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

Кафедра штучного інтелекту

Проектування та аналіз обчислювальних алгоритмів

Робоча програма навчальної дисципліни (Силабус)

Реквізити навчальної дисципліни

Рівень вищої освіти	***Перший (бакалаврський)***
Галузь знань	*12 Інформаційні технології*
Спеціальність	*122 Комп'ютерні науки*
Освітня програма	*Системи і методи штучного інтелекту*
Статус дисципліни (код)	*Нормативна*
Форма навчання	*очна(денна)/дистанційна/змішана*
Рік підготовки, семестр	*2 курс, осінній семестр*
Обсяг дисципліни	*6 кредитів ЄКТС*
Семестровий контроль/ контрольні заходи	*Залік/МКР*
Розклад занять	*Rozklad.kpi.ua*
Мова викладання	*Українська*
Інформація про керівника курсу / викладачів	Лектор: *к.т.н., доцент, Тимошенко Юрій Олександрович, https://intellect.kpi.ua/profile/tyo6* Практичні: *к.т.н., доцент, Тимошенко Юрій Олександрович*
Розміщення курсу	Googleclassroom https://classroom.google.com/c/NjE5MzE2MTUxOTQ0

Програма навчальної дисципліни

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

Розробка та аналіз обчислювальних алгоритмів є важливою галуззю інформатики, яка зосереджена на створенні ефективних алгоритмів для вирішення різноманітних обчислювальних проблем. Ця дисципліна включає як теоретичні, так і практичні аспекти, спрямовані на те, щоб озброїти студентів навичками розробки алгоритмів, аналізу їх ефективності та розуміння їх поведінки в різних сценаріях. Він забезпечує основу для ефективного вирішення складних проблем і формує основу різноманітних програм, починаючи від аналізу даних і оптимізації до штучного інтелекту та комп'ютерної графіки. Мета цієї навчальної дисципліни — надати студентам міцну основу алгоритмічного мислення, що дозволить їм розробляти інноваційні рішення для обчислювальних завдань. Оскільки технологія розвивається, ця дисципліна залишається вирішальною для того, щоб програмне забезпечення та системи могли справлятися зі зростаючими масштабами та складністю сучасних обчислювальних проблем.

Ця дисципліна включає:

- Проектування алгоритмів;
- Аналіз алгоритмів;
- Структури даних;
- Сортування та пошук;
- Динамічне програмування;
- Жадібні алгоритми;
- Алгоритми на графах;
- Теорія складності;
- Рандомізовані алгоритми;
- Паралельні та розподілені алгоритми;

У процесі навчання студент має оволодіти такими компетентностями:

ЗК 1 Здатність до абстрактного мислення, аналізу та синтезу; ЗК 2 Здатність застосовувати знання у практичних ситуаціях; ЗК 3 Знання та розуміння предметної області та розуміння професійної діяльності; ЗК 6 Здатність вчитися і оволодівати сучасними знаннями; ФК 17 Здатність забезпечувати моделювання технічних та інформаційних об'єктів і систем штучного інтелекту, проводити експерименти за заданими методиками з обробкою й аналізом результатів; ФК 22 Здатність використовувати мови штучного інтелекту при розробці програмного забезпечення інтелектуальних інформаційних систем, здатність орієнтуватися в різних типах інтелектуальних систем і технологій; ставити завдання побудови інтелектуальних систем для вирішення завдання вибору варіантів в проблемної області, що погано формалізується; ФК 24 Здатність орієнтуватися в сучасних напрямках розвитку ШІ та нових засобах побудови систем штучного інтелекту та знаходити та розробляти новітні ефективні алгоритми.

