

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»

Інститут комп'ютерних технологій, автоматики та метрології (ІКТА)
/назва навчально-наукового інституту/

Кафедра електронних обчислювальних систем (ЕОМ)
/назва /

«ЗАТВЕРДЖУЮ»

Голова науково-методичної комісії
спеціальності 123 “Комп'ютерна інженерія”
/назва /

_____/_____
/підпис/ /ініціали та прізвище /
Протокол від « » _____ р. №

РОБОЧА ПРОГРАМА НАВЧАЛЬНОЇ ДИСЦИПЛІНИ

	СК1.18. Алгоритми та моделі обчислень
	<small>/код і назва навчальної дисципліни/</small>
	бакалавр
	<small>/рівень вищої освіти/</small>
вид дисципліни	обов'язкова
	<small>(обов'язкова / за вибором)</small>
мова викладання	українська
освітня програма	ОПП “Комп'ютерна інженерія”
	<small>/назва/</small>
галузь знань	12 “Інформаційні технології”
	<small>/шифр і назва/</small>
спеціальність	123 “Комп'ютерна інженерія”
	<small>/шифр і назва /</small>

Львів – 2025 рік

Робоча програма з навчальної дисципліни Алгоритми та моделі обчислень для
/назва /
здобувачів освіти за освітньою програмою ОПП “Комп’ютерна інженерія”
/ назва освітньої програми /

Розробники:

ст. вик. каф. ЕОМ _____ / Н.Б. Козак /
/посада, науковий ступінь та вчене звання/ /підпис/ /ініціали та прізвище/

Гарант освітньої програми _____ / Є.Я. Ваврук /
/підпис/ /ініціали та прізвище/

Робоча програма розглянута та схвалена на засіданні кафедри ЕОМ
/назва/

Протокол від «_» _____ року №

Завідувач кафедри ЕОМ _____ / _____ /
/назва / /підпис/ /ініціали та прізвище /

1. Структура навчальної дисципліни

Найменування показників	Всього годин	
	Денна форма навчання	Заочна форма навчання
Кількість кредитів/год	6/180	-
Усього годин аудиторної роботи, у т.ч.:	90	-
• лекційні заняття, год.	45	-
• семінарські заняття, год.	-	-
• практичні заняття, год.	15	-
• лабораторні заняття, год.	30	-
Усього годин самостійної роботи, у т.ч.:	45	-
• контрольні роботи, к-сть/год.	-	-
• розрахункові (розрах.-графічні) роботи, к-сть/год.	-	-
• індивідуальне наук.-досл. завдання, к-сть/год.	-	-
• підготовка до навч. занять та контр. заходів, год.	45	-
Екзамен	+	-
Залік	-	-

Частка аудиторного навчального часу студента у відсотковому вимірі:
денної форми навчання – 50% ; заочної форми навчання –

2. Мета та завдання навчальної дисципліни

2.1. Мета вивчення навчальної дисципліни та результати навчання

Метою вивчення навчальної дисципліни є виробити у студентів чітке та систематизоване уявлення про алгоритми та моделі обчислень. Внаслідок вивчення навчальної дисципліни студент повинен бути здатним продемонструвати такі **теоретичні результати** навчання:

- 1) знання теорії автоматів;
- 2) знання теорії формальних мов;
- 3) знання теорії обчислюваності;
- 4) знання теорії складності обчислень;
- 5) знання різних моделей обчислень;
- 6) знання базових алгоритмів обробки інформації.

2.2. Завдання навчальної дисципліни відповідно до освітньої програми

Завдання навчальної дисципліни передбачає отримання у здобувачів освіти здатності застосовувати знання про алгоритми та моделі обчислень для формування наступних (відповідно до освітньо-професійної програми “Комп’ютерна інженерія”) компетентностей:

загальні компетентності:

- ЗК1. Здатність до абстрактного мислення, аналізу і синтезу.
ЗК2. Здатність вчитися і оволодівати сучасними знаннями.
ЗК3. Здатність застосовувати знання у практичних ситуаціях.

ЗК11. Базові знання фундаментальних наук, в обсязі, необхідному для освоєння загально-професійних дисциплін.

ЗК14. Креативність, здатність до системного мислення.

ЗК15. Потенціал до подальшого навчання.

