

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»

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

Кафедра електронних обчислювальних машин (ЕОМ)
/назва /

«ЗАТВЕРДЖУЮ»

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

Мельник А.О. /
/підпис/ /ініціали та прізвище /

Протокол від « 20 » серпня 2021 р. № 1

РОБОЧА ПРОГРАМА НАВЧАЛЬНОЇ ДИСЦИПЛІНИ

СК1.18. Алгоритми та моделі обчислень

/код і назва навчальної дисципліни/
бакалавр

/рівень вищої освіти/
вид дисципліни _____ за вибором
(обов'язкова / за вибором)
мова викладання _____ українська

освітня програма _____ ОПП “Комп’ютерна інженерія”

/назва/
галузь знань _____ 12 ”Інформаційні технології”

/шифр і назва/
спеціальність _____ 123 “Комп’ютерна інженерія”

/шифр і назва /

Львів – 2021 рік

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

Розробники:

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

Гарант освітньої програми _____ / _____ /
/підпис/ /ініціали та прізвище/

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

Протокол від «20» серпня 2021 року № 1

Завідувач кафедри **ЕОМ** _____ /А.О. Мельник /
/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 (розрахунково-графічні) роботи, к-сть/год.	-	
• індивідуальне науково-дослідне завдання, к-сть/год.	-	
• підготовка до навчальних занять та контрольних 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

Виробити у студентів чітке та систематизоване уявлення про алгоритми та моделі обчислень.

2.2. Завдання навчальної дисципліни відповідно до освітньої програми

Виробити у студентів здатність застосовувати знання про алгоритми та моделі обчислень.

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

В результаті вивчення курсу студенти повинен оволодіти:

зnanнями:

- теорії автоматів,
- теорії формальних мов,
- теорії обчислюваності,
- теорії складності обчислень,
- основних моделей обчислень,
- базових алгоритмів обробки інформації,
- теоретичних основ квантових обчислень;

практичними вміннями:

- відображати та читати алгоритми подані різними методами,
- виконувати аналіз алгоритмів,

- синтезувати алгоритми використовуючи різні алгоритмічні стратегії,
- застосовувати базові алгоритми обробки інформації засобами узагальненого програмування(для мов C++, C# та Java),
- застосовувати функційні моделі обчислень та виконувати програмування комп’ютерних систем застосовуючи парадигму функційного програмування,
- застосовувати паралельні моделі обчислень та виконувати програмування комп’ютерних систем застосовуючи парадигму реактивного програмування,
- застосовувати рівночасні моделі обчислень та виконувати програмування комп’ютерних систем застосовуючи парадигму подійно-орієнтованого програмування.

2.4. Перелік попередніх та супутніх і наступних навчальних дисциплін

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

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].

Додатково до курсу включений розділ присвячений квантовим обчисленням, які являють собою якісно новий етап розвитку комп’ютерних технологій.

