Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»

Кафедра обчислювальної техніки

Технології програмування на мові Python

Робоча програма кредитного модуля (Силабус)

Реквізити навчальної дисципліни

Рівень вищої освіти	***Перший (бакалаврський)***
Галузь знань	*12 Інформаційні технології*
Спеціальність	*121 Інженерія програмного забезпечення* *123 Комп'ютерна інженерія* *126 Інформаційні системи та технології*
Освітня програма	*Інженерія програмного забезпечення інформаційних систем* *Інженерія програмного забезпечення комп'ютерних систем* *Комп'ютерні системи та мережі* *Інтегровані інформаційні системи* *Інформаційні управляючі системи та технології* *Інформаційне забезпечення робототехнічних систем*
Статус дисципліни	*Вибіркова*
Форма навчання	*Очна (денна)*
Рік підготовки, семестр	*3 курс, осінній семестр*
Обсяг дисципліни	*4 кредити, 120 годин* *54 години аудиторної роботи, 66 годин самостійної роботи*
Семестровий контроль, контрольні заходи	*Залік, лабораторні роботи, тести, модульна контрольна робота*
Розклад занять	*Згідно розкладу на осінній семестр поточного навчального року за посиланням* schedule.kpi.ua
Мова викладання	*Українська*
Інформація про керівника курсу, викладачів	*Асистент кафедри обчислювальної техніки, доктор філософії Шульга Максим Володимирович,* shulha.maksym@edu.kpi.ua
Розміщення курсу	*Платформа дистанційного навчання «Сікорський» в середовищі Google Workspace for Education*: https://classroom.google.com/c/Nzc0OTU1Mjg1ODk5?cjc=lxayc5yr

Програма навчальної дисципліни

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

Метою навчальної дисципліни є надання студентам всебічної підготовки з технологій програмування на Python і їх застосування у сучасній інженерії програмного забезпечення. Це включає розробку додатків з використанням фреймворків Python, розгортання рішень за допомогою контейнеризації та хмарних обчислень, проєктування мікросервісів і безсерверних архітектур, а також забезпечення безпеки та моніторингу додатків.

Предметом навчальної дисципліни є технології програмування на мові Python, що надають студентам розуміння Python, що включає фреймворки, контейнеризацію, хмарні обчислення, безпеку та моніторинг.

Завданням вивчення навчальної дисципліни є:

- ефективно використовувати Python для розробки додатків, включаючи веб-проєкти з Flask та Django;
- застосовувати технології контейнеризації, такі як Docker, Kubernetes та OpenShift, для оптимізації процесів розгортання;
- проєктувати та реалізовувати мікросервіси й безсерверні архітектури для створення масштабованих і гнучких додатків;
- розгортати додатки на Python у хмарних середовищах;
- інтегрувати можливості генеративного штучного інтелекту у Python-додатки;
- розуміти та впроваджувати заходи безпеки для захисту додатків протягом усього процесу розробки;
- застосовувати інструменти моніторингу, журналювання та спостереження для забезпечення надійності та продуктивності додатків.

Вивчення дисципліни підсилює наступні **загальні та фахові компетенції**:

- Для спеціальності *121 Інженерія програмного забезпечення*:
    - Здатність до абстрактного мислення, аналізу та синтезу.
    - Здатність застосовувати знання у практичних ситуаціях.
    - Здатність працювати в команді.
    - Здатність ідентифікувати, класифікувати та формулювати вимоги до програмного забезпечення.
    - Здатність брати участь у проектуванні програмного забезпечення, включаючи проведення моделювання (формальний опис) його структури, поведінки та процесів функціонування.
    - Здатність розробляти архітектури, модулі та компоненти програмних систем.
    - Здатність формулювати та забезпечувати вимоги щодо якості програмного забезпечення у відповідності з вимогами замовника, технічним завданням та стандартами.
    - Здатність дотримуватися специфікацій, стандартів, правил і рекомендацій в професійній галузі при реалізації процесів життєвого циклу.
    - Здатність аналізувати, вибирати і застосовувати методи і засоби для забезпечення інформаційної безпеки (в тому числі кібербезпеки).
- Для спеціальності *123 Комп'ютерна інженерія*:
    - Здатність до абстрактного мислення, аналізу та синтезу.
    - Здатність застосовувати знання у практичних ситуаціях.
    - Здатність вчитися і оволодівати сучасними знаннями.
    - Здатність використовувати сучасні методи і мови програмування для розроблення алгоритмічного та програмного забезпечення.
    - Здатність створювати системне та прикладне програмне забезпечення комп’ютерних систем та мереж.
    - Здатність забезпечувати захист інформації, що обробляється в комп’ютерних та кіберфізичних системах та мережах з метою реалізації встановленої політики інформаційної безпеки.