фахові компетентності:

ФК4. Здатність забезпечувати захист інформації, що обробляється в комп'ютерних та кіберфізичних системах та мережах з метою реалізації встановленої політики інформаційної безпеки.

ФК6. Здатність проектувати, впроваджувати та обслуговувати комп'ютерні системи та мережі різного виду та призначення.

ФК7. Здатність використовувати та впроваджувати нові технології, включаючи технології розумних, мобільних, зелених і безпечних обчислень, брати участь в модернізації та реконструкції комп'ютерних систем та мереж, різноманітних вбудованих і розподілених додатків, зокрема з метою підвищення їх ефективності.

ФК12. 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.

2.3. Результати навчання відповідно до освітньої програми, методи навчання і викладання, методи оцінювання досягнення результатів навчання

Результатами навчання є формування у здобувачів освіти таких *практичних вмінь*:

- 1) відображати та читати алгоритми подані різними методами;
- 2) виконувати аналіз алгоритмів;
- 3) синтезувати алгоритми використовуючи різні алгоритмічні стратегії;
- 4) застосовувати базові алгоритми обробки інформації засобами узагальненого програмування(для мов C++, C# та Java);
- 5) застосовувати функційні моделі обчислень та виконувати програмування комп'ютерних систем застосовуючи парадигму функційного програмування;
- 6) застосовувати паралельні моделі обчислень та виконувати програмування комп'ютерних систем застосовуючи парадигму реактивного програмування;
- 7) застосовувати рівночасні моделі обчислень та виконувати програмування комп'ютерних систем застосовуючи парадигму подійно-орієнтованого програмування.

Передбачено такі формальні(відповідно до освітньо-професійної програми "Комп'ютерна інженерія") **програмні результати навчання:**

Результати навчання	Методи навчання і викладання	Методи оцінювання рівня досягнення результатів навчання
ЗН2. Мати навички проведення експериментів, збирання даних та моделювання в комп'ютерних системах.	Лекції, лабораторні та практичні заняття - інформаційно-рецептивний метод, репродуктивний метод, евристичний метод, метод проблемного викладу, самостійна робота – репродуктивний метод, дослідницький метод.	Поточний та екзаменаційний контроль. Методи оцінювання знань: виконання та захист лабораторних робіт, практичних робіт, вибіркове усне опитування. Екзамен – письмове опитування.

УМ1. Вміти застосовувати знання для ідентифікації, формулювання і розв'язування технічних задач спеціальності, використовуючи методи, що є найбільш придатними для досягнення поставлених цілей.	Лекції, лабораторні та практичні заняття - інформаційно-рецептивний метод, репродуктивний метод, евристичний метод, метод проблемного викладу, самостійна робота – репродуктивний метод, дослідницький метод.	Поточний та екзаменаційний контроль. Методи оцінювання знань: виконання та захист лабораторних робіт, практичних робіт, вибіркове усне опитування. Екзамен – письмове опитування.
УМ2. Вміти розв'язувати задачі аналізу та синтезу засобів, характерних для спеціальності.	Лекції, лабораторні та практичні заняття - інформаційно-рецептивний метод, репродуктивний метод, евристичний метод, метод проблемного викладу, самостійна робота – репродуктивний метод, дослідницький метод.	Поточний та екзаменаційний контроль. Методи оцінювання знань: виконання та захист лабораторних робіт, практичних робіт, вибіркове усне опитування. Екзамен – письмове опитування.
УМ3. Вміти системно мислити та застосовувати творчі здібності до формування нових ідей.	Лекції, лабораторні та практичні заняття - інформаційно-рецептивний метод, репродуктивний метод, евристичний метод, метод проблемного викладу, самостійна робота – репродуктивний метод, дослідницький метод.	Поточний та екзаменаційний контроль. Методи оцінювання знань: виконання та захист лабораторних робіт, практичних робіт, вибіркове усне опитування. Екзамен – письмове опитування.
УМ4. Вміти застосовувати знання технічних характеристик, конструктивних особливостей, призначення і правил експлуатації програмно-технічних засобів комп'ютерних систем та мереж для вирішення технічних задач спеціальності.	Лекції, лабораторні та практичні заняття - інформаційно-рецептивний метод, репродуктивний метод, евристичний метод, метод проблемного викладу, самостійна робота – репродуктивний метод, дослідницький метод.	Поточний та екзаменаційний контроль. Методи оцінювання знань: виконання та захист лабораторних робіт, практичних робіт, вибіркове усне опитування. Екзамен – письмове опитування.

