задавання елементів через кому). Одержання елементу кортежу по індексу. Застосування функцій len(), min(), max() та методів index() і count() до кортежів. Створення множини за допомогою функції set(). Перебір елементів множини в циклі for. Застосування функції len() до множини. Означення та застосування операторів на множинах, які виконують логічні перетворення множин. **Завдання на СРС.** Підготовка та налаштування фрагментів програм для інтерпретатора та компілятора (Приклади 1–39 за матеріалами лекції 13).
14	**Означення, властивості та застосування кортежів, множин та діапазонів. Ітератори** (продовження) Методи для роботи з множинами: copy(), add(), remove(), discard(), pop(), clear(). Генератори множин та їх синтаксис. Незмінювані множини типу frozenset. Оператори та

№ з/п	Назва теми лекції та перелік основних питань (перелік дидактичних засобів, посилання на літературу та завдання на СРС)
	методи, що підтримують перетворення даних типу frozenset. Розгляд даних типу «діапазон», як послідовностей. Параметри та формат функції range для задавання діапазону. Функції, що підтримують перетворення даних типу «діапазон»: доступ до елементів по індексу, одержання зрізу, перевірка на входження, перевірка на невходження, функції len(), min(), max(), методи index() і count(). Ітератори, що представлені функціями модуля itertools. Функції count(), cycle(), repeat(). **Завдання на СРС.** Підготовка та налаштування фрагментів програм для інтерпретатора та компілятора (Приклади 1–28 за матеріалами лекції 14).
15	**Означення, властивості та застосування словників** Означення словників як відображень та порівняння їх властивостей з властивостями асоціативних масивів в інших мовах програмування. Способи створення словників: за допомогою функції dict(), явний спосіб (шляхом перерахування у спеціальному форматі), шляхом задавання словника поелементно, за допомогою методу dict.fromkeys(). Операції над словниками: доступ до елементів словника, перевірка існування ключа за допомогою оператора not in, методів get() або setdefault(). Визначення кількості ключів за функцією len(). Вилучення елементів словника за функцією del. Перебір елементів словника за допомогою циклу for. Сортування по ключах методом sort() або функцією sorted(). Методи для роботи зі словниками: keys(), values(), items(), in, not in, get(), set default(), pop (), popitem(), clear(), update(), copy(), deepcopy(). Генератори словників. **Завдання на СРС.** Підготовка та налаштування фрагментів програм для інтерпретатора та компілятора (Приклади 1–35 за матеріалами лекції 15).
16	**Визначення та властивості функцій користувача в Python** Визначення функції як фрагменту коду багаторазового використання. Схема створення функції за допомогою ключового слова def. Інструкція return. Функції зворотного виклику. Атрибути функції, вивід списку атрибутів за допомогою функції dir(). Строгий порядок слідування визначення та виклику функції. Необов'язкові параметри функції та їх зіставлення по ключах. Змінне число параметрів функції. Комбінування параметрів. **Завдання на СРС.** Підготовка та налаштування фрагментів програм для інтерпретатора та компілятора (Приклади 1–28 за матеріалами лекції 16).
17	**Визначення та властивості функцій користувача в Python** (продовження). Анонімні функції lambda. Способи задавання та сфери використання анонімних функцій. Функції-генератори. Застосування методу __next__ до функцій-генераторів. Виклик функції-генератора з функції-генератора за допомогою ключового слова yield from. Декоратори функцій. Рекурсія, обчислення факторіала. Видимість глобальних та локальних змінних. Одержання словників глобальних та локальних ідентифікаторів за допомогою функцій globals() та locals(). Вкладені функції. Анотації функцій. Атрибут об'єкта функції __anotations__. **Завдання на СРС.** Підготовка та налаштування фрагментів програм для інтерпретатора та компілятора (Приклади 1–25 за матеріалами лекції 17).
18	**Модулі і пакети** Визначення модуля в Python. Головний модуль "__main__". Атрибут об'єкта модуля "__main__": __name__. Інструкція import та варіанти її застосування для імпортування