По завершенню курсу студент має набути наступні результати навчання:

1. **Знання**:

- Студенти повинні вміти аналізувати проблеми реального світу, визначати їхні обчислювальні аспекти та розробляти алгоритмічні рішення для ефективного вирішення цих проблем.
- Студенти повинні розуміти, як аналізувати часову та просторову складність алгоритмів, що дозволяє їм оцінювати ефективність і масштабованість своїх рішень.
- Студенти повинні бути знайомі та вміти застосовувати різні алгоритмічні парадигми, такі як динамічне програмування, жадібні алгоритми та «розділяй і володарюй» для вирішення складних проблем.
- Студенти повинні добре розуміти відповідні математичні поняття, такі як індукція, рекурентні співвідношення, ймовірність і комбінаторика, щоб аналізувати та розробляти алгоритми.
- Студенти повинні мати базові знання про розробку алгоритмів для паралельних і розподілених обчислювальних середовищ, а також знати про проблеми та переваги таких систем.
- Студенти повинні бути здатні критично оцінювати алгоритмічні рішення, враховуючи такі фактори, як ефективність, правильність і відповідність конкретним контекстам.
- Студенти повинні знати про етичні наслідки розробки алгоритмів, особливо у випадках, коли алгоритми впливають на прийняття рішень, конфіденційність і суспільний добробут.
- Студенти повинні мати можливість досліджувати та пропонувати нові алгоритмічні рішення, сприяючи розвитку галузі шляхом досліджень та інновацій.

2. **Уміння**:

- Студенти повинні вміти розробляти ефективні алгоритми для різноманітних типів проблем, беручи до уваги такі фактори, як правильність, ефективність та придатність до обслуговування.
- Студенти повинні вміти вибирати відповідні структури даних для різних сценаріїв проблем, а також розуміти їхні характеристики та компроміси.
- Алгоритми графів: Студенти повинні вміти працювати з графами, розуміти такі поняття, як вершини, ребра, обхід, найкоротші шляхи, зв'язність і застосовувати відповідні алгоритми графів.
- Студенти повинні розуміти та вміти застосовувати різні алгоритми сортування та пошуку, враховуючи їхню ефективність та компроміси.
- Студенти повинні вміти оптимізувати алгоритми для покращення продуктивності, враховуючи такі фактори, як мінімізація складності часу, зменшення зайвих обчислень та оптимізація використання пам'яті.

- Студенти повинні вміти перетворювати алгоритми у функціональний код і демонструвати вміння кодувати, налагоджувати, тестувати та оптимізувати свої реалізації.
- Студенти повинні мати можливість ефективно документувати та повідомляти про свої проекти алгоритмів, аналізи та впровадження технічній і нетехнічній аудиторії.

3. **Комунікації**:

- Комунікативні навички: студенти розвиватимуть свою здатність доносити складні концепції обчислювальних алгоритмів до нетехнічної аудиторії та ефективно співпрацювати в групових проектах;
- бути зрозумілим;
- толерантно ставитися до осіб, що мають інші культурні чи гендерні-вікові особливості;

4. **Автономність та відповідальність**:

- Студенти повинні усвідомлювати необхідність безперервного навчання в галузі, що швидко розвивається, бути в курсі нових алгоритмічних методів, інструментів і досягнень.
- Обґрунтовувати власну позицію, робити самостійні висновки;
- Відповідально ставитись до професійного самовдосконалення, навчання та саморозвитку.

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

Загалом, студенти закінчать цей курс «Проектування та аналіз обчислювальних алгоритмів» відображають комплексний набір навичок, які студенти повинні отримати під час вивчення дизайну та аналізу обчислювальних алгоритмів, озброюючи їх для вирішення складних обчислювальних завдань у різних областях.

3. Зміст навчальної дисципліни

Розділ 1: Вступ до алгоритмічного мислення та аналізу

Тема 1.1 : Введення в алгоритми та їх значення. Аналіз алгоритмів з точки зору часової та просторової складності.

Тема 1.2: Асимптотична нотація (Big O, Big Omega, Big Theta). Алгоритми грубої сили та їх обмеження.

