МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Тернопільський національний технічний університет
імені Івана Пулюя

Кафедра програмної інженерії

Методичні вказівки для самостійної роботи

з дисципліни

«Розподілені та паралельні системи обчислень»

СЕН ID: **6843**

для здобувачів третього (освітньо-наукового) рівня вищої освіти ОНП
«Інженерія програмного забезпечення»

Тернопіль — 2025

УДК 004.4(075.8)
ББК 32.973.202я73
Б83

Укладач:
В.М. Бревус, PhD

Рецензент:
В.В. Яцишин, к.т.н., доцент
завідувач кафедри систем штучного інтелекту та аналізу даних ТНТУ ім. І. Пулюя

Розглянуто й затверджено на засіданні кафедри програмної інженерії.
Протокол № 1 від 01.09.2025.

Розглянуто й затверджено на засіданні методичної комісії факультету комп'ютерно-інформаційних систем та програмної інженерії Тернопільського національного технічного університету імені Івана Пулюя.
Протокол № 1 від 01.09.2025.

Б83 Методичні вказівки для самостійної роботи з дисципліни «Розподілені та паралельні системи обчислень» для здобувачів третього (освітньо-наукового) рівня вищої освіти ОНП «Інженерія програмного забезпечення» / Укладач: Бревус В.М. – Тернопіль: ТНТУ імені Івана Пулюя, 2025 – 24 с.

Відповідальний за випуск: М.Р. Петрик, докт. фіз.-мат. наук, професор

АНОТАЦІЯ

Ці методичні вказівки для самостійної роботи з дисципліни «Розподілені та паралельні системи обчислень» формують у здобувачів вищої освіти фахові компетентності та дослідницькі здібності, необхідні для розв'язання задач проєктування та розробки розподілених систем обчислень.

Практикум базується на комплексному наскрізному проєкті, що охоплює весь цикл розробки розподіленої системи обробки даних — від вивчення фундаментальних принципів (теорема CAP, моделі узгодженості, відмовостійкість) до практичної реалізації з використанням Apache Spark і PySpark. Методичні вказівки орієнтовані на практичне засвоєння здобувачами навичок роботи з паралельною обробкою великих даних, розробки моделей машинного навчання в розподіленому середовищі та контейнеризації аналітичних застосунків.

Документ призначений для здобувачів третього (освітньо-наукового) рівня вищої освіти ОНП «Інженерія програмного забезпечення» та визначає порядок організації самостійної роботи при виконанні практичних робіт. Містить детальні завдання, теоретичне підґрунтя, методичні рекомендації та вимоги до оформлення результатів. Буде корисним також для аспірантів, науковців та інженерів, що спеціалізуються в галузі розподілених систем та паралельних обчислень.

Ключові слова: розподілені системи, паралельні обчислення, Apache Spark, PySpark, теорема CAP, машинне навчання, контейнеризація, методичні вказівки.

ABSTRACT

These methodological guidelines for independent study of the discipline “Distributed and Parallel Computing Systems” develop professional competencies and research abilities in students of higher education necessary for solving tasks of designing and developing distributed computing systems.

The practicum is based on a comprehensive end-to-end project covering the entire development cycle of a distributed data processing system — from studying fundamental principles (CAP theorem, consistency models, fault tolerance) to practical implementation using Apache Spark and PySpark. The guidelines are oriented toward the practical mastery of students’ skills in parallel processing of large datasets, development of machine learning models in distributed environments, and containerization of analytical applications.

The document is intended for students of the third (educational and scientific) level of higher education in the “Software Engineering” educational and scientific program and defines the organization of independent work during practical activities. It contains detailed tasks, theoretical background, methodological recommendations, and requirements for presenting results. It will also be useful for postgraduate students, researchers, and engineers specializing in distributed systems and parallel computing.

Keywords: distributed systems, parallel computing, Apache Spark, PySpark, CAP theorem, machine learning, containerization, methodological guidelines.

Зміст

Вступ

Дисципліна «Розподілені та паралельні системи обчислень» присвячена розвитку навичок та компетенцій у проєктуванні та розробці розподілених систем обчислень на основі сучасних фреймворків та інструментів. Цей практикум надає здобувачам освітньо-наукового рівня вищої освіти можливість набути глибокого розуміння теоретичних концепцій та практичних навичок через комплексний наскрізний проєкт, який охоплює весь цикл розробки системи паралельної обробки даних — від вивчення фундаментальних принципів до практичної реалізації, оптимізації та розгортання в хмарному середовищі.