№ з/п	Назва теми лекції та перелік основних питань (перелік дидактичних засобів, посилання на літературу та завдання на СРС)
	модулів. Імпорт кількох модулів однією інструкцією. Функція getattr() для динамічного формування назви атрибута в ході виконання програми. Перевірка існування атрибута за допомогою функції hasattr(). Способи одержання скомпільованого файла з розширенням *.pyc. Шляхи пошуку модулів. Повторне завантаження модулів. Формат функції reload(). **Завдання на СРС.** Підготовка та налаштування фрагментів програм для інтерпретатора та компілятора (Приклади 1–15 за матеріалами лекції 18).
19	**Модулі і пакети** (продовження) Означення пакета. Файл ініціалізації пакета __init__.py. Одержання доступу до ідентифікаторів імпортованого модуля. Імпорт з головного модуля та з файлів пакета. **Завдання на СРС.** Підготовка та налаштування програм для компілятора (Приклади 1–6 за матеріалами лекції 19).
20	**Об'єктно-орієнтоване програмування (ООП)** Основні означення, що пов’язані з об'єктно-орієнтованим програмуванням. Особливості інкапсуляції, спадкування та поліморфізму для ООП в Python. Визначення класу й створення екземпляра класу. Синтаксис створення атрибуту класу. Спосіб створення методу класу за допомогою інструкції def. Застосування змінної self для доступу до атрибутів та методів всередині класу. Методи __init__() для задавання початкових значень атрибутам екземпляра класу. Метод __del__(), як деструктор, що викликається при видаленні екземпляра класу. Ідея інкапсуляції та її реалізація в Python за допомогою методів класу. Перезавантаження операторів на прикладі методу __str__. Спадкування. Поняття базового класу (суперкласу) та похідного класу (підкласу). Множинне спадкування. **Завдання на СРС.** Підготовка та налаштування програм для компілятора (Приклади 1–14 за матеріалами лекції 20).
21	**Об'єктно-орієнтоване програмування (ООП)** (продовження) Особливості реалізації поліморфізму в Python. Перезавантаження методів суперкласів їх підкласами. Домішки та їх використання. Застосування функції getattr() до значення атрибута по його назві. Функція setattr () для задавання значення атрибуту за його назвою. Видалення атрибуту функцією delattr() та перевірка його значення функцією hasattr(). Принцип відкритості атрибутів в мові Python (public). Спеціальні методи екземпляра класу: виклик екземпляра класу як функції (__call__()), метод __getattr__ викликається при доступі до неіснуючого атрибута класу. Метод __getattribute__(), який викликається при доступі до будь-якого атрибута класу, метод __setattr__(), який викликається при спробі присвоювання значення атрибуту екземпляра класу, метод __delattr__(), який викликається при видаленні атрибута за допомогою інструкції del та ряд інших методів, що викликаються при виконанні арифметичних та логічних операцій. **Завдання на СРС.** Підготовка та налаштування програм для компілятора (Приклади 1–16 за матеріалами лекції 21).
22	**Налагодження програм в PyCharm** Спектр можливостей з налагодження програм в PyCharm. Основні етапи налагодження. Налаштування параметрів програми Debugger. Створення точок останову (breakpoints) у вхідному коді. Опції точок останову. Видалення точок останову. Запуск сеансу налагодження. Налаштування додатка. Припинення й поновлення сеансу