2.4. Перелік попередніх та супутніх і наступних навчальних дисциплін

№ з/п	Попередні навчальні дисципліни	Супутні і наступні навчальні дисципліни
1	Програмування, частина I (Основи алгоритмізації та програмування)	Архітектура комп'ютера
2	Програмування, частина II (Об'єктно орієнтоване програмування)	
3	Дискретна математика	

3. Анотація навчальної дисципліни

Курс «Алгоритми та моделі обчислень» розроблений на основі класичних літературних джерел [1, 2, 3, 4, 5, 6] зважаючи на програми аналогічних курсів Массачусетського технологічного інституту [7, 8]. Основу дисципліни становлять наступні розділи.

- Теорія алгоритмів (*в зарубіжній літературі – теорія обчислень, англ. Theory of Computation*) [3]. Тут розглядаються:

- теорія автоматів (*англ. Automata theory*);
- теорія формальних мов (*англ. Formal language theory*);
- теорія обчислюваності (*англ. Computability theory*);
- теорія складності обчислень (*англ. Computational complexity theory*).

- Моделі обчислень (*англ. Models of Computation*) [4, 5].

- Методи розробки алгоритмів(алгоритмічні стратегії)(*англ. Algorithm Design Paradigm, Algorithmic Paradigm, Algorithmic Technique, Algorithmic Strategy*) [6].

Для базових алгоритмів обробки інформації та бібліотек популярних мов програмування, які їх реалізують, розглядаються зразки коду програм, в яких велику увагу приділено використанню:

- узагальненого програмування;
- метапрограмування;
- регулярних виразів та нотації Бекуса-Наура.

Також окрім імперативного(в основному процедурного та об'єктно-орієнтованого) програмування, в наведених зразках коду показано застосування:

- парадигми функційного програмування;
- парадигми реактивного програмування;
- парадигми подійно-орієнтованого програмування.

4. Опис навчальної дисципліни

4.1. Лекційні заняття

№ з/п	Назви тем	Кількість годин
1.	<p>Частина 1. Тема №1. Вступ до теорії алгоритмів.</p> <p>1.1. Неформальне тлумачення алгоритму. Поняття алгоритму. Основні властивості алгоритму. Параметри алгоритму. Базові структури алгоритмів (алгоритмічні конструкції). Теорема Бьоме-Якопіні. Рекурсивні алгоритми. Паралельні алгоритми. Недетерміновані алгоритми. Імовірнісні алгоритми (Probabilistic algorithms).</p> <p>1.2. Формалізація поняття алгоритму(продовження в частині курсу, яка присвячена моделям обчислень(тема №6)). Введення в теорію алгоритмів. Абстрактні моделі алгоритму. Формальні алгоритмічні системи(ФАС). Скінченний автомат. Перетворювачі(трансдуктори) на основі детермінованого скінченного автомату. Автомат з магазинною пам'яттю(МП-автомат, PDA). Машина Тюрінга та її варіанти. Числення Поста. Нормальні алгоритми Маркова. Регістрова машина. РАМ-машина. ПРАМ-машина та варіанти пам'яті із впорядкованим доступом.</p> <p>1.3. Формальні граматики та формальні мови. Формальні граматики, мови та ієрархія Чьомські. Регулярні мови та вирази. КВ-мови(контекстно-вільні мови) та нотації БНФ.</p>	6
2.	<p>Частина 1. Тема №2. Основи аналізу алгоритмів.</p> <p>2.1. Обчислювальна складність. Складність по часу виконання алгоритму. Ємнісна (просторова) складність алгоритму (складність по об'єму пам'яті).</p> <p>2.2. Структурна складність. Цикломатична складність. Структурна складність обчислень поданих структурною матрицею потокового графу алгоритму.</p> <p>2.3. Ієрархії класів складності. Теорема про ієрархії класів часової складності. Теорема про ієрархії класів просторової складності. Поліноміальна ієрархія та РН-клас складності. Експоненціальна ієрархія. Ієрархія Гжегорчика. Арифметична ієрархія. Булева ієрархія.</p>	4
3.	<p>Частина 1. Тема №3. Методи відображення та синтез алгоритмів.</p> <p>3.1. Методи відображення алгоритмів. Вербальне та аналітичне подання алгоритму. Подання алгоритму псевдокодом або з використанням формальних мов. Блок-схема алгоритму. Граф потоку керування. Граф алгоритму. Потоковий граф алгоритму. Діаграма Нассі-Шнайдермана. Діаграми UML. Подання алгоритму структурною матрицею потокового графу алгоритму.</p> <p>3.2. Типи та структури даних. Класифікація типів даних. Базові типи даних. Похідні типи даних. Перетворення типів. Поняття абстрактного типу даних (АТД). Контейнери та колекції. Вектор. Кортеж. Стек. Черга. Черга з пріоритетом. Двобічна черга та двобічна черга з пріоритетом. Список. Одно- та двобічнозв'язні списки. Список з пропусками. Розгорнутий зв'язаний список. Граф. Дерево. Множина. Мультимножина. Асоціативний масив (словник). Мультисловник. Реалізації АТД. Реалізація абстрактних типів даних з оцінюванням складності операцій.</p> <p>3.3. Синтез алгоритмів. Покрокове проектування алгоритмів. Підходи при синтезі алгоритмів(алгоритмічні стратегії). Повний перебір. Метод зменшення розміру задачі та приклади застосування. Метод декомпозиції(«розділяй та володарюй») та приклади застосування. Метод перетворень та приклади застосування. Динамічне програмування та приклади застосування. Метод гілок і границь. Альфа-бета відсікання. Евристичні алгоритми. Метод спроб і помилок (Trial and error). Скупі (жадібні) алгоритми та локальний пошук. Ітераційне вдосконалення алгоритму. Просторово-часовий компроміс (стратегія балансування) при проектуванні алгоритмів та приклади застосування. Оцінювання складності алгоритму під час застосування кожної стратегії.</p>	4