- Здатність проектувати, впроваджувати та обслуговувати комп'ютерні системи та мережі різного виду та призначення.
- Здатність ідентифікувати, класифікувати та описувати роботу програмно-технічних засобів, комп'ютерних та кіберфізичних систем, мереж та їхніх компонентів шляхом використання аналітичних методів і методів моделювання.
- Здатність розробляти, адаптувати, використовувати програмно-технічні засоби для збирання, оброблення та інтелектуального аналізу даних в комп'ютерних системах: паралельних, розподілених, хмарних, безпечних, інтелектуальних, розумних, Інтернет речей, зокрема, з використанням штучного інтелекту.

- Для спеціальності *126 Інформаційні системи та технології*:
  - Здатність до абстрактного мислення, аналізу та синтезу.
  - Здатність застосовувати знання у практичних ситуаціях.
  - Здатність вчитися і оволодівати сучасними знаннями.
  - Здатність до алгоритмічного мислення при розробці програмного забезпечення інформаційно-управляючих систем.
  - Здатність аналізувати, вибирати і застосовувати методи і засоби для забезпечення інформаційної безпеки.
  - Здатність до розробки і використання інтелектуальних інформаційних систем, технологій генерації та аналізу знань, алгоритмів штучного інтелекту для вирішення прикладних задач і підтримки прийняття рішень в різних прикладних областях життєдіяльності людини.
  - Здатність застосовувати стандарти в області інформаційних систем та технологій при розробці функціональних профілів, побудові та інтеграції систем, продуктів, сервісів і елементів інфраструктури організації.
  - Здатність розробляти, модифікувати та використовувати програмно-апаратні засоби для збирання, оброблення та інтелектуального аналізу даних.
  - Здатність використовувати технології розподілених обчислень, віртуалізації серверних систем, проєктувати корпоративні обчислювальні системи, застосовувати кластерні та гетерогенні розподілені обчислювальні системи для розв'язання прикладних задач і проведення наукових досліджень.

У відповідності до вищезазначеного, підсилені загальні та фахові компетенції дадуть наступні результати навчання:

- Для спеціальності *121 Інженерія програмного забезпечення*:
  - *Аналізувати, цілеспрямовано шукати і вибирати необхідні для вирішення професійних завдань інформаційно-довідникові ресурси і знання з урахуванням сучасних досягнень науки і техніки.* — Студенти отримують навички пошуку, оцінки та застосування актуальної інформації для розробки ПЗ і прийняття технічно обґрунтованих рішень.
  - *Знати та вміти використовувати методи та засоби збору, формулювання та аналізу вимог до програмного забезпечення.* — Студенти навчаються формувати технічні та функціональні вимоги, що забезпечують якісне проектування та реалізацію ПЗ.

- *Застосовувати на практиці ефективні підходи щодо проектування програмного забезпечення.* — Студенти набувають практичних умінь проектувати структуру і архітектуру програмних систем із застосуванням сучасних методологій.
- *Знати і застосовувати методи розробки алгоритмів, конструювання програмного забезпечення та структур даних і знань.* — Студенти отримують вміння створювати ефективні алгоритми, працювати зі структурами даних та будувати логіку програмних модулів.
- *Мати навички командної розробки, погодження, оформлення і випуску всіх видів програмної документації.* — Студенти отримують досвід колективної роботи над проєктом, ведення документації та організації робочого процесу.
- *Знати, аналізувати, вибирати, кваліфіковано застосовувати засоби забезпечення інформаційної безпеки (в тому числі кібербезпеки) і цілісності даних відповідно до розв'язуваних прикладних завдань та створюваних програмних систем.* — Студенти набувають умінь забезпечувати безпеку ПЗ та захист даних у розроблюваних додатках.