Тема 1.3 : Парадигми проектування алгоритмів: розділяй і володарюй, динамічне програмування, жадібні алгоритми.

Розділ розкриває наступні питання: Що таке алгоритми та їх значення; Алгоритм подання псевдокоду; Основи аналізу алгоритмів: часова та просторова складність; Асимптотична нотація: велика О, Омега та Тета; Приклади аналізу простих алгоритмів

Розділ 2: Методи проектування алгоритмів

Тема 2.1: Розділяй і володарюй: принципи та приклади (наприклад, сортування злиттям).

Тема 2.2: Жадібні алгоритми: принципи та приклади (наприклад, кодування Хаффмана).

Тема 2.3: Динамічне програмування: принципи та приклади (наприклад, послідовність Фібоначчі).

Тема 2.4: Відкат: принципи та приклади (наприклад, проблема N-Королев).

Розділ 3: Структури даних для ефективних алгоритмів

Тема 3.1: Масиви, зв'язані списки, стеки та черги.

Тема 3.2: Дерева: бінарні дерева, бінарні дерева пошуку, збалансовані дерева (наприклад, дерева AVL).

Тема 3.3: Купи та пріоритетні черги

Тема 3.4: Хешування та хеш-таблиці. Графіки: термінологія, представлення та основні алгоритми (наприклад, BFS, DFS)

Розділ 4: Алгоритми сортування та пошуку

Тема 4.1: Бульбашкове сортування, сортування виділенням, сортування вставкою.

Тема 4.2: Швидке сортування, сортування злиттям, сортування купою.

Тема 4.3: Радиксне сортування, ковшове сортування.

Тема 4.4: Двійковий пошук та його різновиди.

Розділ 5: Розширені алгоритмічні концепції

Тема 5.1: Амортизований аналіз: приклади (наприклад, динамічні масиви).

Тема 5.2: Рандомізовані алгоритми: принципи та приклади (наприклад, рандомізоване швидке сортування).

Тема 5.3: Онлайн-алгоритми: конкурентний аналіз і приклади (наприклад, онлайн-планування)

Розділ 6: Алгоритми графів

Тема 6.1: Обхід графа: BFS, DFS.

Тема 6.2: Алгоритми найкоротшого шляху: алгоритм Дейкстри, алгоритм Беллмана-Форда.

Тема 6.3: Мінімальні остовні дерева: алгоритм Прима, алгоритм Крускала.

Тема 6.4: Компоненти сильного зв'язку та топологічне сортування.

Розділ 7: Обчислювальна складність

Тема 7.1: Ознайомлення з P, NP, NP-повнотою.

Тема 7.2: Редукція та поняття твердості.

Тема 7.3: Теорема Кука та NP-повнота SAT

Розділ 8: Паралельні та розподілені алгоритми

Тема 8.1: Основи паралельних обчислень і паралельних алгоритмів.

Тема 8.2: Паралельні структури даних і алгоритми.

Тема 8.3: Проблеми синхронізації та спілкування.

Тема 8.4: Розподілені алгоритми та їх застосування

Розділ 9: Застосування обчислювальних алгоритмів

Тема 9.1: Алгоритми в штучному інтелекті, зокрема в машинному навчанні.

Тема 9.2: Алгоритми в криптографії та безпеці.

Тема 9.3: Алгоритми мережевого аналізу та оптимізації

Розділ 10: Етика та соціальні наслідки

Тема 10.1: Алгоритмічна упередженість і справедливість.

Тема 10.2: Проблеми конфіденційності в алгоритмічному прийнятті рішень.

Тема 10.3: Вплив алгоритмів на суспільство: фейкові новини, бульбашки фільтрів тощо.

Розділ 11: Проект і практичне застосування

Тема 11.1: Розробка та реалізація алгоритмів для реальних проблем.

Тема 11.2: Аналіз ефективності реалізованих алгоритмів.