Для базових алгоритмів обробки інформації та бібліотек популярних мов програмування, які їх реалізовують, 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	
		ДФН	ЗФН
1.	<p>Частина 1. Тема №1. Вступ до теорії алгоритмів.</p> <p>1.1. Неформальне тлумачення алгоритму.</p> <ul style="list-style-type: none"> 1.1.1. Історія поняття алгоритму. 1.1.2. Визначення алгоритму. 1.1.3. Основні властивості алгоритму. 1.1.4. Параметри алгоритму. 1.1.5. Базові структури алгоритмів(<i>алгоритмічні конструкції</i>). Теорема Бьюма-Якопіні. 1.1.6. Рекурсивні алгоритми. 1.1.7. Паралельні алгоритми. 1.1.8. Недетерміновані алгоритми. 1.1.9. Імовірнісні алгоритми(<i>Probabilistic algorithms</i>). 1.1.10. Рандомізований(увипадковлені) алгоритми(<i>Randomized algorithms</i>). <p>1.2. Формалізація поняття алгоритму. (продовження в частині курсу, яка присвячена моделям обчислень(тема №6))</p> <ul style="list-style-type: none"> 1.2.1. Введення в теорію алгоритмів. 1.2.2. Абстрактні моделі алгоритму. 1.2.3. Формальні алгоритмічні системи(<i>FAC</i>). 1.2.4. Скінчений автомат. <ul style="list-style-type: none"> 1.2.4.1. Детермінований скінчений автомат(<i>DFA</i>). 1.2.4.2. Недетермінований скінчений автомат(<i>NFA</i>) та імовірнісний автомат(<i>PA</i>). 1.2.4.3. ϕ-автомат. 1.2.4.4. Автомат з однобуквеними переходами. Формування автомата з однобуквеними переходами за заданим недетермінованим автомatem. 1.2.4.5. Видалення непродуктивних та недосяжних станів скінченного автомату. 1.3.4.6. Видалення λ-переходів та ϵ-переходів у недетермінованому скінченному автоматі. 1.3.4.7. Детермінізація квазідетермінованого скінченного автомату. 1.3.4.8. Мінімізація скінченого детермінованого автомату. 1.2.5. Перетворювачі(<i>трансдуктори</i>) на основі детермінованого скінченного автомату. <ul style="list-style-type: none"> 1.2.5.1. Автомат Мура(<i>Moore machine</i>). 1.2.5.2. Автомат Міллі(<i>Mealy machine</i>). 1.2.6. Магазинний автомат(<i>PDA</i>). 1.2.7. Машина Тюрінга та її варіанти. 1.2.8. Числення Поста. 1.2.9. Нормальний алгоритм Маркова. 1.2.10. Регістрова машина. 1.2.11. РАМ-машина. 1.2.12. ПРАМ-машина та варіанти пам'яті із впорядкованим доступом. <p>1.3. Формальні граматики та формальні мови.</p> <ul style="list-style-type: none"> 1.3.1. Формальні граматики, мови та ієрархія Чомські. 1.3.2. Регулярні мови та вирази. <ul style="list-style-type: none"> 1.3.2.1. Властивості граматик регулярних мов. Автоматні граматики. Доповнення автоматної мови. 1.3.2.2. Лема про накачку для регулярних мов. 1.3.2.3. Знаходження мови для заданої регулярної граматики. 1.3.2.4. Регулярні вирази. 1.3.2.5. Формування регулярного виразу для заданого недетермінованого скінченного автомату. 1.3.2.6. Формування недетермінованого скінченного автомату для заданої регулярної граматики. 1.3.2.7. Формування недетермінованого скінченного автомату для заданого регулярного виразу. 1.3.3. КВ-мови(<i>контекстно-вільні мови</i>) та нотації БНФ. <ul style="list-style-type: none"> 1.3.3.1. Властивості граматик КВ-мов. 1.3.3.2. Лема про накачку для КВ-мов. 1.3.3.3. Дерева розбору КВ-граматик. 1.3.3.4. Створення МП-автомату для заданої КВ-граматики. 1.3.3.5. Нотації БНФ та РБНФ. 1.3.3.6. Нормальна форма Хомського для КВ-граматик. 1.3.3.7. Алгоритм Кока-Касамі-Янгера для КВ-граматик у нормальній формі Хомського. <p>1.4. Формалізація поняття алгоритму.(дововнення)</p> <ul style="list-style-type: none"> 1.4.1. Детермінізація недетермінованого скінченного автомату. 	6	