- Для спеціальності *123 Комп'ютерна інженерія*:
  - *Вміти застосовувати знання для ідентифікації, формулювання і розв'язування технічних задач спеціальності, використовуючи методи, що є найбільш придатними для досягнення поставлених цілей.* — Студенти отримують практичні навички визначати завдання, обирати ефективні методи та реалізовувати рішення для комп'ютерних систем і мереж.
  - *Вміти розробляти програмне забезпечення для вбудованих і розподілених застосувань, мобільних і гібридних систем, розраховувати, експлуатувати, типове для спеціальності обладнання.* — Студенти набувають умінь створювати ПЗ для сучасних комп'ютерних систем і використовувати обладнання за призначенням.
  - *Вміти ефективно працювати як індивідуально, так і у складі команди.* — Студенти здобувають досвід командної розробки проєктів, узгодження дій і спільного досягнення результатів.
  - *Вміти ідентифікувати, класифікувати та описувати роботу комп'ютерних систем та їх компонентів.* — Студенти набувають умінь аналізувати структуру та функціонування систем для ефективного проєктування та модернізації.
  - *Виконувати проектування та розрахунки параметрів складових компонентів комп'ютерів та комп'ютерних систем, універсального та спеціального призначення, локальних та глобальних комп'ютерних мереж, мережі Інтернет, Інтернету речей.* — Студенти отримують навички інженерного проєктування та оптимізації IT-систем та мереж.
  - *Вміти розроблювати та обслуговувати бази даних для IT-інфраструктур.* — Студенти набувають практичних навичок роботи з базами даних для забезпечення функціонування і надійності IT-систем.
- Для спеціальності *126 Інформаційні системи та технології*:
  - *Використовувати базові знання інформатики й сучасних інформаційних систем та технологій, навички програмування, технології безпечної роботи в комп'ютерних мережах, методи створення баз даних та*

інтернет-ресурсів, технології розроблення алгоритмів і комп'ютерних програм мовами високого рівня із застосуванням об'єктно-орієнтованого програмування для розв'язання задач проектування і використання інформаційних систем та технологій. — Студенти отримують практичні навички програмування та створення інформаційних систем, а також забезпечення їх безпеки та надійності.

- *Проводити системний аналіз об'єктів проектування та обґрунтовувати вибір структури, алгоритмів та способів передачі інформації в інформаційних системах та технологіях.* — Студенти навчаються аналізувати об'єкти і приймати обґрунтовані рішення щодо архітектури і функціональності систем.
- *Демонструвати знання сучасного рівня технологій інформаційних систем, практичні навички програмування та використання прикладних і спеціалізованих комп'ютерних систем та середовищ з метою їх запровадження у професійній діяльності.* — Студенти здобувають компетенції для ефективного застосування сучасних IT-рішень у професійних проєктах.
- *Вміти застосовувати методи та засоби аналізу даних, обирати та використовувати математичні моделі, будувати стратегії розв'язання практичних задач, в тому числі в галузі штучного інтелекту, обґрунтовувати вибір методу оптимізації при розв'язанні прикладних проблем у спеціалізованих сферах професійної діяльності.* — Студенти набувають умінь обробляти дані, моделювати системи та застосовувати алгоритми для оптимізації рішень.
- *Аргументувати вибір програмних та технічних засобів для створення інформаційних систем та технологій на основі аналізу їх властивостей, призначення і технічних характеристик з урахуванням вимог до системи і експлуатаційних умов, мати навички налагодження та тестування програмних і технічних засобів інформаційних систем та технологій.* — Студенти отримують навички професійного проєктування та тестування IT-систем і програмного забезпечення.
- *Знати методи захисту інформації, моделі безпеки інформаційних систем, методи проєктування, розробки та реалізації вимог до забезпечення якості інформаційних управляючих систем та використовувати ці знання при створенні захищених інформаційних систем.* — Студенти навчаються забезпечувати інформаційну безпеку та якість розроблюваних систем.