Тема 11.3: Документування та представлення алгоритмічних рішень

Розділ 12: Майбутні тенденції та нові алгоритми

Тема 12.1: Квантові алгоритми та їх потенційний вплив.

Тема 12.2: Нові алгоритмічні проблеми в нових технологіях

Ця структура курсу охоплює широкий спектр тем, від фундаментальних алгоритмічних методів до більш складних концепцій та їх практичного застосування. Він надає студентам всебічне розуміння розробки алгоритмів, аналізу та їхнього значення для вирішення складних обчислювальних задач.

У заключення відмітимо, що кожен розділ може включати додаткові питання, залежно від глибини та широти курсу. Представлені розділи забезпечують структурований підхід до вивчення алгоритмів та їх стратегічних застосувань, а також включають практичні проекти, щоб дають змогу студентам розвинути відповідні практичні навички.

4. Навчальні матеріали та ресурси

Базова:

1. Стивен Скиена Алгоритми. Керівництво по розробці. 2 видання. Київ, Видавництво Print2print, 2018, С. 720
2. Панос Луридас Самый краткий и понятный курс Київ, Видавництво Print2print, 2022, С. 192.
3. Роберт Мартін., Чистий код., Київ, : Видавництво «Фабула», 2019. — 368 с.
4. George T. Heinemann., Algorithms in a Nutshell: A practical Guide., 2 Edition. — 2016, 390 p.
5. Thomas H. Cornmen, Charles E. Leiserson, Roland L. Rivest, Clifford Stein., Introduction to Algorithms., 4 Edition.

Додаткова:

6. Медовз Донела., Мистецтво мислити системно,. Київ, Видавництво «Віват», 2023. — 304 с.
7. Новотарский М.А., Алгоритми та методи обчислень., Київ, Видавництво «КПІ ім. Ігоря Сікорського», 2019, 407 с.
8. Проектування та аналіз обчислювальних алгоритмів. Практикум [Електронний ресурс] : навчальний посібник для здобувачів першого (бакалаврського) рівня вищої освіти за освітньою програмою «Комп’ютерні технології в біології та медицині» спеціальності 122 «Комп’ютерні науки» / КПІ ім. Ігоря Сікорського ; уклад.: І. В. Федорін. – Електронні текстові данні (1 файл: 2,41 Мбайт). – Київ : КПІ ім. Ігоря Сікорського, 2023. – 140 с.
9. Федорін, І. В. Проектування та аналіз обчислювальних алгоритмів: вступ до алгоритмів [Електронний ресурс] : навчальний посібник для здобувачів першого (бакалаврського) рівня вищої освіти за освітньою програмою «Комп’ютерні технології в біології та медицині» спеціальності 122 «Комп’ютерні науки» / І. В. Федорін ; КПІ ім. Ігоря Сікорського. – Електронні текстові дані (1 файл: 1,98 Мбайт). – Київ : КПІ ім. Ігоря Сікорського, 2022. – 116 с.

ОНЛАЙН-КУРСИ З «Проектування та аналіз обчислювальних алгоритмів»

1. https://classroom.google.com/c/NjE5MzE2MTUxOTQ0

Навчальний контент

5. Методика опанування навчальної дисципліни (освітнього компонента)