Призначення методичних вказівок

Ці методичні вказівки розраховані на здобувачів третього (освітньо-наукового) рівня вищої освіти ОНП «Інженерія програмного забезпечення» та визначають порядок організації самостійної роботи при виконанні практичних робіт. Вказівки містять детальні завдання, теоретичне підґрунтя, методичні рекомендації та вимоги до оформлення результатів роботи. Матеріали курсу доступні в СЕН ТНТУ ім. І. Пулюя ID6843.

Усі практичні роботи повинні бути виконані з дотриманням принципів академічної доброчесності, відповідно до Положення про академічну доброчесність учасників освітнього процесу Тернопільського національного технічного університету імені Івана Пулюя.

Метою вивчення навчальної дисципліни «Розподілені та паралельні системи обчислень»

є формування у здобувачів глибокого, системного уявлення про фундаментальні принципи, архітектуру та методи проєктування розподілених та паралельних систем. Курс охоплює теоретичні основи розподілених обчислень (теорема CAP, моделі узгодженості, відмовостійкість) та їх практичну реалізацію з використанням Apache Spark і PySpark для паралельної обробки великих даних, машинного навчання в розподіленому середовищі, а також сучасні підходи до контейнеризації та хмарного розгортання.

Завдання навчальної дисципліни

За результатами вивчення дисципліни здобувач повинен продемонструвати такі результати навчання:

Завдання навчальної дисципліни

За результатами вивчення дисципліни здобувач повинен продемонструвати такі результати навчання:

Знати:

1. Фундаментальні принципи та архітектурні моделі розподілених та паралельних систем обчислень.
2. Теорему CAP, моделі узгодженості даних (eventual consistency, strong consistency) та їх застосування.
3. Принципи забезпечення відмовостійкості, реплікації та масштабованості в розподілених системах.
4. Архітектуру Apache Spark та принципи лінивих обчислень (lazy evaluation).
5. Основні структури даних Spark: RDD, DataFrame та Dataset, їх переваги та області застосування.
6. Принципи роботи Spark SQL та оптимізації запитів в розподіленому середовищі.
7. Методи очищення, трансформації та агрегації даних з використанням PySpark.
8. Алгоритми машинного навчання в MLlib та побудову конвеєрів обробки даних (pipelines).
9. Техніки оптимізації Spark-застосунків та моніторингу продуктивності.
10. Принципи контейнеризації аналітичних застосунків з Docker та розгортання в хмарному середовищі.

Вміти:

1. Аналізувати архітектурні рішення розподілених систем з точки зору теореми CAP та вибирати відповідні моделі узгодженості.
2. Проєктувати відмовостійкі та масштабовані розподілені системи з урахуванням вимог до продуктивності.
3. Створювати та маніпулювати RDD та DataFrame в PySpark для обробки великих наборів даних.
4. Застосовувати трансформації та дії (actions) для ефективної обробки розподілених даних.
5. Розробляти власні функції (UDF) та інтегрувати їх у конвеєри обробки даних.
6. Будувати моделі машинного навчання з використанням Spark MLlib для класифікації, регресії та кластеризації.
7. Оптимізувати продуктивність Spark-застосунків через налаштування конфігурації та архітектури.
8. Контейнеризувати та розгортати аналітичні застосунки з використанням Docker та хмарних платформ.
9. Проводити моніторинг та налагодження розподілених обчислень в реальному часі.

Вивчення навчальної дисципліни передбачає підсилення та розвиток у здобувачів компетентностей:

- **Інтегральна компетентність.** Здатність продукувати нові ідеї, розв'язувати комплексні проблеми професійної та/або дослідницько-інноваційної діяльності у сфері інженерії програмного забезпечення та з дотичних до неї міждисциплінарних напрямах, застосовувати методологію наукової та педагогічної діяльності, проводити власне наукове дослідження, результати якого мають наукову новизну, теоретичне та практичне значення.
- **ЗК02.** Здатність розв'язувати комплексні проблеми у сфері інженерії програмного забезпечення та з дотичних до неї міждисциплінарних напрямах на основі системного наукового світогляду та загального культурного кругозору із дотриманням принципів професійної етики та академічної доброчесності.
- **ФК01.** Здатність інтегрувати знання з різних галузей, застосовувати системний підхід та враховувати нетехнічні аспекти при розв'язанні комплексних проблем інженерії програмного забезпечення й проведенні досліджень.
- **ФК05.** Здатність до розроблення нових та вдосконалення існуючих моделей, методів, засобів, процесів у сфері інженерії програмного забезпечення, які забезпечують розвиток або надають нові можливості технологіям розробки та супроводження програмного забезпечення.
- Здатність розробляти розподілені та багатопотокові застосунки.