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

При вивченні цієї дисципліни використовуються знання студентів з дисциплін: "Програмування", "Вступ до операційної системи Linux", "Організація баз даних", "Інженерія програмного забезпечення".

Для успішного засвоєння дисципліни «Технології програмування на мові Python» студентам доцільно мати базові теоретичні та практичні знання, необхідні для розробки програмного забезпечення, роботи з сучасними інформаційними системами та застосуванням Python у різних IT-проєктах, які набуті під час вивчення базових та професійно орієнтованих дисциплін, і охоплюють такі напрями:

- Програмування та алгоритмічне мислення;
- Основи об'єктно-орієнтованого та функціонального програмування;

- Робота з масивами, структурами даних та базами даних;
- Основи веб-розробки та використання фреймворків Python.

Знання та навички, отримані під час вивчення дисципліни, можуть бути застосовані під час вирішення навчальних та практичних завдань, що виникають під час:

- Розробки та супроводу програмного забезпечення різного призначення;
- Проєктування масштабованих та гнучких IT-систем і додатків;
- Впровадження безпечних та надійних веб- і корпоративних рішень;
- Тестування, моніторингу та оптимізації програмних систем;
- Інтеграції створених рішень у корпоративні або хмарні середовища.

Здобуті навички можуть бути використані при виконанні курсових та бакалаврських проєктів, а також у практикоорієнтованих ініціативах, таких як стартапи, хакатони або дослідницькі лабораторії, пов'язані з розробкою програмного забезпечення, веб-додатків та IT-інфраструктури на основі Python.

3. Зміст навчальної дисципліни

Розділ 1. Технології програмування на Python
- Тема 1.1. Знайомство з Python
- Тема 1.2. Розробка додатків з Flask
- Тема 1.3. Розробка додатків з Django і базами даних
- Тема 1.4. Робота з контейнерами: Docker, Kubernetes і OpenShift
- Тема 1.5. Розробка додатків на основі мікросервісів і безсерверних архітектур
- Тема 1.6. Хмарні обчислення для додатків на Python
- Тема 1.7. Розробка додатків на основі генеративного штучного інтелекту з використанням Python

Розділ 2. Безпека та моніторинг додатків
- Тема 2.1. Основи безпеки додатків на Python
- Тема 2.2. Ризики безпеки додатків та найкращі практики
- Тема 2.3. Моніторинг додатків
- Тема 2.4. Журналювання та спостереження

4. Навчальні матеріали та ресурси

Базові ресурси:

1. E3: Python Programming: Web Development, Flask, Django, FastAPI. – E3, 2025. – 438 p.
2. David Ashley: Foundation Dynamic Web Pages with Python: Create Dynamic Web Pages with Django and Flask. – Apress, 2020. – 213 p.
3. Ben Shaw, Saurabh Badhwar, Andrew Bird, Bharath Chandra K.S., Chris Guest: Web Development with Django: Learn to Build Modern Web Applications with a Python-Based Framework. – Packt Publishing, 2021. – 826 p.

Додаткові ресурси:

1. Pierre Bourque and Richard Fairley: Guide to the Software Engineering Body of Knowledge, Version 3.0 SW. – IEEE Computer Society, 2014. – 346 p.
2. David Beazley, Brian Jones: Python Cookbook, 3rd ed. – O'Reilly Media, 2013. – 704 p.
3. Ian Sommerville: Software Engineering, 10th ed. – Addison-Wesley, 2016. – 790 p.
4. Karl Wiegers, Joy Beatty: Software Requirements, 3rd ed. – Microsoft Press, 2013. – 672 p.

5. Paul Clements, Felix Bachmann, Len Bass, David Garlan, James Ivers, Reed Little, Paulo Merson, Robert Nord, Judith Stafford: Documenting Software Architectures: Views and Beyond, 2nd ed. – Pearson Education, 2010. – 592 p.