<p>2.</p> <p>Частина 1. Тема №2. Методи відображення та синтез алгоритмів.</p> <p>2.1. Методи відображення алгоритмів.</p> <ul style="list-style-type: none"> 2.1.1. Вербальне та аналітичне подання алгоритму. 2.1.2. Подання алгоритму псевдокодом або з використанням формальних мов. 2.1.3. Схематичне подання алгоритму. <ul style="list-style-type: none"> 2.1.3.1. Просте графічне подання алгоритму. <ul style="list-style-type: none"> 2.1.3.1.1. Блок-схема алгоритму. 2.1.3.1.2. Граф потоку керування. 2.1.3.1.3. Граф алгоритму. 2.1.3.1.4. Потоковий граф алгоритму. 2.1.3.2. Структурограма. <ul style="list-style-type: none"> 2.1.3.2.1. Діаграма Нассі-Шнайдермана. 2.1.3.3. Діаграми UML. <ul style="list-style-type: none"> 2.1.3.3.1. Множина структурних та поведінкових діаграм UML. 2.1.3.3.2. Подання задачі поведінковими діаграмами. <ul style="list-style-type: none"> 2.1.3.3.2.1. Діаграма прецедентів(<i>діаграма варіантів використання</i>). 2.1.3.3.3. Подання обчислень поведінковими діаграмами. <ul style="list-style-type: none"> 2.1.3.3.3.1. Діаграма послідовності. 2.1.3.3.3.2. Діаграма комунікації. 2.1.3.3.3.3. Узагальнена діаграма взаємодій. 2.1.3.3.3.4. Діаграма стану. 2.1.3.3.3.5. Діаграма діяльності. 2.1.3.3.4. Структурні діаграми. <ul style="list-style-type: none"> 2.1.3.3.4.1. Діаграма класів. 2.1.3.3.4.2. Діаграма компонентів. 2.1.3.3.4.3. Діаграма розгортання. 2.1.4. Подання алгоритму структурною матрицею потокового графу алгоритму. <p>2.2. Синтез алгоритмів.</p> <ul style="list-style-type: none"> 2.2.1. Покрокове проектування алгоритмів. 2.2.2. Підходи при синтезі алгоритмів(<i>алгоритмічні стратегії</i>). <ul style="list-style-type: none"> 2.2.2.1. Повний перебір. 2.2.2.2. Метод зменшення розміру задачі. 2.2.2.3. Метод декомпозиції. 2.2.2.4. Метод перетворень. 2.2.2.5. Динамічне програмування. 2.2.2.6. Дерево розв'язків та його застосування при проектуванні алгоритмів. <ul style="list-style-type: none"> 2.2.2.6.1. Дерево розв'язків(<i>дерево рішення</i>). 2.2.2.6.2. Бектрекінг(<i>перебір з поверненням</i>). 2.2.2.6.3. Метод гілок і границь. 2.2.2.6.4. Альфа-бета відсікання. 2.2.2.7. Евристичні алгоритми. <ul style="list-style-type: none"> 2.2.2.7.1. Особливості евристичних алгоритмів. 2.2.2.7.2. Метод спроб і помилок(<i>Trial and error</i>). 2.2.2.7.3. Скупі(<i>жадібні</i>) алгоритми та локальний пошук. <ul style="list-style-type: none"> 2.2.2.7.3.1. Особливості скупих алгоритмів. 2.2.2.7.3.2. Локальний пошук. 2.2.2.8. Ітераційне вдосконалення алгоритму. 2.2.2.9. Просторово-часовий компроміс при проектуванні алгоритмів. 	<p>4</p>
--	----------