Програмні результати:

- **ПРН03.** Пропонувати нові ефективні методи і моделі розроблення, впровадження, супроводу та забезпечення якості програмного забезпечення та управління відповідними процесами на всіх етапах життєвого циклу.
- **ПРН07.** Розробляти та досліджувати концептуальні, математичні і комп'ютерні моделі процесів і систем для отримання нових знань та/або створення інноваційних продуктів у інженерії програмного забезпечення та дотичних міждисциплінарних напрямах.
- **ПРН09.** Формулювати та вирішувати задачі оптимізації, адаптації, прогнозування, керування та прийняття рішень щодо процесів, засобів та ресурсів розробки, впровадження, супроводу та експлуатації програмного забезпечення.

Завданням лекційних занять є ознайомлення здобувачів з фундаментальними принципами розподілених та паралельних систем (теорема CAP, моделі узгодженості, відмовостійкість), архітектурою Apache Spark, принципами розподіленої обробки даних, алгоритмами машинного навчання в MLlib та сучасними підходами до оптимізації та розгортання аналітичних застосунків.

Завдання практичних занять полягає у практичному засвоєнні здобувачами навичок роботи з PySpark для обробки та аналізу великих наборів даних, розробки моделей машинного навчання та контейнеризації застосунків для хмарного розгортання.

1. ТЕМАТИЧНИЙ ПЛАН НАВЧАЛЬНОЇ ДИСЦИПЛІНИ

Дисципліна: Розподілені та паралельні системи обчислень.
Рівень: третій (освітньо-науковий).
Спеціальність: 121 Інженерія програмного забезпечення.
Форма контролю: залік.
Обсяг: 4.5 ECTS (135 годин): 48 год. аудиторних (24 лекції, 24 практичні), 87 год. самостійна робота.

При вивченні дисципліни здобувач ознайомлюється з фундаментальними принципами, архітектурою та методами проєктування розподілених та паралельних систем. Тематичний план складається з двох модулів, що охоплюють теоретичні основи розподілених обчислень (теорема CAP, моделі узгодженості, відмовостійкість) та їх практичну реалізацію з використанням Apache Spark і PySpark для паралельної обробки великих даних, машинного навчання в розподіленому середовищі, а також сучасні підходи до контейнеризації та хмарного розгортання.

1.1 Структура змістових модулів

Модуль 1 формує базу: фундаментальні принципи розподілених систем, архітектуру Apache Spark, роботу з DataFrame та Spark SQL, інженерію ознак та очищення даних. Модуль 2 поглиблює: машинне навчання з MLlib, застосування моделей регресії та класифікації, налагодження та оптимізацію Spark-застосунків, розгортання в різних середовищах.

1.2 Тематичний план (лекції)

№	Тема заняття та короткий зміст	ДФН	ЗФН
	Модуль 1		
1	**Тема 1. Фундаментальні принципи розподілених систем.** Архітектурні моделі, теорема CAP, моделі узгодженості.	2	
2	**Тема 2. Вступ до Apache Spark та PySpark.** Архітектура Spark, RDD, ліниві обчислення.	2	
3	**Тема 3. Робота з DataFrame та Spark SQL.** Створення, трансформації, дії (actions), використання SQL-запитів.	4	
4	**Тема 4. Інженерія ознак та очищення даних.** Робота з різними типами даних, об'єднання наборів, очищення.	4	
	Модуль 2		
5	**Тема 5. Вступ до машинного навчання з MLlib.** Побудова конвеєрів (pipelines), основні алгоритми.	4	
6	**Тема 6. Застосування моделей регресії та класифікації.** Практичні приклади та оцінка моделей.	4	
7	**Тема 7. Налагодження та оптимізація Spark-застосунків.** Моніторинг, профілювання, техніки оптимізації.	2	
8	**Тема 8. Розгортання Spark-застосунків.** Режими розгортання (Standalone, YARN, Kubernetes).	2	
	Усього годин	**24**	

1.3 Практичні заняття (прикладний наскрізний проєкт)

Практичні роботи організовані навколо наскрізного проєкту **«Система аналізу великих даних з використанням Apache Spark»**, що базується на роботі з реальними наборами даних для розподіленої обробки, машинного навчання та розгортання в хмарному середовищі.