№ з/п	Назва теми лекції та перелік основних питань (перелік дидактичних засобів, посилання на літературу та завдання на СРС)
	налагоджувача. Дії під час сеансу налаштування: моніторинг інформації про налагодження, вивчення припиненої програми, вивчення Frames, додавання, редагування й видалення watches, обчислення виразів та обмеження при обчисленні виразів, обчислення виразів у редакторі, інспектування спостережуваних елементів, перехід з Debug Tool Window до вхідного коду, знаходження точки поточного виконання. Покрокове виконання програм. **Завдання на СРС.** Виконати налаштування 2–4 Лабораторної роботи з застосуванням інструментів налаштування Debugger PyCharm.
23	**Помилки в програмі Python та методи обробки виключень** Опис типів помилок в програмі Python. Інструкція try...except...else...finally. Формати інструкції try. Застосування функції exc_info() для одержання інформації про виключення. Поняття про протокол менеджерів контексту. Інструкція with...as. Формати інструкції with...as. Метод __exit__() та його формат. Конструкція with open() та її використання при обробці виключень. Виключення користувача. Застосування інструкцій raises assert для створення виключень користувача. Основне поняття про клас Exception та його атрибути. **Завдання на СРС.** Підготовка та налаштування програм для компілятора (Приклади 1–21 за матеріалами лекції 23).
24	**Підходи до зберігання інформації в програмах на мові програмування Python. Робота з файлами.** Основні поняття про файл та типи файлів. Відкриття файлу та формат функції open(). Абсолютний та відносний шлях до файлу Перетворення відносного шляху у абсолютний за допомогою функції abspath() з модуля os.path. Можливі задавання шляхів до файлу та їх модифікація за допомогою функції sep з модуля os.path. Встановлення шляху до поточного каталогу з допомогою функції chdir() з модуля os. Одержання шляху до файлу, що виконується, за допомогою атрибута __file__. Одержання повного шляху до файлу з використанням атрибута __file__ та функції abspath() з модуля os.path. Модифікатори відкриття файлу. Особливості роботи з механізмом буферизації файлів. Поняття про кодування текстових файлів та особливості роботи з різними кодуваннями. Методи для роботи з файлами: close(), write(), writelines(), read(). **Завдання на СРС.** Підготовка та налаштування програм для компілятора (Приклади 1–19 за матеріалами лекції 24).
25	**Підходи до зберігання інформації в програмах на мові програмування Python. Робота з файлами** (продовження) Методи для роботи з файлами: readline(), readlines(), __next__(), flush(), fileno(), truncate(), tell(), seek(), seekable(). Права доступу до файлів і каталогів. Визначення прав доступу з допомогою функцій access () та chmod() з модуля os. Функції для маніпулювання файлами з модуля shutil: copyfile(), move(), rename(), remove(). **Завдання на СРС.** Підготовка та налаштування фрагментів програм для інтерпретатора та компілятора (Приклади 1–22 за матеріалами лекції 25).
26	**Підходи до зберігання інформації в програмах на мові програмування Python. Робота з файлами** (продовження) Перевірка наявності файлу функцією exists(), розміру файлу функцією getsize(), часу

№ з/п	Назва теми лекції та перелік основних питань (перелік дидактичних засобів, посилання на літературу та завдання на СРС)
	останнього доступу функцією getatime(), часу створення функцією getctime(), часу останньої зміни функцією getmtime (). Функції для перетворення шляху до файлу: abspath(), isabs(), basename(), dirname(), split(), splitdrive(), splitext(), join(). Перенаправлення вводу/виводу. Функція print(). Стандартний ввід і вивід. **Завдання на СРС.** Підготовка та налаштування фрагментів програм для інтерпретатора та компілятора (Приклади 1–26 за матеріалами лекції 26).
27	**Підходи до зберігання інформації в програмах на мові програмування Python. Робота з файлами** (продовження) Збереження об'єктів у файлі. Функції модуля pickle для роботи з файлами: dump(), load(), dumps(), loads(). Використання модуля shelve для зберігання даних в файлі по ключу. Функції для роботи з каталогами: getcwd(), mkdir(), listdir(), walk(), rmdir(), rmtree(), isdir(), islink (). Виключення, які виникають при виконанні операцій з файлами. **Завдання на СРС.** Підготовка та налаштування фрагментів програм для інтерпретатора та компілятора (Приклади 1–19 за матеріалами лекції 27).
28	**Графічний інтерфейс користувача. Бібліотека tkinter (віджети).** Підключення бібліотеки tkinter. Варіанти застосування інструкції import при роботі з tkinter. Створення головного вікна. Створення віджета. Задавання властивостей віджетів. Визначення подій та їх обробників. Розміщення віджета за допомогою методу pack. Відображення головного вікна. Огляд віджетів, які реалізовані в бібліотеці tkinter. Кнопка (Button), лейбл (Label), однорядкове текстове поле (Entry), багаторядкове текстове поле (Text), список (Listbox), фрейм (Frame), галочка (Checkbutton), галочка з одним вибором (Radiobutton), шкала (Scale), скролбар (Scrollbar). Принципи взаємного розміщення віджетів у tkinter. **Завдання на СРС.** Підготовка та налаштування фрагментів програм для компілятора (Приклади 1–28 за матеріалами лекції 28).
29	**Графічний інтерфейс користувача. Бібліотека tkinter (віджети)** (продовження). Пакувальник pack(). Пакування кількох віджетів одного над одним. Розміщення кількох віджетів поряд. Опції пакувальника pack(): anchor, expand, fill, ipadx та ipady, padx та pady, side. Пакувальник grid(). Аргументи пакувальника grid: row, rowspan, column, columnspan, padx / pady, ipadx / ipady, sticky, in_. Додаткові функції пакувальника grid(). Пакувальник place. Синтаксис пакувальника place та його опції: anchor, bordermode, height, width, relheight, relwidth, relx, rely, x, y. **Завдання на СРС.** Підготовка та налаштування фрагментів програм для компілятора (Приклади 1–13 за матеріалами лекції 29).
30	**Графічний інтерфейс користувача. Бібліотека tkinter (віджети)** (продовження). Віджети меню: меню верхнього рівня (toplevel), випливаюче вікно (pop-up), та випадаюче меню (pull-down). Просунуті віджети типу OptionMenu. Синтаксис та властивості меню. Методи доступу на об'єктах меню. Кнопка меню (Menubutton), синтаксис та властивості. Застосування меню типу Toplevel: синтаксис, властивості та методи. **Завдання на СРС.** Підготовка та налаштування фрагментів програм для компілятора (Приклади 1–6 за матеріалами лекції 30).