4.	<p>Частина 1. Тема №4. Базові алгоритми обробки інформації.</p> <p>4.1. Алгоритми пошуку. Послідовний пошук та оцінка його складності. Послідовний пошук з бар'єром та оцінка його складності. Бінарний пошук та оцінка його складності. Порозрядний пошук та оцінка його складності. Зовнішній пошук та оцінка його складності. Застосування ґеш-таблиць для пошуку. Розв'язання колізій при ґешуванні відкритою адресацією та методом ланцюжків.</p> <p>4.2. Алгоритми сортування даних. Сортування вибором та оцінка його складності. Сортування вставками та оцінка його складності. Сортування обміном та оцінка його складності. Сортування злиттям та оцінка його складності. Сортування Шелла та оцінка його складності. Швидке сортування та оцінка його складності. Пірамідальне сортування та оцінка його складності. Порозрядне сортування та оцінка його складності. Мережі сортування. Зовнішнє сортування.</p> <p>4.3. Алгоритми порівняння зі зрізцем. Примітивний алгоритм пошуку підрядка та оцінка його складності. Алгоритм Рабіна-Карпа та оцінка його складності. Алгоритм Кнута-Морріса-Пратта та оцінка його складності. Алгоритм Бойєра-Мура та оцінка його складності. Пошук підрядків за допомогою скінчених автоматів. Наближене порівняння рядків.</p> <p>4.4. Чисельні алгоритми. Матриці та дії з ними. Алгоритм Копперсміта-Вінограда та алгоритм Штрассена. Робота з довгими числами. Многочлени та швидке перетворення Фур'є. Системи алгебраїчних рівнянь. Розв'язання систем лінійних рівнянь. Розв'язання нелінійних рівнянь. Алгоритми апроксимації і інтерполяція чисельних функцій.</p> <p>4.5. Алгоритми на графах та мережеві алгоритми. Пошук в ширину і глибину у графі. Побудова кістякового дерева (каркасу графа). Каркас мінімальної ваги. Метод Дж. Крускала. Метод Р. Пріма. Досяжність. Визначення зв'язності. Пошук зв'язних компонентів. Двоzv'язність. Ейлерові цикли. Гамільтонові цикли. Фундаментальна множина циклів. Побудова найкоротших шляхів з виділеної вершини. Алгоритм Дейкстри. Алгоритм Беллмана-Форда. Побудова найкоротших шляхів між двома вершинами. Алгоритм Флойда-Воршелла. Метод генерації всіх максимальних незалежних множин графа. Задача про найменше покриття. Задача про найменше розбиття. Розфарбування графа. Пошук мінімального розфарбування вершин графа. Потoki в мережах. Метод побудови максимального потоку в мережі. Методи наближеного рішення задачі комівояжера(метод локальної оптимізації, алгоритм Ейлера, алгоритм Крістофідеса). Складність алгоритмів на графах.</p> <p>4.6. Паралельні та розподілені алгоритми. Методи паралельного виконання програми за допомогою спільної пам'яті або за допомогою передачі повідомлень. Організація паралельних обчислень відповідно до принципу консенсусу і на основі вибору. Методи визначення завершення паралельних обчислень. Паралельний пошук, паралельне сортування, паралельні чисельні алгоритми, паралельні алгоритми на графах.</p>	12