№	Тема заняття та короткий зміст	ДФН	ЗФН
1	**ПР№1. Основи роботи з PySpark та аналіз даних.**	6	
2	**ПР№2. Інженерія ознак та побудова конвеєра обробки.**	6	
3	**ПР№3. Побудова та оцінка моделі машинного навчання.**	6	
4	**ПР№4. Оптимізація та інтерпретація результатів.**	6	
	Усього годин	**24**	

1.4 Самостійна робота

Рекомендовані напрями самостійного опрацювання узгоджені з логікою поглиблення матеріалу.

№	Найменування робіт	ДФН	ЗФН
1	Алгоритми консенсусу: Paxos та Raft. Аналіз ключових наукових праць.	6	
2	Порівняльний аналіз архітектур потокової обробки: Spark Streaming vs. Apache Flink.	6	
3	Паралелізм даних та моделей у розподіленому навчанні глибоких мереж (Horovod).	6	
4	Внутрішня архітектура Spark: оптимізатори Catalyst та Tungsten.	7	
5	Сучасні хмарні патерни: Service Mesh та Serverless (FaaS) для розподілених систем.	7	
6	Підготовка до практичних занять	30	
	Усього годин	**62**	

1.5 Очікувані результати опанування тематичного плану

Після завершення тем здобувач повинен:

- розуміти фундаментальні принципи та архітектурні моделі розподілених систем;
- аналізувати архітектурні рішення з точки зору теореми CAP та моделей узгодженості;
- створювати та маніпулювати RDD та DataFrame в PySpark;
- застосовувати трансформації та дії для ефективної обробки розподілених даних;
- розробляти власні функції (UDF) та інтегрувати їх у конвеєри обробки даних;

- будувати моделі машинного навчання з використанням Spark MLlib;
- оптимізувати продуктивність Spark-застосунків через налаштування конфігурації;
- контейнеризувати та розгортати аналітичні застосунки з використанням Docker;
- проводити моніторинг та налагодження розподілених обчислень.

1.6 Основні компоненти наскрізного проєкту

Дані: великі набори даних для розподіленої обробки (CSV, JSON, Parquet).
Проміжні артефакти: Spark-сесія, модулі ETL, конвеєри обробки ознак, моделі MLlib, контейнеризований застосунок.
Методичні акценти: ефективність розподіленої обробки, оптимізація продуктивності, масштабованість рішення.

1.7 Короткі критерії оцінювання

- Повнота виконання практичних етапів (код + пояснення).
- Ефективність використання Spark-функціоналу та оптимізації.
- Якість побудованих моделей машинного навчання та їх валідації.
- Структурованість коду та якість документації.
- Академічна доброчесність (оригінальність реалізації, коректні посилання).

1.8 Рекомендовані інструменти

Python, PySpark, Apache Spark, Jupyter Notebooks, Docker, Spark MLlib, pandas (для локальної обробки), matplotlib та seaborn (для візуалізації), при розширенні — Kubernetes для оркестрації контейнерів у хмарному середовищі.

1.9 Базові та допоміжні джерела

Перелік літератури й ресурсів наведено у відповідному розділі видання (див. повну робочу програму). Рекомендовано розпочати з Kleppmann «Designing Data-Intensive Applications» (2017) та Chambers & Zaharia «Spark: The Definitive Guide» (2018).

1.10 Примітки щодо адаптації плану

Тематичний план може коригуватися (оновлення практичних кейсів, модернізація інструментарію) за рішенням кафедри з фіксацією змін у протоколах.

2. ЗАГАЛЬНІ РЕКОМЕНДАЦІЇ ДО ОРГАНІЗАЦІЇ САМОСТІЙНОЇ РОБОТИ З ДИСЦИПЛІНИ

Обов'язковим елементом успішного засвоєння навчального матеріалу дисципліни «Розподілені та паралельні системи обчислень» є самостійна робота здобувачів з вітчизняною і зарубіжною літературою.

Самостійна робота є основним засобом оволодіння навчальним матеріалом у час, вільний від нормованих навчальних занять, тобто лекційних і практичних. Особливо важливою є самостійна робота з дослідження сучасних наукових публікацій у галузі розподілених систем, алгоритмів консенсусу та технологій великих даних.

2.1 Основні види самостійної роботи

На які повинні звертати увагу здобувачи:

- вивчення лекційного матеріалу з фундаментальних принципів розподілених систем;
- опрацювання та вивчення рекомендованої наукової літератури та актуальних статей;
- підготовка до практичних занять з PySpark та Apache Spark;
- виконання практичних завдань з обробки великих даних та машинного навчання;
- дослідження алгоритмів консенсусу (Paxos, Raft) та сучасних архітектур;
- аналіз кейсів хмарних платформ та контейнерних технологій;
- підготовка до поточного та підсумкового контролю.