Лекційні заняття

№	*Назва теми лекції та перелік основних питань*
1	**РОЗДІЛ 1. Вступ до алгоритмічного мислення та аналізу.** **Лекція №1 Введення в обчислювальні алгоритми** Визначення поняття «алгоритм» та показати його значення в обчисленнях. Обговорення процесу розробки алгоритму. Ввести поняття «аналіз алгоритмів і їх складність». Рекомендована література: : [1] – С. 12 – 29, [5]– С. 6 – 44, [6]– С.22 – 29,[1] – С. 8 –12; [2] – С. 8 – 24; [5] – С. 30 – 36, 56 – 58.
2	**Лекція 2: Аналіз алгоритму: нотація Big O** Пояснення позначення великого О та його значення. Розрізняти складність у найкращому випадку, найгіршому випадку та середньому випадку. Аналіз основних алгоритмів за допомогою нотації Big O. Рекомендована література: [1] – С. 24 –29; [2] – С. 8 – 24
3	**Лекція 3: Алгоритми сортування** Ознайомлення з поняттям алгоритмів сортування. Проведення порівняння між сортування бульбашок, сортування вставкою та сортування виділенням. Аналіз їх часових складностей та придатності для різних сценаріїв. Рекомендована література: [1] – С. 66 –77**.**
4	**Розділ 2: Методи проектування алгоритмів** **Лекція 4: Алгоритми розділяй і володарюй** Пояснення парадигми розділяй і володарюй. Обговорення сортування злиттям як приклад. Проаналізуйти часову складність сортування злиттям. Література: [1]– С. 55 –60; [2] – С. 51 – 68.
5	**Лекція 5: Динамічне програмування** Визначити поняття «динамічне програмування» та його застосування. Уявити концепцію підпроблем, що перекриваються, і їх оптимальних підструктур. Проілюструвати прикладами, такими як послідовність Фібоначчі та проблема рюкзака. Література: [1] – С. 70 –77.
6	**Лекція 6: Жадібні алгоритми** Ознайомлення з жадібними алгоритмами та їх характеристиками. Пояснити властивість жадібного вибору та його оптимальну підструктуру. Використати такі приклади, як проблема розміни монет і кодування Хаффмана. Література: [1-5].
7	**Розділ 6: Алгоритми графів** **Лекція 7: Алгоритми графів: пошук у ширину та пошук у глибину** Ознайомитись із алгоритмами обходу графів. Пояснити стратегії BFS і DFS. Обговорити їх застосування для пошуку зв'язаних компонентів і найкоротших шляхів. Література: [1-5].
8	**Лекція 8: Графові алгоритми: алгоритми Дейкстри та Беллмана-Форда** Представити алгоритми пошуку найкоротших шляхів у зважених графах. Обговорити алгоритм Дейкстри та його обмеження. Ознайомитись з алгоритмом Беллмана-Форда для обробки ребер негативної ваги. Література: [1-5].
9	**Лекція 9: Алгоритми графів: Мінімальні остовні дерева** Визначити мінімальні остовні дерева (MST) та їх застосування. Ознайомлення з алгоритмами Крускала та Прима. Порівняйняння їх часової складності та сценаріїв використання. Література: [1-5].
10	**Лекція 10: Динамічне програмування. Частина 2** Охоплення передових методів динамічного програмування. Обговорення підходів до запам'ятовування та табуляції. Застосовування динамічного програмування для вирішення більш складних завдань. Література: [1-5].
11	**Лекція 11: Алгоритми зворотного відстеження** Ознайомлення з поняттям бектрекінгу. Пояснення структури алгоритму зворотного відстеження. Досліджування прикладів, як-от завдання N-Queens і судоку. Література: [1-5].