№ з/п	Назва теми лекції та перелік основних питань (перелік дидактичних засобів, посилання на літературу та завдання на СРС)
31	**Графічний інтерфейс користувача. Бібліотека tkinter (віджети)** (продовження). Віджет, що задає тло для малювання (Canvas): синтаксис властивості та методи. Прийоми малювання на Canvas. Віджет-контейнер (PaneWindow): синтаксис, властивості та методи. Модуль messagebox та його застосування для відображення вікон повідомлень у прикладних програмах. **Завдання на СРС.** Підготовка та налаштування фрагментів програм для компілятора (Приклади 1–7 за матеріалами лекції 31).
32	**Графічний інтерфейс користувача. Бібліотека tkinter (віджети)** (продовження). Події та зв'язування. Розгляд об'єкта Event та його атрибутів: widget, x, y, x_root, y_root, char, keysym, keycode, num, width, height, type. Зв'язування віджета з подіями за допомогою bind(). Представлення подій та основні види подій. Формати подій. Екземпляри класів та зв'язування класів. Методи зв'язування класів. Інші callback-подібні методи. Протоколи та їх застосування. **Завдання на СРС.** Підготовка та налаштування фрагментів програм для компілятора (Приклади 1–7 за матеріалами лекції 32).
33	**Поняття про регулярні вирази та їх застосування в Python** Синтаксис регулярних виразів. Створення скомпільованого шаблону за допомогою функції compile(). Формат функції compile(). Застосування прапорів компіляції. Прив'язка регулярного виразу. Метасимволи вибору між альтернативними значеннями. Класи та квантифікатори в регулярних виразах. Звернення до іменованих фрагментів всередині шаблону. Вирази в круглих дужках, які задають опції регулярних виразів. **Завдання на СРС.** Підготовка та налаштування фрагментів програм для інтерпретатора та компілятора (Приклади 1–26 за матеріалами лекції 33).
34	**Поняття про регулярні вирази та їх застосування в Python** (продовження). Пошук першого збігу з шаблоном. Формат методу mach(). Перевірка збігу з довільною частиною рядка. Формат методу search. Метод fullmach() та його формат. Властивості та методи об'єкта, що повертається методами match() і search(). Пошук всіх збігів з шаблоном. Метод findall() та його формат. Особливості застосування методу finditer() та його формат. Заміна в рядку методом sub(). Формат методу sub(). Метод subn(): формат та застосування. Інші методи обробки рядків: split(). **Завдання на СРС.** Підготовка та налаштування фрагментів програм для інтерпретатора та компілятора (Приклади 1–29 за матеріалами лекції 34).
35	**Програми на Python для взаємодії через локальні обчислювальні мережі.** Початкові відомості про взаємодію комп'ютерів через локальні обчислювальні мережі. Поняття TCP, UDP, FTP та HTTP протоколи взаємодії. Створення TCP-клієнта. Створення структури даних типу pickle для передачі. Використання контекстного менеджера користувача SocketManager. **Завдання на СРС.** Підготовка та налаштування фрагментів програм для компілятора (Приклади 1–8 за матеріалами лекції 35).
36	**Програми на Python для взаємодії через локальні обчислювальні мережі.** Створення TCP-сервера на базі модуля високого рівня SocketServer. Розгляд основних принципів роботи TCP-сервера. Функції та методи для прийому, зберігання та контролю даних. **Завдання на СРС.** Підготовка та налаштування фрагментів програм для компілятора

№ з/п	Назва теми лекції та перелік основних питань (перелік дидактичних засобів, посилання на літературу та завдання на СРС)
	(Приклади 1–11 за матеріалами лекції 36).