2.2 Опрацювання лекційного матеріалу

У системі різних форм навчально-виховної роботи особливе місце належить лекції, де викладач надає здобувачу основну інформацію, навчає розмірковувати, аналізувати, допомагає опанувати ключові знання, а також спрямовує самостійну роботу здобувача.

Зв'язок лекції і самостійної роботи здобувача розглядається в таких напрямах:

- лекція як головна початкова ланка, що визначає зміст і обсяг самостійної роботи здобувача;
- методичні прийоми читання лекцій, що активізують самостійну роботу здобувачів;
- самостійна робота, яка сприяє поглибленому засвоєнню теми на базі прослуханої лекції.

Перший етап самостійної роботи починається з процесу слухання і записування лекції. Правильно складений конспект лекції — найефективніший засіб стимулювання подальшої самостійної роботи здобувачів. Здобувач повинен чітко усвідомити, що конспект — це короткий тезовий запис головних положень навчального матеріалу. Складання і вивчення конспекту — перший етап самостійної роботи здобувача над вивченням теми чи розділу. Конспект допомагає в раціональній підготовці до практичних занять, заліку, у визначенні напряму і обсягу подальшої роботи з літературними джерелами.

Під час підготовки до лекції здобувач повинен опрацювати матеріал попередної лекції з використанням підручників та інших джерел літератури. На лекціях висвітлюють тільки основні теоретичні положення та найбільш актуальні проблеми, тому більшість питань виноситься на самостійне опрацювання.

2.3 Підготовка до практичних занять

Підготовка до практичних занять розпочинається з опрацювання лекційного та методичного матеріалу до заданого заняття. Здобувач повинен самостійно ознайомитися з відповідним розділом робочої програми, підготувати відповіді на контрольні запитання, які подані в програмі у певній послідовності згідно з логікою засвоєння навчального матеріалу.

Практичні роботи збагачують і закріплюють теоретичні знання здобувачів, розвиваючи їхню творчу активність, допомагають у набутті практичних навичок роботи за предметом навчальної дисципліни.

У процесі підготовки до практичних робіт самостійна робота здобувачів є обов'язковою частиною навчальної роботи, без якої успішне і якісне засвоєння навчального матеріалу неможливе. Це свідчить про необхідність керування самостійною роботою здобувачів з боку викладача завдяки проведенню цілеспрямованих організаційних і контрольних заходів.

Викладач у вступній лекції рекомендує здобувачам основну і додаткову літературу, а також методичні рекомендації до самостійної роботи та до організації практичних робіт з дисципліни. У методичних вказівках з кожної теми наведено перелік питань для теоретичної підготовки до заняття.

У разі, коли здобувач не може самостійно розібратися в якомусь питанні, він може отримати консультацію у викладача (згідно з графіком проведення консультацій викладачами кафедри). Добре організовані консультації дозволяють спрямувати самостійну роботу в потрібному напрямі, зробити раціональною і підвищити її ефективність.

2.4 Структурована програма самостійної роботи

№	Найменування розділів самостійної роботи	ДФН	ЗФН
1	Алгоритми консенсусу: Paxos та Raft. Аналіз ключових наукових праць.	6	–
2	Порівняльний аналіз архітектур потокової обробки: Spark Streaming vs. Apache Flink.	6	–
3	Паралелізм даних та моделей у розподіленому навчанні глибоких мереж (Horovod).	6	–
4	Внутрішня архітектура Spark: оптимізатори Catalyst та Tungsten.	7	–
5	Сучасні хмарні патерни: Service Mesh та Serverless (FaaS) для розподілених систем.	7	–
6	Підготовка до практичних занять	30	–
7	Підготовка до підсумкового контролю	25	–
	Разом	**87**	–

2.5 Рекомендована послідовність роботи

Для кожного модуля:

1. **Етап 1 — Підготовка до лекцій.** Прочитайте стислий огляд теми з рекомендованої літератури перед лекцією, щоб отримати загальне розуміння матеріалу.
2. **Етап 2 — Слухання лекції та конспектування.** Активно слухайте лекцію, записуйте ключові концепції, приклади та висновки викладача.
3. **Етап 3 — Опрацювання конспекту.** Відразу після лекції переглянете конспект, уточніть незрозумілі місця, доповніть конспект інформацією з посилань.
4. **Етап 4 — Поглиблене вивчення теми.** Прочитайте відповідні розділи підручників та монографій з рекомендованого списку літератури, особливу увагу приділіть актуальним науковим статтям з розподілених систем.
5. **Етап 5 — Підготовка до практичного заняття.** Встановіть необхідне програмне забезпечення (PySpark, Docker), ознайомтеся з документацією Apache Spark, підготуйте відповіді на контрольні запитання.
6. **Етап 6 — Виконання практичної роботи.** Активно беріть участь у практичному занятті з PySpark, виконуйте завдання з обробки великих даних та машинного навчання, задавайте питання.