Інформаційні ресурси:

1. Дистанційний курс на платформі дистанційного навчання «Сікорський» в середовищі Google Workspace for Education: Технології програмування на мові Python — https://classroom.google.com/c/Nzc0OTU1Mjg1ODk5?cjc=lxayc5yr.

Обладнання, що необхідне для проведення занять

Лекційні заняття проводяться в аудиторії, яка обладнана проектором, лабораторні заняття – в комп'ютерній лабораторії. Для дистанційних занять потрібен доступ до Інтернету та програмне й апаратне забезпечення для проведення відеоконференцій.

Навчальний контент

5. Методика опанування навчальної дисципліни (освітнього компонента)

Структура навчальної дисципліни

Назви розділів, тем	*Розподіл навчального часу*				
	Всього	*в тому числі*			
		Лекції	*Практичні (семінарські) заняття*	*Лабораторні роботи (комп'ютерний практикум)*	*СРС*
1	*2*	*3*	*4*	*5*	*6*
Розділ 1. Технології програмування на Python					
Тема 1.1. Знайомство з Python	**8**	2			6
Тема 1.2. Розробка додатків з Flask	**12**	2		4	6
Тема 1.3. Розробка додатків з Django і базами даних	**14**	4		4	6
Тема 1.4. Робота з контейнерами: Docker, Kubernetes і OpenShift	**14**	4		4	6
Тема 1.5. Розробка додатків на основі мікросервісів і безсерверних архітектур	**14**	4		4	6
Тема 1.6. Хмарні обчислення для додатків на Python	**12**	4		2	6
Тема 1.7. Розробка додатків на основі генеративного штучного інтелекту з використанням Python	**8**	2			6
Разом за розділом 1	**82**	**22**	**0**	**18**	**42**
Розділ 2. Безпека та моніторинг додатків					
Тема 2.1. Основи безпеки додатків на Python	**4**	2			2

1	*2*	*3*	*4*	*5*	*6*
Тема 2.2. Ризики безпеки додатків та найкращі практики	**4**	2			2
Тема 2.3. Моніторинг додатків	**4**	2			2
Тема 2.4. Журналювання та спостереження	**4**	2			2
Разом за розділом 2	**16**	**8**	**0**	**0**	**8**
Модульна контрольна робота	**10**				**10**
Залік	**12**	6			**6**
Разом за семестр	**120**	**36**	**0**	**18**	**66**

Лекційні заняття

№ з/п	*Назва теми лекції та перелік основних питань (перелік дидактичних засобів, посилання на літературу та завдання на СРС)*
1	*2*
1	**Тема 1.1. Знайомство з Python** Основи Python, включаючи типи даних, вирази, змінні та структури даних, застосування логіки програмування Python за допомогою розгалуження, циклів, функцій, об'єктів і класів, використання бібліотек Python, доступ до веб-даних за допомогою API та веб-скрапінгу з Python.
2	**Тема 1.2. Розробка додатків з Flask** Кроки та процеси, пов'язані зі створенням додатка Python, створення модулів Python, запуск модульних тестів та пакування додатків, функції Flask і розгортання додатків в Інтернеті за допомогою фреймворку Flask.
3 4	**Тема 1.3. Розробка додатків з Django і базами даних** Бази даних, створення моделі даних зв'язків між сутностями для реляційної бази даних, створення запитів SQL для вставки, вибору, оновлення та видалення даних у базі даних, використання Django ORM для створення об'єктно-орієнтованих баз даних.
5 6	**Тема 1.4. Робота з контейнерами: Docker, Kubernetes і OpenShift** Використання контейнерів, переміщення додатків в будь-якому середовищі, створення власних хмарних додатків за допомогою Docker, Kubernetes, OpenShift і Istio, опис та використання архітектури Kubernetes, щоб налаштувати та використовувати систему управління контейнерами на основі всього життєвого циклу.
7 8	**Тема 1.5. Розробка додатків на основі мікросервісів і безсерверних архітектур** Основи мікросервісів, їхні переваги та контраст з монолітними архітектурами, створення кінцевих точок REST API та їх виклик за допомогою cURL і Postman, використання SwaggerUI для документування та тестування API, створення та розгортання мікросервісів за допомогою контейнерів Docker та безсерверних технологій, як-от IBM Code Engine.