5.	<p>Частина 1. Тема №5. Бібліотеки основних алгоритмів обробки інформації для популярних мов програмування.</p> <p>5.1. Застосування базових алгоритмів при узагальненому програмуванні на C++ засобами STL(Standard Template Library). Базові типи бібліотеки. Засоби бібліотеки для роботи з стрічками та вводом/виводом. Контейнерні класи бібліотеки. Ітератори. Арифметичні функціональні об'єкти. Предикати. Адаптери-заперечувачі. Адаптери-зв'язувачі. Адаптери вказівників на функції. Адаптери методів. Модифікуючі алгоритми та немодифікуючі алгоритми бібліотеки. Алгоритми пошуку бібліотеки. Алгоритми сортування бібліотеки. Чисельні алгоритми бібліотеки. Розподільники пам'яті для контейнерних класів бібліотеки.</p> <p>5.2. Застосування базових алгоритмів при узагальненому програмуванні на C++ засобами Boost. Застосування boost::any. Застосування boost::assign. Застосування boost::function. Застосування boost::bind. Застосування boost::optional. Застосування boost::variant. Застосування boost::lexical_cast. Застосування boost::spirit. Застосування boost::filesystem. Застосування boost::asio. Застосування boost::static_assert. Метапрограмування за допомогою boost::mpl.</p> <p>5.3. Застосування базових алгоритмів при узагальненому програмуванні на Java засобами JCL(Java Class Library). Засоби для узагальненого програмування на Java. Поняття ітератора в контексті застосування бібліотеки. Інтерфейс Collection. Інтерфейс Set та його реалізації. Інтерфейс Queue та його реалізації. Інтерфейс List та його реалізації. Інтерфейс Map та його реалізації. Алгоритми з допоміжного класу Collections. Використання бібліотеки для конкурентних обчислень.</p> <p>5.4. Застосування базових алгоритмів при узагальненому програмуванні на C# засобами FCL(Framework Class Library). Загальний огляд мови C#. Засоби для узагальненого програмування на C#. Огляд узагальнених та неузагальнених засобів бібліотеки. Поняття ітератора в контексті застосування бібліотеки. Узагальнений та неузагальнений інтерфейси ICollection. Узагальнений та неузагальнений інтерфейси IList та їх реалізації. Узагальнений та неузагальнений класи Stack. Узагальнений та неузагальнений класи Queue. Неузагальнений клас BitArray. Узагальнений та неузагальнений інтерфейси IDictionary та їх реалізації. Використання бібліотеки для конкурентних обчислень.</p> <p>5.5. Застосування алгоритмів лінійної алгебри при програмуванні на C++ за допомогою стандарту BLAS.</p> <p>5.6. Застосування алгоритмів обробки сигналів при програмуванні на C++ та Python засобами OpenCV(Open Source Computer Vision Library).</p>	7