7. **Етап 7 — Закріплення матеріалу.** Експериментуйте з різними налаштуваннями Spark-застосунків, досліджуйте додаткові бібліотеки та інструменти для розширення розуміння теми.

8. **Етап 8 — Підготовка до модульного контролю.** Повторіть весь матеріал модуля, розв'яжіть тестові завдання, готуйтеся до тестування.

2.6 Поради щодо ефективної самостійної роботи

- **Планування часу.** Розподіліть самостійну роботу рівномірно протягом семестру, уникайте «штурму» перед іспитом.
- **Організація робочого простору.** Створіть умови для концентрації уваги: чистий стіл, мінімум відволікань, необхідні матеріали.
- **Активне читання.** Не просто читайте текст, але й робіть нотатки, складайте схеми, створюйте опорні конспекти.
- **Практичне застосування.** Намагайтеся застосовувати теоретичні знання до розв'язання практичних задач з обробки великих даних, експериментуйте з різними алгоритмами машинного навчання в MLlib.
- **Взаємодія з однолітками.** Обговорюйте складні питання розподілених систем з одногрупниками, створюйте спільні проєкти з Apache Spark, діліться досвідом налагодження та оптимізації.
- **Консультації з викладачем.** Не стримуйтеся звертатися за допомогою, коли виникають труднощі в розумінні матеріалу.
- **Регулярне повторення.** Повторюйте раніше вивчений матеріал, щоб закріпити знання та запобігти їх забуванню.

4. СИСТЕМА ПОТОЧНОГО Й ПІДСУМКОВОГО КОНТРОЛЮ ЗНАНЬ ЗДОБУВАЧІВ

4.1 Форми контролю

Оцінювання знань, вмінь і навичок здобувачів включає ті види занять, які згідно з програмою навчальної дисципліни «Розподілені та паралельні системи обчислень» передбачають лекційні, практичні роботи, самостійну роботу.

Перевірку і оцінювання знань здобувачів проводять в наступних формах:

- оцінювання роботи і знань здобувачів під час практичних занять;
- оцінювання виконання і захист практичних робіт;
- складання проміжного контролю знань за змістовими модулями (модульний контроль);
- здача залікового тестування.

4.2 Модульний контроль

Для кожного змістовного модуля передбачено певну форму поточного контролю. Результати поточного контролю автоматично, без участі здобувача, зараховуються при модульному контролі.

Максимальна оцінка при I модульному контролі — 40 балів:

- Теоретичний курс (тестування): 20 балів
- Практична робота: 20 балів

Максимальна оцінка при II модульному контролі — 35 балів:

- Теоретичний курс (тестування): 20 балів
- Практична робота: 15 балів

4.3 Підсумковий контроль

Підсумковий контроль здійснюється у формі залік з максимальною оцінкою 25 балів.

Максимальна оцінка навчальної дисципліни — 100 балів.

Розподіл балів між модулями:

- Модуль 1: 40 балів
- Модуль 2: 35 балів
- Залік: 25 балів

4.4 Критерії оцінювання за компонентами

Теоретичний курс (тестування): Оцінюється глибина розуміння теоретичного матеріалу, знання основних концепцій ідентифікації систем, методів аналізу та моделювання.

Практичні роботи: Оцінюється якість виконання практичних завдань, прикладне застосування теоретичних знань, коректність результатів та обґрунтованість висновків.