1	2
9 10	**Тема 1.6. Хмарні обчислення для додатків на Python** Основи хмарних обчислень, включаючи їх визначення, моделі обслуговування (IaaS, PaaS, SaaS), моделі розгортання та ключові компоненти інфраструктури, нові тенденції, такі як HybridMulticloud, Microservices, Serverless, Cloud Native, DevOps і Application Modernization, сервіси від основних хмарних платформ, такі як AWS, Azure, Google Cloud, IBM Cloud і Alibaba Cloud.
11	**Тема 1.7. Розробка додатків на основі генеративного штучного інтелекту з використанням Python** Створення генеративних додатків на базі штучного інтелекту для різноманітних випадків використання, робота над практичними проєктами зі створення чат-ботів та додатків з використанням різних генеративних моделей та технологій штучного інтелекту.
12	**Тема 2.1. Основи безпеки додатків на Python** Як безпека вписується в робочий процес, практичні знання про концепції та термінологію безпеки, ключові стратегії пом'якшення ризиків для захисту додатка протягом розробки та впровадження.
13	**Тема 2.2. Ризики безпеки додатків та найкращі практики** Відкритий проєкт забезпечення безпеки веб-додатків (OWASP) та його 10 найголовніших проблем безпеки, найкращі практики написання коду та залежності програмного забезпечення.
14	**Тема 2.3. Моніторинг додатків** Моніторинг додатків, поширені терміни, що використовуються в моніторингу, та чому моніторинг важливий для розробників, синтетичний моніторинг та його важливість.
15	**Тема 2.4. Журналювання та спостереження** Концепція ведення журналу додатків та її важливість, спостереження, його переваги та три основи спостереження.

Лабораторні заняття

№ з/п	*Назва лабораторної роботи (комп'ютерного практикуму)*	*Кількість ауд. годин*
1	*2*	*3*
1	Розробка додатків з Flask (Розділ 1, Тема 1.2)	4
2	Розробка додатків з Django і базами даних (Розділ 1, Тема 1.3)	4
3	Робота з контейнерами: Docker, Kubernetes і OpenShift (Розділ 1, Тема 1.4)	4
4	Розробка додатків на основі мікросервісів і безсерверних архітектур (Розділ 1, Тема 1.5)	4
5	Хмарні обчислення для додатків на Python (Розділ 1, Тема 1.6)	2
	Разом:	**18**

6. Самостійна робота студента

У процесі виконання індивідуальних завдань студенти повинні закріпити знання, отримані під час лекційних занять та лабораторних робіт, самостійно вивчати визначені теми,

поглиблювати свої знання для подальшого навчання. Самостійна робота студентів полягає в наступному:

- підготовці до лекційного заняття шляхом вивчення матеріалу, що розглядався на попередньому лекційному занятті;
- підготовці до лабораторних занять шляхом виконання лабораторних робіт з вивченням теоретичних питань, які розглядаються на лекційних заняттях;
- оформленні протоколу про виконання лабораторних робіт;
- підготовці до залікової роботи.

Політика та контроль

7. Політика навчальної дисципліни (освітнього компонента)

Всі студенти повинні відвідувати лекційні та лабораторні заняття – як при очному навчанні (фізичне відвідування занять в аудиторіях), так і при дистанційному навчанні (віртуальне відвідування онлайн-занять).

Під час занять студенти повинні дотримуватись певних дисциплінарних правил:

- забороняється запізнюватись на заняття;
- не допускаються сторонні розмови або інший шум, що заважає проведенню занять;
- виходити з аудиторії під час заняття допускається лише з дозволу викладача.
- не допускається користування мобільними телефонами та іншими технічними засобами без дозволу викладача;
- не допускається займатися діяльністю, яка прямо не стосується навчальної дисципліни.