6.	<p>Частина 2. Тема 6. Моделі обчислень.</p> <p>6.1. Елементи теорії моделей. Поняття теорії моделей. Моделі та структури в теорії моделей. Інтерпретації формальних мов. Теорія повноти (<i>Completeness Theorem</i>) (для логіки першого порядку). Принцип перенесення (<i>Transfer Principle</i>) моделей (для логіки першого порядку). Визначуваність (<i>Definability</i>).</p> <p>6.2. Теорія обчислюваності. Основи обчислюваності (<i>Computability</i>). Визначення теорії обчислюваності. Поняття обчислюваної функції (<i>Computable Function</i>). Примітивно-рекурсивні функції (<i>Primitive Recursive Functions</i>). Частково-рекурсивні функції (<i>Partial Recursive Functions</i>). Теза Черча-Тюрінга (<i>Church–Turing Thesis</i>). Розв’язність та розпізнаваність. Задача про прийняття рішень (<i>Decision problem</i>). Розв’язні, напіврозв’язні та нерозв’язні проблеми. Проблема зупинки машини Тюрінга (<i>Halting Problem</i>). Нумерації та універсальність (<i>Gödel Numbering, Enumeration, Universality</i>). Зведення задач (<i>Problem Reductions</i>).</p> <p>6.3. Елементи теорії категорій. Поняття теорії категорій. Приклади категорій. Дуальна категорія (<i>Dual category</i>). Морфізми в категоріях (<i>Morphisms</i>). Початкові та термінальні об’єкти (<i>Initial and terminal objects</i>). Функтори в теорії категорій (<i>Functors</i>). Натуральні перетворення (<i>Natural transformations</i>). Категорії множин (<i>Set category</i>). Категорії типів (<i>Type categories</i>). Картезіансько-замкнені (<i>декартово замкнені</i>) категорії (<i>Cartesian closed categories</i>). Моноїдальні (<i>тензорні</i>) категорії (<i>Monoidal categories</i>).</p> <p>6.4. Імперативний та декларативний підходи до програмування. Класифікація парадигм програмування. Імперативне програмування. Декларативне програмування. Семантична відмінність: опис як «як зробити» проти «що зробити». Сучасні тренди у парадигмах програмування.</p> <p>6.5. Функційні моделі обчислень та парадигма функційного програмування. Лямбда числення. Підстановки та перетворення при застосуванні лямбда числення. Розширення чистого лямбда числення. Теорема про нерухому точку. Редекси і нормальна форма. Теорема Черча-Россера. Редукція термів в лямбда численні. Типізоване лямбда числення. Поняття типу в типізованому лямбда численні. Просте типізоване лямбда числення. Підстановка типу та уніфікація. Комбінаторна логіка(як варіант лямбда числення). Комбінаційна логіка(як функційна модель обчислень). Абстрактна система переписувань(<i>Abstract rewriting system</i>). Парадигма функційного програмування. Мова функційного програмування Haskell. Загальний огляд засобів функційного програмування на C++.</p> <p>6.6. Паралельні моделі обчислень та парадигма реактивного програмування. Паралельні моделі обчислень. Мережа процесів Кана. Мережа Петрі. Мережа взаємодій. Синхронний потік даних.Парадигма реактивного програмування. Функційне реактивне програмування. Реактивне програмування за допомогою бібліотеки ReactiveX. Реалізації ReactiveX для популярних мов програмування. Загальний огляд реактивного програмування за допомогою фреймворку Spring WebFlux з набору фреймворків Spring. Паралельне програмування.</p> <p>6.7. Шаблони проектування програмного забезпечення. Використання шаблонів при проектуванні програмного забезпечення. GoF-шаблони. GRASP-шаблони. Шаблони рівночасних обчислень(<i>Concurrency pattern</i>). Шаблони архітектури програмного забезпечення(<i>Architectural pattern</i>). Використані шаблони в ядрі Linux.</p>	12
Усього годин		45