5. КРИТЕРІЇ ОЦІНЮВАННЯ РЕЗУЛЬТАТІВ НАВЧАННЯ ЗДОБУВАЧІВ

5.1 Шкала оцінювання

Сума балів за навчальну діяльність	Оцінка ECTS	Оцінка за національною шкалою
90 – 100	A	Відмінно
82–89	B	Добре
74–81	C	Добре
64–73	D	Задовільно
60–63	E	Задовільно
35–59	FX	Незадовільно з можливістю повторного складання
0–34	F	Незадовільно з обов'язковим повторним вивченням дисципліни

5.2 Таблиця розподілу балів

Модуль 1		**Модуль 2**	
Аудиторна та самостійна робота		Аудиторна та самостійна робота	
Теоретичний курс (тестування)	Практична робота	Теоретичний курс (тестування)	Практична робота
20	**20**	**20**	**15**
Підсумковий контроль (залік): 25			
Разом з дисципліни: 100			

5.3 Деталізація оцінювання за модулями

Модуль 1			**Модуль 2**		
Тема	Вид робіт	К-ть балів	Тема	Вид робіт	К-ть балів
Тема 1-2	ПР№1	10	Тема 5-6	ПР№3	5
Тема 3-4	ПР№2	10	Тема 7-8	ПР№4	10
Тестування: 20 балів			Тестування: 20 балів		

5.4 Критерії виставлення оцінок

Оцінка «A» (90–100 балів):

- Глибоке розуміння фундаментальних принципів розподілених систем та теореми CAP
- Безпомилково виконані всі практичні роботи з PySpark та Apache Spark
- Логічне та обґрунтоване розв'язання складних задач з обробки великих даних
- Демонстрація аналітичних здібностей та творчого підходу до оптимізації

Оцінка «B» (82–89 балів):

- Добре розуміння архітектури Apache Spark та принципів розподілених обчислень
- Якісне виконання практичних робіт з машинного навчання в MLlib з незначними помилками
- Коректне застосування моделей узгодженості та відмовостійкості
- Достатня глибина та точність у розробці аналітичних застосунків

Оцінка «C» (74–81 балів):

- Задовільне розуміння основних тем курсу
- Виконання практичних робіт з деякими помилками
- Розв'язання стандартних задач без істотних проблем
- Задовільне обґрунтування відповідей

Оцінка «D–E» (60–73 балів):

- Мінімальне розуміння основних концепцій
- Практичні роботи виконані з значними прогалинами
- Розв'язання простих задач з помилками
- Слабке обґрунтування відповідей

Оцінка «FX» (35–59 балів):

- Неповне розуміння теоретичного матеріалу
- Суттєві недоліки у виконанні практичних робіт
- Помилки при розв'язанні базових задач
- Можливість повторного складання або доопрацювання

Оцінка «F» (0–34 балів):

- Неадекватне розуміння основних концепцій
- Неякісне виконання практичних робіт
- Неспроможність розв’язати навіть прості задачі
- Вимагає обов’язкового повторного вивчення дисципліни

6. ПЕРЕЛІК КОНТРОЛЬНИХ ЗАПИТАНЬ З ДИСЦИПЛІНИ

6.1 Запитання з Модулю 1

1. Що таке розподілена система? Наведіть приклади практичного застосування.
2. Сформулюйте теорему CAP та поясніть її значення для проєктування систем.
3. Які моделі узгодженості даних ви знаєте? Опишіть eventual consistency.
4. Що таке відмовостійкість у розподілених системах? Основні підходи.
5. Поясніть архітектуру Apache Spark та принцип лінивих обчислень.
6. Яка різниця між RDD, DataFrame та Dataset у Spark?
7. Опишіть основні трансформації та дії (actions) у PySpark.
8. Як працює Spark SQL та які його переваги?
9. Назвіть основні методи очищення даних у PySpark.
10. Що таке інженерія ознак та як вона реалізується в Spark?

6.2 Запитання з Модулю 2

11. Які основні алгоритми машинного навчання доступні в MLlib?
12. Як створити та використати конвеєр (pipeline) обробки даних?
13. Поясніть принципи класифікації та регресії в розподіленому середовищі.
14. Що таке кластеризація та як вона реалізується в Spark?
15. Назвіть методи оцінки якості моделей машинного навчання.
16. Як розробляти власні функції (UDF) у PySpark?
17. Опишіть техніки оптимізації продуктивності Spark-застосунків.
18. Що таке моніторинг та профілювання в Apache Spark?
19. Які режими розгортання Spark-застосунків ви знаєте?
20. Як використовувати Docker для контейнеризації аналітичних застосунків?