Результати виконання лабораторних робіт повинні бути оформлені у електронному форматі у вигляді файлів звітів. Такі файли повинні містити результати виконання відповідних завдань та відповідати вимогам та методичним вказівкам для кожної роботи.

Роботи, які здаються із порушенням термінів, оцінюються на нижчу оцінку.

Лабораторні роботи та модульна контрольна робота перевіряються на наявність плагіату. Наявність плагіату призводить до незадовільної оцінки роботи без можливості перездачі роботи.

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

Підсумкова рейтингова оцінка (R) студента складається з балів, які він отримує:

- за лабораторні роботи ($R_{ЛАБ}$);
- за тести (R_T);
- за модульну контрольну роботу ($R_{МКР}$);
- за залікову роботу ($R_З$).

Відповідно до "Положення про систему оцінювання результатів навчання в КПІ ім. Ігоря Сікорського", затвердженого Наказом №1/273 від 14.09.2020 р., застосована система оцінювання типу РСО-1, для якої рейтингова оцінка складається з:

- стартової оцінки ($R_С$) – оцінювання заходів впродовж семестру: $R_С = R_{ЛАБ} + R_T + R_{МКР}$;
- оцінки за залікову роботу ($R_З$).

Таким чином, $R = R_С + R_З$.

Студенти, які мають стартову оцінку (R_C) не меншу за деяке значення мінімально можливої загальної позитивної оцінки ($R_{МІН}$) та згодні з тим, що їх стартова оцінка (R_C) дорівнюватиме підсумковій рейтинговій оцінці (R), звільняються від написання залікової роботи.

Студенти, які мають стартову оцінку (R_C) меншу за деяке значення мінімально можливої загальної позитивної оцінки ($R_{МІН}$) або бажають виконати залікову роботу, отримують стартову оцінку (R_C), що дорівнює оцінці за лабораторні роботи та модульну контрольну роботу ($R_{ЛАБ} + R_{МКР}$).

Протягом курсу студенти виконують 5 (п'ять) лабораторних робіт, 8 (вісім) тестів та модульну контрольну роботу. Тест вважається контрольним заходом.

Розрахунок шкал оцінювання

Шкала оцінювання (максимально можлива оцінка) лабораторної роботи: 10 балів.
Шкала оцінювання всіх лабораторних робіт: $R_{ЛАБ}$ = 10 балів × 5 = 50 балів.
Шкала оцінювання (максимально можлива оцінка) тесту: 5 балів.
Шкала оцінювання всіх тестів: R_T = 5 балів × 8 = 40 балів.
Шкала оцінювання модульної контрольної роботи: $R_{МКР}$ = 10 балів.
Шкала стартової оцінки: $R_C = R_{ЛАБ} + R_T + R_{МКР}$ = 50 балів + 40 бали + 10 балів = 100 балів.
Шкала оцінювання залікової роботи: $R_З$ = 40 балів.
Підсумкова рейтингова шкала: $R = R_C$ = 100 балів або $R = R_{ЛАБ} + R_{МКР} + R_З$ = 60 балів + 40 балів = 100 балів.
Мінімально можлива загальна позитивна оцінка: $R_{МІН}$ = 60 балів.

Нижня межа позитивного оцінювання кожного контрольного заходу (тест) є не менше 60% від балів, визначених для цього контрольного заходу (5 балів × 0,6 = 3 бали), а негативний результат оцінюється у 0 балів. Якщо студент не проходив або не з'явився на контрольний захід, його результат оцінюється у 0 балів.

Терміни здачі робіт

Термін здачі лабораторних робіт визначається наступним чином:

- лабораторна робота 1 – з 3 (третього) до 6 (шостого) тижня навчання включно;
- лабораторна робота 2 – з 5 (п'ятого) до 8 (восьмого) тижня навчання включно;
- лабораторна робота 3 – з 7 (сьомого) до 10 (десятого) тижня навчання включно;
- лабораторна робота 4 – з 7 (сьомого) до 12 (дванадцятого) тижня навчання включно;
- лабораторна робота 5 – з 7 (сьомого) до 14 (чотирнадцятого) тижня навчання включно.