Самостійна робота студента

Самостійна робота студента складається з теоретичної та практичної складової. Теоретична складова передбачає вивчення додаткового матеріалу, що поглиблює знання, які отримані на лекції. Матеріал, який необхідно додатково вивчити, та літературні джерела для вивчення даного матеріалу вказані у таблиці 2. Практична складова самостійної роботи студента полягає у виконанні лабораторних робіт. Завдання циклу лабораторних занять полягає у набутті студентами необхідних практичних навичок з розробки прикладного програмного забезпечення скриптовою мовою програмування Python для розв'язування обчислювальних задач на комп'ютерах. Теми лабораторних робіт наведено в таблиці 3.

Таблиця 3.

Лабораторні роботи

№ з/п	Назва лабораторної роботи	Кількість ауд.годин
1	Лабораторна робота №1. Загальні поняття про технічні характеристики та програмне забезпечення сучасних комп'ютерів (Тема 1, Тема 2).	4
2	Лабораторна робота №2. Базові типи даних, змінні та оператори мови програмування Python (Тема 3, Тема 4, Тема 5).	6
3	Лабораторна робота №3. Робота з даними типу str, bytes та bytearray (Тема 6, Тема 7)	4
4	Лабораторна робота №4. Списки, кортежі, множини і діапазони в Python (Тема 8, Тема 9).	4
5	Лабораторна робота №5. Словники та функції користувача (Тема 10, Тема 11).	6
6	Лабораторна робота №6. Об'єктно-орієнтоване програмування на Python. Класи та екземпляри класів (Тема 13).	6
7	Лабораторна робота №7. Робота з файлами на Python. Запис у файл, зчитування з файлу, модифікація (Тема 16)	6

Політика та контроль

6. Політика навчальної дисципліни (освітнього компонента)

Під час занять з навчальної дисципліни «Програмування» студенти повинні дотримуватись певних дисциплінарних правил:

- забороняється запізнюватись на заняття;
- не допускається користування мобільними телефонами та іншими технічними засобами без дозволу викладача.

Лабораторні роботи здаються особисто з попередньою перевіркою теоретичних знань, які необхідні для виконання лабораторної роботи. Перевірка практичних результатів включає перевірку коду та виконання тестових завдань.

В процесі навчання викладач має право нарахувати до 5 заохочувальних балів за дострокове виконання лабораторної роботи, за проявлений творчий підхід при виконанні індивідуального

завдання або за активну участь у обговоренні питань, що пов'язані з тематикою лекції або практичного заняття.

За виконання та здачу лабораторної роботи після зазначеного дедлайну, за значну кількість пропущених занять, або за порушення правил поведінки на заняттях викладач може призначити до 5 штрафних балів. (У період військового стану штрафні бали не застосовуються)

При проведені контрольних заходів та при виконанні лабораторних робіт студенти повинні дотримуватися правил академічної доброчесності. При виявленні значного відсотку списування або плагіату викладач може відмовити у прийнятті даної роботи та вимагати доброчесного виконання навчального плану.

7. Види контролю та рейтингова система оцінювання результатів навчання (РСО)

Види контролю з навчальної дисципліни «Програмування» включають:

Лабораторні роботи:

Заплановано самостійне виконання семи лабораторних робіт.

Теми лабораторних робіт узгоджені у часі та за змістом з темами лекцій. Виконання лабораторних робіт у повному обсязі дозволяє набути практичних навичок програмування алгоритмічною мовою Python та оволодіти сучасними технологіями програмування.

Поточний контроль:

Передбачено 2 поточних тестування закритими тестами у системі TCEXAM, які повністю охоплюють тематику даної навчальної дисципліни. Кожний поточний закритий тест містить 30 питань та триває 30 хв. У випадку дистанційного навчання закритий поточний тест проводиться у терміни, які визначаються викладачем.. При очному навчанні час чергового поточного тестування призначається викладачем за узгодженням зі студентами.