6.3 Спеціальні питання для практичних робіт

21. Як здійснюється основна робота з PySpark для аналізу даних?
22. Назвіть етапи інженерії ознак та побудови конвеєра обробки.
23. Як проводити побудову моделі машинного навчання у розподіленому середовищі?
24. Поясніть процес оптимізації та інтерпретації результатів.
25. Які алгоритми консенсусу Paxos та Raft ви знаєте?
26. Як порівнювати архітектури Spark Streaming та Apache Flink?
27. Що таке паралелізм даних та моделей у навчанні глибоких мереж?
28. Опишіть внутрішню архітектуру Spark: Catalyst та Tungsten.
29. Які сучасні хмарні патерни Service Mesh та Serverless?
30. Як забезпечити масштабованість розподілених систем?

7. РЕКОМЕНДОВАНА ЛІТЕРАТУРА

7.1 Базова література

1. Kleppmann, M. (2017). *Designing Data-Intensive Applications: The Big Ideas Behind Reliable, Scalable, and Maintainable Systems*. O'Reilly Media.

2. Chambers, B., & Zaharia, M. (2018). *Spark: The Definitive Guide. Big Data Processing Made Simple*. O'Reilly Media.

3. Tanenbaum, A. S., & van Steen, M. (2016). *Distributed Systems: Principles and Paradigms* (3rd ed.). Pearson.

4. Kane, S. P., & Matthias, K. (2018). *Docker: Up & Running*. O'Reilly Media.

5. Lynch, N. A. (1996). *Distributed Algorithms*. Morgan Kaufmann.

7.2 Допоміжна література

6. Lamport, L. (1998). *The Part-Time Parliament*. ACM Transactions on Computer Systems.

7. Ongaro, D., & Ousterhout, J. (2014). *In Search of an Understandable Consensus Algorithm (Raft)*. USENIX ATC.

8. Karau, H., Konwinski, A., Wendell, P., & Zaharia, M. (2015). *Learning Spark*. O'Reilly Media.

9. Abadi, D. J. (2018). *The Case for Serverless Data Systems*. arXiv.

10. Apache Spark Documentation. *Best Practices and Optimization Techniques*. – https://spark.apache.org/docs/latest/tuning.html

7.3 Інформаційні ресурси та електронні джерела

11. Apache Spark. Офіційна документація. – https://spark.apache.org/docs/latest/

12. PySpark API Documentation. – https://spark.apache.org/docs/latest/api/python/

13. Docker. Офіційна документація. – https://docs.docker.com/

14. Kubernetes. Офіційна документація. – https://kubernetes.io/docs/

15. GitHub: Awesome Apache Spark. – https://github.com/awesome-spark/awesome-spark

16. Databricks Community Edition: Free Spark Environment. – https://community.cloud.databricks.com/

17. DataCamp: Big Data with PySpark Track. – https://www.datacamp.com/tracks/big-data-with-pyspark

7.4 Матеріали дисципліни

- Електронний курс з дисципліни «Розподілені та паралельні системи обчислень» доступний у СЕН ТНТУ ім. І. Пулюя, ID: 6843
- Конспект лекцій з дисципліни «Розподілені та паралельні системи обчислень» для здобувачів третього (освітньо-наукового) рівня вищої освіти ОНП «Інженерія програмного забезпечення» / Укладач: Бревус В.М. – Тернопіль: ТНТУ імені Івана Пулюя, 2025 – 74 с.
- Збірник практичних робіт з дисципліни «Розподілені та паралельні системи обчислень» для здобувачів третього (освітньо-наукового) рівня вищої освіти ОНП «Інженерія програмного забезпечення» / Укладач: Бревус В.М. – Тернопіль: ТНТУ імені Івана Пулюя, 2025 – 32 с.

7.5 Рекомендований список для поглибленого вивчення

- **Для теоретичної підготовки:** Матеріали конспекту лекцій та класичні підручники з розподілених систем, теореми CAP та алгоритмів консенсусу.
- **Для практичного застосування:** Репозиторії на GitHub з готовими реалізаціями Apache Spark, порівняльні дослідження архітектур та case studies з великих даних.
- **Для розробки проєктів:** Датасети для обробки великих даних, Python екосистема (PySpark, MLlib, pandas), Docker та Kubernetes для контейнеризації.
- **Для актуальної інформації:** Наукові журнали та конференції з розподілених систем (IEEE, ACM, USENIX), блоги Databricks та Apache Foundation.
- **Для спеціалізації у хмарних технологіях:** Документація AWS, Azure, Google Cloud Platform, матеріали про Service Mesh та Serverless архітектури.

Зазначені матеріали можна знайти у:

- Бібліотеці ТНТУ ім. І. Пулюя;
- Електронних базах даних Інтернету;
- Науковому каталогу УДКон (UNILIB);
- Світовій системі цитування Google Scholar.