12	**Лекція 12: Алгоритми зіставлення рядків** Обговорення важливості відповідності рядків. Представити алгоритм грубої сили та Рабіна-Карпа. Проаналізуйте їх часові складності та плюси/мінуси. Література: [1-5].
13	**Лекція 13: Алгоритми обчислювальної геометрії** Знайомство з обчислювальною геометрією та її застосуваннями. Розглядання таких тем, як опукла оболонка, перетин відрізків і діаграми Вороного. Виділення алгоритмічних проблем та рішення в геометричних задачах. Література: [1-5].
14	**Лекція 14: Алгоритми мережевого потоку** Визначення проблеми мережевого потоку та його застосування в реальному світі. Ознайомлення із алгоритмом Форда-Фулкерсона. Обговорення теореми про максимальний потік і мінімальний зріз. Література: [1-5].
15	**Розділ 7: Обчислювальна складність** **Лекція 15: Алгоритми NP-повноти та апроксимації** Пояснення поняття NP-повноти. Представлення проблеми P проти NP. Представити апроксимаційні алгоритми та їх роль у розв'язуванні складних задач. Література: [1-5].
16	**Розділ 5: Розширені алгоритмічні концепції** **Лекція 16: Рандомізовані алгоритми** Обговорення алгоритмів, які використовують випадковість у своєму проектуванні. Ознайомлення з поняттям рандомізації та її перевагами. Дослідження таких прикладів, як рандомізоване швидке сортування та алгоритми Монте-Карло. Література: [1-5].
17	**Розділ 8: Паралельні та розподілені алгоритми** **Лекція 17: Паралельні та розподілені алгоритми** Ознайомлення з концепціями паралельних і розподілених обчислень. Обговорити розробку паралельного алгоритму та його проблеми. Розглянути алгоритми паралельного сортування та паралельного множення матриць. Література: [1-5].
18	**Розділ 12: Розділ 12: Майбутні тенденції та нові алгоритми** **Лекція 18: Нові тенденції в обчислювальних алгоритмах** Дослідження останніх тенденцій, таких як квантові алгоритми та машинне навчання. Обговорення та вплив цих тенденцій на розробку алгоритмів. Заохочення студентів бути в курсі досягнень у цій галузі. Література: [1-5].

Практичні заняття

№	Назва теми занять
1	ПЗ 1. Реалізація та аналіз алгоритму. Основні питання: Запросити студентів реалізувати алгоритми сортування (наприклад, бульбашкове сортування, швидке сортування) з нуля та порівняти їх ефективність, використовуючи різні набори даних. Використання різних алгоритмів вирішення таких проблем, як проблема ранця або проблема найдовшої загальної підпослідовності. Аналіз часової та просторової складності їх реалізації та їх порівняння із теоретичними прогнозами.
2	ПЗ 2. Виклики кодування. Основні питання: Запропонувати завдання для кодування, які вимагають використання алгоритмів графів, як-от пошук найкоротших шляхів або виявлення циклів. Запропонувати студентам проблеми маніпулювання рядками, як-от зіставлення шаблонів або виявлення паліндрому за допомогою різних алгоритмів.
3	ПЗ 3. Проектне навчання: Основні питання: Призначити студентам проекти, які передбачають вирішення

	реальних проблем за допомогою алгоритмів, наприклад рішення диференціальних чи інтегральних рівнянь, систем рівнянь, мінімізації функцій тощо.
4	ПЗ 4. Візуалізація алгоритму. Основні питання: Заохотити студентів створювати візуалізації виконання алгоритму за допомогою таких інструментів, як D3.js або matplotlib Python. Візуалізація алгоритмів сортування може бути особливо проникливою.
5	ПЗ 5. Вправи на аналіз складності. Основні питання: Надати студентам фрагменти коду та попросити їх проаналізувати часову та просторову складність даного коду. Нехай вони виявлять вузькі місця та запропонують оптимізацію.
6	ПЗ 6. Алгоритмічне вирішення задач. Основні питання: Познайомити студентів з такими онлайн-платформами, як LeetCode, HackerRank або Codeforces для вирішення алгоритмічних задач. Встановіть щотижневі або раз на два тижні завдання з вирішення проблем, щоб розвинути їхні навички.
7	ПЗ 7. Спільні проекти. Основні питання: Призначити групові проекти, де студенти спільно працюють над складнішими алгоритмічними завданнями або проблемами, які потребують інтеграції кількох алгоритмів.
8	ПЗ 8. Алгоритм аналізу з реальними даними. Основні питання: Надати студентам реальні набори даних і попросити їх проаналізувати дані за допомогою відповідних алгоритмів. Наприклад, провести аналіз даних соціальних мереж за допомогою графових алгоритмів.
9	ПЗ 9. Алгоритмічні дослідження та презентація. Основні питання: Призначити студентам теми, пов'язані з останніми досягненнями або дослідженнями в області обчислювальних алгоритмів. Попросити студентів провести дослідження та представити свої висновки.
10	ПЗ 10. Хакатони та конкурси кодування: Основні питання: Організовати студентам хакатони або змагання з кодування в класі, де студенти зможуть застосувати свої алгоритмічні знання для вирішення завдань, визначених на час. Запросити студентів провести свої дослідження та представити свої висновки.