Термін здачі тестів визначається:

- часом кінця заняття, на якому проводиться тест, у разі очного навчання;
- 2 (двома) днями після дня заняття, на якому проводиться тест, у разі дистанційного навчання.

Тести проводяться на наступних тижнях навчання: 2 (другий), 4 (четвертий), 6 (шостий), 8 (восьмий), 10 (десятий), 11 (одинадцятий), 13 (тринадцятий), 15 (п'ятнадцятий).

Заохочувальні та штрафні бали

Лабораторні роботи, які здаються до початку терміну здачі, оцінюються у заохочувальні +2 (плюс два) бали додатково до максимально можливої оцінки.

Лабораторні роботи, які здаються після терміну здачі без поважних причин, оцінюються у штрафні −2 (мінус два) бали додатково до максимально можливої оцінки.

Визначення умов отримання позитивного результату календарного контролю

Календарний контроль, як правило, проводиться на 7 та 13 тижні навчання. Необхідною умовою отримання позитивного результату календарного контролю є деяке значення поточної оцінки студента (R_{C1}, R_{C2}), яке не менше деякого мінімального значення ($R_{МІНК1}$, $R_{МІНК2}$) відповідно для першого та другого календарного контролю. Мінімальне значення для отримання позитивного результату календарного контролю визначається як 60% від максимально можливої оцінки R_C на час проведення календарного контролю. Тобто, при здачі робіт згідно визначених термінів на час першого календарного контролю повинні бути здані 2 (дві) лабораторні роботи та 3 (три) тести, а на час другого календарного контролю – 4 (чотири) лабораторні роботи та 7 (сім) тестів. Таким чином:

- R_{C1} = 10 балів × 2 + 5 балів × 3 = 20 балів + 15 балів = 35 балів;
- R_{C2} = 10 балів × 4 + 5 балів × 7 = 40 балів + 35 балів = 75 балів;
- $R_{МІНК1}$ = 35 балів × 0,6 = 21 бал;
- $R_{МІНК2}$ = 75 балів × 0,6 = 45 балів.

Тобто, для отримання позитивного результату першого календарного контролю необхідно заробити принаймні 21 бал, для другого календарного контролю – 45 балів.

Визначення умов допуску до заліку

Умовою допуску до заліку є виконання з певною успішністю та необхідної кількості завдань впродовж семестру, а також дотримання вимог академічної доброчесності, що описані в Кодексі честі Національного технічного університету України «Київський політехнічний інститут імені Ігоря Сікорського», ухваленого Рішенням Вченої ради від 5 квітня 2021 р. (Протокол №4). Тобто, студент повинен впродовж семестру заробити деякий ненульовий стартовий рейтинг (R_C > 0), виконати модульну контрольну роботу, принаймні 3 (три) лабораторні роботи та принаймні 4 (чотири) тести.

Мінімальний стартовий рейтинг для допуску до заліку ($R_{МІНДОП}$) визначається як 60% від мінімально можливої загальної позитивної оцінки ($R_{МІН}$). Звідси значення $R_{МІНДОП}$ = $R_{МІН}$ × 0,6 = 60 балів × 0,6 = 36 балів. Таким чином, умовою допуску до заліку буде R_C >= $R_{МІНДОП}$, тобто, значення стартового рейтингу R_C має бути не менше 36 балів.

Підсумкова рейтингова оцінка переводиться в оцінку за університетською шкалою.

Таблиця переведення підсумкової рейтингової оцінки в університетську шкалу оцінок

Підсумкова рейтингова оцінка *R*	Університетська шкала оцінок
95...100	Відмінно
85...94	Дуже добре
75...84	Добре
65...74	Задовільно
60...64	Достатньо
36...59	Незадовільно
0...35	Не допущено

Робочу програму навчальної дисципліни (силабус):

Склав асистент кафедри обчислювальної техніки, доктор філософії Шульга М.В.

Ухвалено кафедрою обчислювальної техніки (протокол № 18 від 24 червня 2026 р.)

Погоджено Методичною комісією факультету інформатики та обчислювальної техніки (протокол № 12 від 26 червня 2026 р.)