<p>3.</p> <p>Частина 1. Тема №3. Основи аналізу алгоритмів.</p> <p>3.1. Обчислювальна складність.</p> <ul style="list-style-type: none"> 3.1.1. Складність по часу виконання алгоритму. <ul style="list-style-type: none"> 3.1.1.1. Оцінка розміру вхідних даних. 3.1.1.2. Залежність часу виконання від розміру вхідних даних. 3.1.1.3. Аналіз нерекурсивних алгоритмів. 3.1.1.4. Аналіз рекурсивних алгоритмів. 3.1.1.5. Аналіз алгоритму для найкращого, середнього та найгіршого випадку. 3.1.1.6. Асимптотичний аналіз. <ul style="list-style-type: none"> 3.1.1.6.1. Загальні визначення. 3.1.1.6.2. O-, Θ-, Ω-, ω-, Θ- нотації. 3.1.1.7. Детермінований(DTIME) та недетермінований(NTIME) ресурси часу виконання. 3.1.1.8. Здійснені класи складності. <ul style="list-style-type: none"> 3.1.1.8.1. DLONGTIME-клас складності(<i>логарифмічний</i>). 3.1.1.8.2. PolylogTIME-клас складності(<i>полілогарифмічний</i>). 3.1.1.8.3. P-клас складності(<i>поліноміальний</i>). 3.1.1.8.4. P-повні задачі. 3.1.1.8.5. RP-клас та coRP-клас складності. 3.1.1.8.6. ZPP-клас складності. 3.1.1.8.7. BPP-клас складності. 3.1.1.8.8. BQP-клас складності. 3.1.1.9. Проблематичні класи складності. <ul style="list-style-type: none"> 3.1.1.9.1. NP- та coNP- класи складності. 3.1.1.9.1. Співвідношення P- та NP- класи складності. 3.1.1.9.2. NP- та coNP- повні задачі. Теорема Кука-Левіна. Стандартні NP-повні задачі. 3.1.1.9.3. NP- складні, еквівалентні, проміжні та прості задачі 3.1.1.9.4. PP-клас складності. 3.1.1.9.5. UP-клас складності. 3.1.1.9.6. #P-клас(<i>вимовляється як «шарп П»</i>) складності та #P-повні задачі. 3.1.1.9.7. \oplusP-клас(<i>вимовляється як «паритет П»</i>) складності. 3.1.1.10. Нездійсненні класи складності. <ul style="list-style-type: none"> 3.1.1.10.1. EXPTIME-клас складності(<i>експоненціальний клас складності</i>). 3.1.1.10.2. NEXPTIME-клас складності. 3.1.1.10.3. 2-EXPTIME-клас складності. 3.1.1.10.4. ELEMENTARY-клас складності. 3.1.1.10.5. R-клас складності. 3.1.1.10.6. PR-клас складності. 3.1.1.10.7. RE-клас складності та coRE-клас складності. 3.1.1.10.8. ALL-клас складності. 3.1.1.11. Аналіз паралельних алгоритмів. <ul style="list-style-type: none"> 3.1.1.11.1. Особливості аналізу паралельних алгоритмів. 3.1.1.11.2. Теорема Брента. Закон Густавсона–Барсіса. Закон Амдала. 3.1.1.11.3. NC-клас складності. 3.1.2. Емнісна(просторова) складність алгоритму(<i>складність по об'єму пам'яті</i>). <ul style="list-style-type: none"> 3.1.2.1. Визначення просторової складності алгоритму. 3.1.2.2. Детермінований(DSPACE) та недетермінований(NSPACE) просторові(емнісні) ресурси. 3.1.2.3. Теорема Севіча. 3.1.2.4. Здійснені класи складності. <ul style="list-style-type: none"> 3.1.2.4.1. L-клас складності. 3.1.2.4.2. PolyL-клас складності(<i>полілогарифмічний</i>). 3.1.2.4.3. SL-клас складності. 3.1.2.4.4. NL-клас складності. 3.1.2.4.5. NL-повні задачі. 3.1.2.4.6. Еквівалентність класів NL та coNL. 3.1.2.4.7. RL-клас складності. 3.1.2.5. Проблематичні класи складності. <ul style="list-style-type: none"> 3.1.2.5.1. PSPACE-клас складності. 3.1.2.5.2. PSPACE-повні задачі. 3.1.2.6. Нездійсненні класи складності. <ul style="list-style-type: none"> 3.1.2.6.1. EXPSPACE-клас складності(<i>експоненціальний клас складності по пам'яті</i>). 3.2. Структурна складність. 3.2.1. Цикломатична складність. <ul style="list-style-type: none"> 3.2.1.1. Цикломатичне число. 3.2.1.2. Цикломатична складність графу потоку керування та графу алгоритму. 3.2.2. Структурна складність обчислень поданих структурною матрицею потокового графу алгоритму. 3.2.3. ACⁱ-класи складності. 3.2.4. ACC⁰-клас складності. 3.2.5. TCⁱ-класи складності. 3.2.6. CC-клас складності. <p>3.3. Ієрархії класів складності.</p> <ul style="list-style-type: none"> 3.3.1. Теорема про ієрархії класів часової складності. 3.3.2. Теорема про ієрархії класів просторової складності. 3.3.3. Поліноміальна ієрархія та PH-клас складності. 3.3.4. Експоненціальна ієрархія. 3.3.5. Ієрархія Джегорчика. 3.3.6. Арифметична ієрархія. 3.3.7. Булева ієрархія. 	<p>2</p>
---	----------