4.2. Практичні та лабораторні заняття

4.2.1. Практичні заняття

№ з/п	Назви тем	Кількість годин	
		ДФН	ЗФН
1	Практичне заняття №1. Вступне заняття. Огляд матеріалу практичних занять. Поняття алгоритму.	2	
2.	Практичне заняття №2. Алгоритм; властивості, параметри та характеристики складності алгоритму. <u>Задання:</u> Порівняти складність арифметичних операцій в римській та десятковій системах числення. (<i>виконується як домашня робота</i>)	2	
3.	Практичне заняття №3. Вплив правила безпосереднього перероблення на характеристики складності алгоритму. <u>Задання:</u> порівняти часову складність трьох алгоритмів знаходження НСД. (<i>виконується як домашня робота</i>)	2	
4.	Практичне заняття №4. Параметри алгоритму. Правило безпосереднього перероблення. Асимптотичні характеристики	2	

	складності алгоритму. Алгоритми з поліноміальною та експоненціальною складністю. <u>Задання:</u> Визначити часову складність заданого алгоритму в “найгіршому випадку”. (виконується як контрольна робота в кінці пари)		
5.	Практичне заняття №5. Використання потокового графу алгоритму при проектуванні паралельних обчислень. <u>Задання:</u> розпаралелити алгоритм шляхом формування потокового графу алгоритму. (виконується як контрольна робота в кінці пари)	2	
6.	Практичне заняття №6. Формальні алгоритмічні системи (ФАС). Машина Тюрінга (МТ). <u>Задання:</u> Побудувати алгоритм для МТ(сформувати “слід” МТ). Підрахувати часову, програмну та місткісну складність. (виконується як контрольна робота в кінці пари)	2	
7.	Практичне заняття №7. Побудова алгоритмів ефективних за часовою складністю. <u>Задання:</u> застосувати метод «гілок і границь» при побудові алгоритму для вирішення задачі квадратичного призначення. (виконується як контрольна робота в кінці пари)	2	
8.	Захист домашніх завдань.	1	
Усього годин		15	

4.2.2. Лабораторні заняття

№ з/п	Назви тем	Кількість годин	
		ДФН	ЗФН
1	Вступне заняття. Інструктаж. Видача завдань.	4	
2.	Виконання лабораторної №1. Алгоритм; властивості, параметри та характеристики складності алгоритму.	4	
3.	Виконання лабораторної №2. Асимптотичні характеристики складності алгоритму; алгоритми з поліноміальною та експоненціальною складністю.	4	
4.	Виконання лабораторної №3. Використання потокового графу алгоритму при проектуванні паралельних обчислень.	4	
5.	Виконання лабораторної №4. Побудова алгоритмів ефективних за часовою складністю; задача квадратичного призначення.	4	
6.	Виконання лабораторної №5. Функційне програмування.	4	
7.	Виконання лабораторної №6. Реактивне програмування.	4	
8.	Заключне заняття. Кінцевий термін захисту лабораторних робіт.	2	
Усього годин		30	

4.3. Самостійна робота

№ з/п	Найменування робіт	Кількість годин	
		ДФН	ЗФН
1.	Підготовка до лабораторних занять.	10	
2.	Підготовка до навчальних занять та контрольних заходів	20	
3.	Самостійне опрацювання теоретичного матеріалу	15	
Усього годин		45	

5. Опис методів оцінювання рівня досягнення результатів навчання

Оцінювання знань студентів з дисципліни “Алгоритми та моделі обчислень” проводиться відповідно до робочого навчального плану у вигляді **семестрового контролю**, який проводиться в кінці семестру і включає в себе результати **поточного контролю** знань студентів, який оцінюється за виконання лабораторних робіт, та **контрольного заходу** – відповідь на відповідний білет на іспиті. Контрольний захід є обов’язковим видом контролю і проводиться в письмово-усній формі в кінці семестру.

Поточний контроль на лекційних заняттях проводиться з метою виявлення готовності студента до занять у таких формах:

- вибіркове усне опитування перед початком занять;
- оцінка активності студента у процесі занять, внесених пропозицій, оригінальних рішень, уточнень і визначень, доповнень попередніх відповідей і т. ін.

Контрольні запитання поділяються на:

- а) тестові завдання – вибрати вірні відповіді;
- б) проблемні – створення ситуацій проблемного характеру;
- в) питання-репліки – виявити причинно-наслідкові зв’язки;
- г) ситуаційні завдання – визначити відповідь згідно певної ситуації;
- д) питання репродуктивного характеру – визначення практичного значення.

6. Критерії оцінювання результатів навчання здобувачів освіти

Максимальна оцінка в балах				
Поточний контроль (ПК)		Екзаменаційний контроль		Разом за дисципліну
Лаб. роботи	Разом за ПК	письмова компонента	усна компонента	
30	30	60	10	100

Порядок та критерії виставлення балів та оцінок:

1. Розподіл балів при умові виконання навчального плану і календарного плану виконання лабораторних та практичних робіт, інакше за результатами проведення семестрового контролю студент вважається не атестованим.

2. Максимальна кількість балів для оцінки поточного контролю (ПК) знань за семестр – 40 балів.

3. Екзаменаційний контроль проводиться в письмовій формі.