6. Самостійна робота студента

Самостійна робота студента складається з виконання індивідуального завдання – написання реферативної роботи на одну із запропонованих тем:

1. **Важливість алгоритмів**:

Алгоритми є фундаментальними для інформатики та багатьох інших галузей. Вони дозволяють комп'ютерам ефективно вирішувати проблеми та використовуються в різних програмах, від розробки програмного забезпечення до штучного інтелекту.

2. **Алгоритмічні парадигми**:

Окрім конкретних алгоритмів, необхідно осягнути ключові алгоритмічні парадигми, такі як розділяй і володарюй, динамічне програмування та жадібні алгоритми. Розуміння цих підходів допомагає у вирішенні широкого кола проблем.

3. **Алгоритм аналізу за межами великого О**:

Хоча нотація Big O є важливою, передові методи аналізу, такі як нотація Омега (Ω) і Тета (Θ), забезпечують більш повне уявлення про продуктивність алгоритму.

4. **Алгоритми апроксимації**:

Обговорити алгоритми, які знаходять майже оптимальні рішення для складних проблем, коли точні рішення важко отримати. Це пов'язує алгоритмічну теорію з практичним вирішенням проблем.

5. **Практична оптимізація**:

Звернути увагу на практичні методи оптимізації, такі як запам'ятовування та скорочення, які можуть значно підвищити ефективність алгоритмів у сценаріях реального світу.

6. **Компроміси**:

Врахувати компромісів між складністю часу, складністю простору та іншими факторами при виборі алгоритмів для конкретних завдань.

7. **Паралельні та розподілені алгоритми**:

Зі зростанням паралельних і розподілених обчислень все більш важливим стає розуміння того, як алгоритми можуть бути розроблені для використання переваг цих архітектур.

8. **Онлайн-алгоритми**:

Познайомити з концепцією алгоритмів, які обробляють дані в режимі реального часу, приймаючи рішення з обмеженою інформацією, яка застосовна в таких сценаріях, як онлайн-реклама та фінанси.

9. **Квантові алгоритми**:

Надати огляд квантових алгоритмів, підкреслюючи їхній потенціал, експоненціально швидке розв'язування певних проблеми, ніж класичні алгоритми, що має значення для криптографії та оптимізації.

10. **Алгоритмічне упередження та справедливість**:

Обговорити етичні міркування, пов'язані з алгоритмами, зокрема упередженість у алгоритмічному прийнятті рішень і важливість розробки справедливих і підзвітних алгоритмів.

11. **Алгоритми налагодження та профілювання**:

Розвиток навичок налагодження та профілювання алгоритмів, щоб ефективно виявляти та вирішувати проблеми продуктивності та помилки.

13. **Реальні програми**:

Продемонструйте різноманітні застосування алгоритмів, починаючи від систем рекомендацій і обробки зображень до оптимізації маршрутів і мовного перекладу.

14. **Найкращі практики кодування**:

Поряд з алгоритмічними техніками, акцентувати увагу на найкращих практиках кодування, щоб гарантувати, що можливо не лише вирішувати проблеми, але й писати чистий, зручний і ефективний код.

Викладач, який веде практичні заняття, перевіряє самостійні роботи студентів та виставляє їм відповідні рейтингові бали у двотижневий термін з призначеної дати здачі студентами індивідуального завдання.