4.	Частина 1. Тема №4. Базові алгоритми обробки інформації. 4.1. Алгоритми пошуку. 4.1.1. Послідовний пошук. 4.1.2. Бінарний пошук. 4.1.3. Дерева бінарного пошуку. 4.1.4. Збалансовані дерева. 4.1.5. Порозрядний пошук. 4.1.6. Зовнішній пошук. 4.1.7. Хешування. 4.1.8. Розв'язання колізій при хешуванні відкритою адресацією та методом ланцюжків. 4.2. Алгоритми сортування даних. 4.2.1. Сортування вибором. 4.2.2. Сортування вставками. 4.2.3. Сортування обміном. 4.2.4. Сортування злиттям. 4.2.5. Сортування Шелла. 4.2.6. Швидке сортування. 4.2.7. Піраміdalne сортування. 4.2.8. Порозрядне сортування. 4.2.9. Мережі сортування. 4.2.10. Зовнішнє сортування. 4.3. Алгоритми порівняння зі взірцем. 4.3.1. Примітивний алгоритм пошуку підрядка. 4.3.2. Алгоритм Рабіна-Карпа. 4.3.3. Алгоритм Кнута-Морріса-Пратта. 4.3.4. Алгоритм Бойєра-Мура. 4.3.5. Пошук підрядків за допомогою скінчених автоматів. 4.3.6. Наближене порівняння рядків. 4.4. Чисельні алгоритми. 4.4.1. Матриці та дії з ними. Алгоритм Копперсміта-Вінограда та алгоритм Штрассена. 4.4.2. Робота з довгими числами. 4.4.3. Многочлені та швидке перетворення Фур'є. 4.4.4. Алгебраїчні системи. 4.4.5. Розв'язання систем лінійних рівнянь. 4.4.6. Розв'язання нелінійних рівнянь. 4.4.7. Алгоритми апроксимації і інтерполяція чисельних функцій. 4.5. Графи та мережеві алгоритми. 4.5.1. Пошук у графі. 4.5.2. Породження всіх каркасів графа. 4.5.3. Каркас мінімальної ваги. Метод Дж. Крускала. Метод Р. Пріма. 4.5.4. Досяжність. Визначення зв'язності. Двозв'язність. 4.5.5. Ейлерові цикли. 4.5.6. Гамільтонові цикли. 4.5.7. Фундаментальна множина циклів. 4.5.8. Алгоритм Дейкстри. 4.5.9. Алгоритм Флойда. 4.5.10. Метод генерації всіх максимальних незалежних множин графа. Задача про найменше покриття. 4.5.11. Задача про найменше розбиття. 4.5.12. Розфарбування графа. 4.5.13. Пошук мінімального розфарбування вершин графа. 4.5.14. Використання задачі про найменше покриття при розфарбуванні вершин графа. 4.5.15. Потоки в мережах. 4.5.16. Метод побудови максимального потоку в мережі. 4.5.17. Методи наближеного рішення задачі комівояжера(метод локальної оптимізації, алгоритм Эйлера, алгоритм Кристофідеса). 4.5.18. Аналіз алгоритмів на графах. 4.6. Паралельні та розподілені алгоритми. 4.6.1. Методи паралельного виконання програми за допомогою спільної пам'яті або за допомогою передачі повідомлень. 4.6.2. Організація паралельних обчислень відповідно до принципу консенсусу і на основі вибору. 4.6.3. Методи визначення завершення паралельних обчислень. 4.6.4. Паралельний пошук, паралельне сортування, паралельні чисельні алгоритми, паралельні алгоритми на графах.	12
----	---	----