4. Максимальна кількість балів для оцінки екзаменаційного контролю – 60 балів.

5. Іспит перед комісією студент складає в письмово-усній формі з фіксацією запитань та оцінок відповідей на екзаменаційному листі.

6. До іспиту студенти допускаються при умові виконання навчального плану (в тому числі усіх лабораторних та практичних робіт).

7. Навчально-методичне забезпечення

1. Конспект лекцій з дисципліни “Алгоритми та моделі обчислень” для студентів першого (бакалаврського) рівня вищої освіти, спеціальності 123 «Комп’ютерна інженерія» / ЕМНК: <https://vns.lpnu.ua/mod/resource/view.php?id=716346>
2. Методичні вказівки до практичних робіт з дисципліни “Алгоритми та моделі обчислень” для студентів першого (бакалаврського) рівня вищої освіти, спеціальності 123 «Комп’ютерна інженерія» / ЕМНК: <https://vns.lpnu.ua/mod/url/view.php?id=728943>
3. Методичні вказівки до лабораторних робіт з дисципліни “Алгоритми та моделі обчислень” для студентів першого (бакалаврського) рівня вищої освіти, спеціальності 123 «Комп’ютерна інженерія» / ЕМНК: <https://vns.lpnu.ua/mod/url/view.php?id=728944>

8. Рекомендована література

Базова

1. Cormen, Thomas H.; Leiserson, Charles E.; Rivest, Ronald L.; Stein, Clifford. Introduction to Algorithms. — 3rd. — MIT Press, 2009. — ISBN 0-262-03384-4.
2. Donald E. Knuth. The Art of Computer Programming, Volumes 1-4A Boxed Set. Third Edition (Reading, Massachusetts: Addison-Wesley, 2011), 3168pp. ISBN 978-0-321-75104-1, 0-321-75104-3.
3. Michael Sipser (2013). Introduction to the Theory of Computation. 3rd. Cengage Learning. ISBN 978-1-133-18779-0
4. Savage, John E. (1998). Models Of Computation: Exploring the Power of Computing. ISBN 978-0-201-89539-1
5. Fernandez, Maribel (2009). Models of Computation: An Introduction to Computability Theory. Undergraduate Topics in Computer Science. Springer. ISBN 978-1-84882-433-1.
6. Anany Levitin (2012). Introduction to the design & analysis of algorithms. 3rd. ISBN-13: 978-0-13-231681-1
7. <https://ocw.mit.edu/courses/mathematics/18-404j-theory-of-computation-fall-2006/>
8. <https://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-045j-automata-computability-and-complexity-spring-2011/>

Допоміжна

1. Новотарський М.А. Алгоритми та методи обчислень. - Київ : КПІ ім. Ігоря Сікорського, 2019. – 407 с.
2. Кренивич А.П. Алгоритми і структури даних. - К.: ВПЦ "Київський Університет", 2021. – 200 с.
3. Каленюк П. І., В. Л. Бакалець. Вступ до числових методів. - Львів : Н. У., 2000. – 184 с.
4. Цегелик Г. Г. Чисельні методи. - Львів : Н. У., 2004. – 407 с.
5. Анджейчак І. А., Федюк Є. М. Практикум з обчислювальної математики, частина 1. - Навч. посіб. – Львів. ДУ «Львів. Політех.», 2000. – 100 с.
6. Анджейчак І. А. та ін. Практикум з обчислювальної математики. Лекції. - Навч. посіб. частина 2, Львів: «Львів. Політех», 2001. – 152с.
7. Гаврилюк І. П., Головань М. С. Методи обчислень. - Підр. у 2-х час. – К. «В. Ш.». 1995 – 367с.

9. Інформаційні ресурси

1. GeeksforGeeks, <https://www.geeksforgeeks.org/>
2. replit, <https://replit.com/templates>
3. Tutorialspoint, <https://www.tutorialspoint.com/codingground.htm>
4. C++ shell, <https://cpp.sh/>

10. Політика щодо академічної доброчесності

Політика щодо академічної доброчесності учасників освітнього процесу формується на основі дотримання принципів академічної доброчесності з урахуванням норм «Положення про академічну доброчесність у Національному університеті «Львівська політехніка» (затверджене вченою радою університету від 20.06.2017 р., протокол № 35).