Екзамен проводиться у вигляді екзаменаційного тесту у системі TCEXAM.

Тест проводиться в екзаменаційну сесію, триває 45 хвилин та складається з 40 питань.

Таблиця відповідності рейтингових балів оцінкам за університетською шкалою:

Кількість балів	*Оцінка*
100-95	Відмінно
94-85	Дуже добре
84-75	Добре
74-65	Задовільно
64-60	Достатньо
Менше 60	Незадовільно
Не виконані умови допуску	Не допущено

Оскільки навчальна дисципліна «Програмування» має семестрову атестацію у вигляді екзамену, рейтингова система оцінювання побудована за типом РСО – 2. Семестровий рейтинг студента складається з балів, які він отримує за види робіт відповідно до таблиці 1.

Таблиця 1

Оцінювання окремих видів навчальної роботи студента (у балах)

Вид навчальної роботи	**Всього за видом роботи**
Виконання та захист лабораторної роботи № 1	10
Виконання та захист лабораторної роботи № 2	10
Виконання та захист лабораторної роботи № 3	10
Виконання та захист лабораторної роботи № 4	10

Вид навчальної роботи	Всього за видом роботи
Виконання та захист лабораторної роботи № 5	10
Виконання та захист лабораторної роботи № 6	10
Виконання та захист лабораторної роботи № 7	10
Результати поточного тестування (5 тестувань х 2 бали)	10
Результати календарного тестування у ТСЕХАМ (2 тестування х 10 балів)	20
Rп	**100**
Визначення семестрового рейтингу за формулою: **Rc=Rп** х 0.6=100 х 0.6 **Rc**	**60**
Екзамен (Re)	**40**
Усього за семестр (**R=Rc+Re**)	**100**

Індивідуальний поточний рейтинг студента (**Rп**) складається з балів, які він отримує за:

1) Поточний контроль виконується у системі ТСЕХАМ.

2) Виконання лабораторних робіт. Протягом семестру студенти виконують 7 лабораторних робіт. Максимальна кількість балів за кожну лабораторну роботу – 10. Бали нараховуються за:

- Теоретична складова – 5 балів,
- Практична складова – 5 балів

Максимальний можливий бал за лабораторну роботу –10 балів.

Максимальна кількість балів за всі лабораторні роботи 7х10=70 балів.

Розрахунок розміру шкали (R) рейтингу.

Сума вагових балів контрольних заходів протягом семестру становить:

R = Rc + Re, де Rc – семестровий рейтинг студента (контрольні, лабораторні роботи).

Оцінювання поточних тестувань:

5х2=10 балів

Оцінювання календарних тестувань

2х10=20 балів

Обчислення семестрового рейтингу відбувається на основі поточного рейтингу Rп за формулою $R_c = \frac{R_п}{100} \cdot 60\%$. Максимальне значення семестрового рейтингу Rc=Rп х 0.6 = 100 х 0.6 = 60 балів;

Екзамен проводиться у вигляді тесту у системі ТСЕХАМ. Тест складається з 40 питань. Час проведення екзаменаційного тестування – 45 хвилин.

Re – ваговий бал екзамену, Re = 40 балів.

Розмір рейтингової шкали для навчальної дисципліни становить:

R = Rc + Re = 60 + 40 = 100 балів.

Необхідною умовою допуску студента до екзамену є його індивідуальний семестровий рейтинг (**R_c**), не менший, ніж 50% від Rc, тобто 30 балів, та відсутність заборгованості з лабораторних робіт. При невиконанні згаданих вимог студент до екзамену не допускається.

8. Додаткова інформація з дисципліни (освітнього компонента)

Викладання дисципліни «Програмування» для напряму «Комп'ютерна інженерія» має свою специфіку, яка пов'язана з тим, що сфера застосування методів машинного аналізу даних постійно

розширюється. Разом з тим, мова програмування Python, яка вивчається в рамках даної навчальної дисципліни, є основною алгоритмічною мовою, яку використовують в таких задачах.

Робочу програму навчальної дисципліни (силабус):

Складено: д.т.н. проф. Новотарський Михайло Анатолійович

Ухвалено кафедрою ОТ (протокол № 13 від 10.05.2023)

Погоджено: Методичною комісією ФІОТ (протокол № 11 від 29.06.2023)