5.	<p>Частина 1. Тема №5. Бібліотеки основних алгоритмів обробки інформації для популярних мов програмування.</p> <p><i>5.1. Застосування базових алгоритмів при узагальненному програмуванні на C++ засобами STL(Standard Template Library).</i></p> <ul style="list-style-type: none"> 5.1.1. Огляд бібліотеки. 5.1.2. Огляд базових типів бібліотеки. 5.1.3. Засоби бібліотеки для роботи з стрічками та вводом/виводом. 5.1.4. Контейнерні класи бібліотеки. <ul style="list-style-type: none"> 5.1.4.1. Послідовні контейнери бібліотеки. 5.1.4.2. Асоціативні контейнери бібліотеки. 5.1.5. Ітератори. 5.1.6. Функціональні об'єкти. <ul style="list-style-type: none"> 5.1.6.1. Арифметичні функціональні об'єкти. 5.1.6.2. Предикати. 5.1.6.3. Адаптери. <ul style="list-style-type: none"> 5.1.6.3.1. Заперечувачі. 5.1.6.3.2. Зн'язувачі. 5.1.6.3.3. Адаптери вказівників на функції. 5.1.6.3.4. Адаптери методів. 5.1.7. Алгоритми бібліотеки. <ul style="list-style-type: none"> 5.1.7.1. Алгоритми сортування та пошуку. 5.1.7.2. Узагальнені чисельні алгоритми. 5.1.7.3. Інші немодифікуючі алгоритми. 5.1.7.4. Інші модифікуючі алгоритми. 5.1.8. Розподільники нам'яті для контейнерів бібліотеки. <p><i>5.2. Застосування базових алгоритмів при узагальненному програмуванні на C++ засобами Boost.</i></p> <ul style="list-style-type: none"> 5.2.1. Огляд набору бібліотек Boost. 5.2.2. Контейнери та алгоритми. 5.2.3. Стрічкові алгоритми. 5.2.4. Окремі засоби набору бібліотек Boost. <ul style="list-style-type: none"> 5.2.4.1. Застосування boost:any. 5.2.4.2. Застосування boost:assign. 5.2.4.3. Застосування boost:function. 5.2.4.4. Застосування boost:bind. 5.2.4.5. Застосування boost:optional. 5.2.4.6. Застосування boost:variant. 5.2.4.7. Застосування boost:lexical_cast. 5.2.4.8. Застосування boost:spirit. 5.2.4.9. Застосування boost:filesystem. 5.2.4.10. Застосування boost:asio. 5.2.4.11. Застосування boost:static_assert. 5.2.5. Метапрограмування за допомогою boost:mpl. <p><i>5.3. Застосування базових алгоритмів при узагальненному програмуванні на Java засобами JCL(Java Class Library).</i></p> <ul style="list-style-type: none"> 5.3.1. Засоби для узагальненного програмування в Java. 5.3.2. Огляд бібліотеки. 5.3.3. Поняття ітератора в контексті застосування бібліотеки. <ul style="list-style-type: none"> 5.3.3.1. Застарілий інтерфейс Enumeration. 5.3.3.2. Інтерфейси Iterator. 5.3.3.3. Інтерфейси Iterable. 5.3.4. Інтерфейс Collection. 5.3.5. Інтерфейс Set та його реалізації. <ul style="list-style-type: none"> 5.3.5.1. Інтерфейси Set, SortedSet та NavigableSet. 5.3.5.2. Класи HashSet, LinkedHashSet та TreeSet. 5.3.6. Інтерфейс Queue та його реалізації. <ul style="list-style-type: none"> 5.3.6.1. Інтерфейси Queue та Deque. 5.3.6.2. Класи LinkedList, ArrayDeque та PriorityQueue. 5.3.7. Інтерфейс List та його реалізації. <ul style="list-style-type: none"> 5.3.7.1. Інтерфейс List. 5.3.7.2. Класи Vector, Stack, ArrayList та LinkedList. 5.3.7.3. Інтерфейс ListIterator. 5.3.8. Інтерфейс Map та засоби реалізації. <ul style="list-style-type: none"> 5.3.8.1. Інтерфейси Map, SortedMap та NavigableMap. 5.3.8.2. Класи Hashtable, TreeMap, HashMap, LinkedHashMap, ArrayList та WeakHashMap. 5.3.9. Алгоритми з допоміжного класу Collections. 5.3.10. Використання бібліотеки для конкурентних обчислень. <ul style="list-style-type: none"> 5.3.10.1. Застосування ключового слова synchronized. 5.3.10.2. Огляд засобів пакету java.util.concurrent. <p><i>5.4. Застосування базових алгоритмів при узагальненному програмуванні на C# засобами FCL(Framework Class Library).</i></p> <ul style="list-style-type: none"> 5.4.1. Засоби узагальненного програмування в C#. 5.4.2. Огляд узагальнених та неузагальнених засобів бібліотеки. 5.4.3. Поняття ітератора в контексті застосування бібліотеки. <ul style="list-style-type: none"> 5.4.3.1. Узагальнені та неузагальнені інтерфейси IEnumerator. 5.4.3.2. Узагальнені та неузагальнені інтерфейси IEnumerable. 5.4.4. Узагальнені та неузагальнені інтерфейси ICollection. 5.4.5. Узагальнені та неузагальнені інтерфейси IList та їх реалізації. <ul style="list-style-type: none"> 5.4.5.1. Узагальнені та неузагальнені інтерфейси IList. 5.4.5.2. Узагальнені класи HashSet, List та Collection. 5.4.5.3. Неузагальнені класи ArrayList та Array. 5.4.6. Узагальнені та неузагальнені класи Stack. 5.4.7. Узагальнені та неузагальнені класи Queue. 5.4.8. Неузагальненій клас BitArray. 5.4.9. Узагальнені та неузагальнені інтерфейси IDictionary та їх реалізації. <ul style="list-style-type: none"> 5.4.9.1. Узагальнені та неузагальнені інтерфейси IDictionary. 5.4.9.2. Узагальнені класи Dictionary, SortedDictionary та SortedList. 5.4.9.3. Неузагальнені класи ListDictionary, HashTable та SortedList. 5.4.10. Використання бібліотеки для конкурентних обчислень. <ul style="list-style-type: none"> 5.4.10.1. Властивості ICollection, IsSynchronized та ICollection.SyncRoot. 5.4.10.2. Огляд засобів простору імен System.Collections.Concurrent. <p><i>5.5. Застосування алгоритмів лінійної алгебри при програмуванні на C++ за допомогою стандарту BLAS.</i></p> <ul style="list-style-type: none"> 5.5.1. Огляд стандарту. 5.5.2. Три рівні функціональності стандарту. 5.5.3. Огляд бібліотек-реалізації стандарту. 5.5.4. Бібліотека-реалізація uBLAS з набору бібліотек Boost. 5.5.5. Приклад коду мовою C++. <p><i>5.6. Застосування алгоритмів обробки сигналів при програмуванні на C++ та Python засобами OpenCV(Open Source Computer Vision Library).</i></p> <ul style="list-style-type: none"> 5.6.1. Огляд бібліотеки. Доступність бібліотеки різним мовам програмування. 5.6.2. Основні модулі бібліотеки. 5.6.3. Типове застосування бібліотеки. 5.6.4. Приклад коду мовою C++. 5.6.5. Приклад коду мовою Python. 	6
----	---	---

6.	Частина 2. Тема 6. Моделі обчислень. 6.1. Елементи теорії моделей. <ul style="list-style-type: none"> 6.1.1. Визначення теорії моделей. 6.1.2. Моделі в теорії моделей. 6.1.3. Інтерпретації формальних мов. 6.1.4. Теорія повноти моделей(<i>для логіки першого порядку</i>). 6.1.5. Принцип перенесення моделей(<i>для логіки першого порядку</i>). 6.1. Елементи теорії обчислюваності. <ul style="list-style-type: none"> 6.2.1. Визначення теорії обчислюваності. 6.2.2. Поняття обчислюваної функції. 6.2.3. Примітивно-рекурсивні функції. 6.2.4. Теза Черча-Тюрінга. 6.2.5. Задача про прийняття рішень. 6.2.6. Необчислюваність. <ul style="list-style-type: none"> 6.2.6.1. Теореми Геделя про неповноту. 6.2.6.2. Поняття необчислюваної функції(<i>Uncomputable function</i>). 6.2.6.3. Задача про зупинку машини Тюрінга. 6.2.6.4. Нерозв'язувана задача про прийняття рішень та степінь Тюрінга. 6.2.7. Обчислення з оракулом. 6.2.8. Зведення однієї задачі до іншої за поліноміальний час. <ul style="list-style-type: none"> 6.2.8.1. Зведення Карпа(<i>Karp reductions, Many-one reductions</i>). 6.2.8.2. Зведення Кука(<i>Cook reduction</i>). 6.2.8.3. Зведення Левіна(<i>Levin reduction</i>). 6.2.8.4. Зведення через таблицю істинності(<i>Truth-table reduction</i>). 6.2.8.5. Зведення Тюрінга(<i>Turing reduction</i>). 6.3. Функційні моделі обчислень та парадигма функційного програмування. <ul style="list-style-type: none"> 6.3.1. Функційні моделі обчислень. <ul style="list-style-type: none"> 6.3.1.1. Лямбда числення. 6.3.1.2. Типізоване лямбда числення. 6.3.1.3. Комбінаційна логіка(як функційна модель обчислень). 6.3.1.4. Абстрактна перезаписуюча система. 6.3.2. Парадигма функційного програмування. 6.4. Паралельні моделі обчислень та парадигма реактивного програмування. Паралельне програмування. <ul style="list-style-type: none"> 6.4.1. Паралельні моделі обчислень. <ul style="list-style-type: none"> 6.4.1.1. Мережа процесів Кана. 6.4.1.2. Мережа Петрі. 6.4.1.3. Мережа взаємодій. 6.4.1.4. Синхронний потік даних. 6.4.2. Парадигма реактивного програмування. 6.4.3. Паралельне програмування. 6.5. Шаблони проектування програмного забезпечення. <ul style="list-style-type: none"> 6.5.1. Використання шаблонів при проектуванні програмного забезпечення. 6.5.2. GoF-шаблони. <ul style="list-style-type: none"> 6.5.2.1. Твірні шаблони(<i>Creational pattern</i>). 6.5.2.2. Структурні шаблони(<i>Structural pattern</i>). 6.5.2.3. Поведінкові шаблони(<i>Behavioral pattern</i>). 6.5.3. GRASP-шаблони. 6.5.4. Шаблони рівночасних обчислень(<i>Concurrency pattern</i>). 6.5.5. Шаблони архітектури програмного забезпечення(<i>Architectural pattern</i>). 6.5.6. Шаблони ядра Linux. 	10
----	---	----

7.	<p>Частина 3. Тема №7. Квантові обчислення.</p> <p>7.1. Базові поняття.</p> <ul style="list-style-type: none"> 7.1.1. Деякі суміжні поняття з фізики елементарних частин. 7.1.2. Суперпозиція квантових станів. 7.1.3. Кубіти. <ul style="list-style-type: none"> 7.1.3.1. Поняття кубіту. 7.1.3.2. Представлення кубіта на сфері Блоха. 7.1.3.3. Вимірювання значень кубітів. 7.1.3.4. Керування станом кубіта: осциляція Рабі. 7.1.4. Неможливість клонування квантових станів. <p>7.2. Квантові вентилі та квантові реєстри.</p> <ul style="list-style-type: none"> 7.2.1. Система кубітів та квантова сплутаність. <ul style="list-style-type: none"> 7.2.1.1. Способи генерації сплутаних станів. 7.2.1.2. Методи детектування сплутаних станів. 7.2.2. Вентиль тотожного перетворення. 7.2.3. Вентиль заперечення. 7.2.4. Вентиль фазового зміщення. 7.2.5. Вентиль перетворення Адамара. 7.2.6. Прямий керуючий вентиль. 7.2.7. Вентиль контролюваного заперечення. 7.2.8. Вентиль Тоффолі. 7.2.9. Вентиль Фредкіна. <p>7.3. Квантової обчислени.</p> <ul style="list-style-type: none"> 7.3.1. Квантова комбінаційна схема. 7.3.2. Квантовий скінченний автомат. 7.3.3. Квантова машина Тюрінга. 7.3.4. Деякі спеціальні форми квантових обчислень. <ul style="list-style-type: none"> 7.3.4.1. Адіабатичні квантові обчислени. 7.3.4.2. Однобічний квантовий комп'ютер. <p>7.4. Важливі алгоритми квантових обчислень.</p> <ul style="list-style-type: none"> 7.4.1. Задача та алгоритм Дойча-Джозі. 7.4.2. Задача та алгоритм Бернштайн-Вазірані. 7.4.3. Задача Саймона. 7.4.4. Алгоритм Шора. 7.4.5. Алгоритм Гровера. 7.4.6. Квантовий відпал. <p>7.5. Засоби квантових обчислень.</p> <ul style="list-style-type: none"> 7.5.1. Програмний каркас Qiskit. <ul style="list-style-type: none"> 7.5.1.1. Загальний огляд Qiskit. 7.5.1.2. Мова проміжного рівня OpenQASM для представлення квантових інструкцій. 7.5.1.3. Платформа IBM Quantum Experience. 7.5.2. Огляд інших засобів квантових обчислень. 	5
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. (виконується як домашня робота)	2	
3.	Практичне заняття №3. Вплив правила безпосереднього перероблення на характеристики складності алгоритму. <u>Задання:</u> порівняти часову складність трьох алгоритмів знаходження НСД. (виконується як домашня робота)	2	

4.	Практичне заняття №4. Параметри алгоритму. Правило безпосереднього перероблення. Асимптотичні характеристики 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)	2	
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 заняття;
- 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, виконання усіх контрольних робіт і календарного плану виконання лабораторних робіт, інакше за результатами проведення семестрового контролю студент вважається не атестованим.
2. Максимальна кількість балів для оцінки поточного контролю (ПК) знань за семestr – 40 балів.
3. Екзаменаційний контроль проводиться в письмово-усній формі.
4. Максимальна кількість балів для оцінки екзаменаційного контролю – 70 балів.
5. Іспит перед комісією студент складає також 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 робіт).

7. Рекомендована література

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/>

8. Політика щодо академічної добросусідності

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

9. УНІФІКОВАНИЙ ДОДАТОК

Національний університет «Львівська політехніка» забезпечує реалізацію права осіб з особливими освітніми потребами на здобуття вищої освіти. Інклюзивні освітні послуги надає Служба доступності до можливостей навчання «Без обмежень», метою діяльності якої є забезпечення постійного індивідуального супроводу навчального процесу здобувачів освіти з інвалідністю та хронічними захворюваннями. Важливим інструментом імплементації інклюзивної освітньої політики в Університеті є Програма підвищення кваліфікації науково-педагогічних працівників та навчально-допоміжного персоналу у сфері соціальної інклюзії та інклюзивної освіти. Звертатися за адресою:

вул. Карпінського, 2/4, I-й н.к., кімн. 112

E-mail: nolimits@lpnu.ua

Websites: <https://lpnu.ua/nolimits>, <https://lpnu.ua/integration>

10. Зміни та доповнення до робочої програми навчальної дисципліни

№ з/п	Зміст внесених змін (доповнень)	Дата і № протоколу засідання кафедри	Примітки