Політика та контроль

7. Політика навчальної дисципліни (освітнього компонента)

Система вимог, які викладач ставить перед студентом:

- правила відвідування занять: відповідно до Наказу 1-273 від 14.09.2020 р. заборонено оцінювати присутність або відсутність здобувача на аудиторному занятті, в тому числі

нараховувати заохочувальні або штрафні бали. Відповідно до РСО даної дисципліни бали нараховують за відповідні види навчальної активності на лекційних та практичних заняттях.

- правила поведінки на заняттях: студент має можливість отримувати бали за відповідні види навчальної активності на лекційних та практичних заняттях, передбачені РСО дисципліни. Використання засобів зв'язку для пошуку інформації на гугл-диску викладача, в інтернеті, в дистанційному курсі на платформі Сікорський здійснюється за умови вказівки викладача;
- політика дедлайнів та перескладань: якщо студент не проходив або не з'явиться на МКР (без поважної причини), його результат оцінюється у 0 балів. Перескладання результатів МКР не передбачено;
- політика щодо академічної доброчесності: Кодекс честі Національного технічного університету України «Київський політехнічний інститут» https://kpi.ua/files/honorcode.pdf встановлює загальні моральні принципи, правила етичної поведінки осіб та передбачає політику академічної доброчесності для осіб, що працюють і навчаються в університеті, якими вони мають керуватись у своїй діяльності, в тому числі при вивченні та складанні контрольних заходів з дисципліни «Проектування та аналіз обчислювальних алгоритмів»;
- при використанні цифрових засобів зв'язку з викладачем (мобільний зв'язок, електронна пошта, переписка на форумах та у соцмережах тощо) необхідно дотримуватись загальноприйнятих етичних норм, зокрема бути ввічливим та обмежувати спілкування робочим часом.

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

Семестровий контроль: Залік

1. **Види контролю.**

Поточний контроль: вправи на лекційних заняттях, тестування, МКР, виконання завдань до практичних занять.

Календарний контроль: провадиться двічі на семестр, як моніторинг поточного стану виконання вимог силабусу. Семестровий контроль: залік.
Умови допуску до семестрового контролю: виконані завдання до практичних занять, семестровий рейтинг більше 30 балів.

Семестровий рейтинг з дисципліни «Проектування та аналіз обчислювальних алгоритмів» складається з рейтингових балів (див. табл.1), і не перевищує $R_{\max} = 100$. В семестрі студент може набрати **60 балів**, відповідно на іспиті – **40 балів**.

Таблиця 1. Система рейтингових балів.

№	Контрольний захід	Бали
1.	*Реферативна робота*	**15**
2.	*Модульна контрольна робота «Проектування та аналіз обчислювальних алгоритмів»*	**30**
3.	*Активна робота студента на практичних заняттях*	**15**

2. Реферативна робота зараховується тільки за умови її захисту студентом. Для захисту реферативної роботи студенту надається не більше двох спроб. В залежності від того, з якої спроби була захищена робота, нараховується наступна кількість балів:

- захист з першої спроби - 30 балів;
- захист з другої спроби - 20 балів.

3. Студент допускається до заліку при виконанні умов:
    - поточний рейтинг за семестр складає не нижче 30 балів;
    - захищена реферативна робота.

Відповідно сумарної кількості балів, що набрані в семестрі та на заліку, студентом отримує оцінку згідно таблиці 2.

Таблиця 2 відповідності рейтингових балів оцінкам за університетською шкалою:

Рейтинг	***Оцінка ECTS***	***Традиційна оцінка***
95 - 100	*відмінно*	*Відмінно*
85 - 94	*дуже добре*	*Добре*
75 - 84	*добре*	
65 - 74	*задовільно*	*Задовільно*
60 - 64	*достатньо*	
менше 60 балів	*незадовільно*	*Незадовільно*
менше 30 балів	*не допущено*	*Не допущено*

9. Додаткова інформація з дисципліни (освітнього компонента)

Теоретичні питання:

1. Важливість алгоритмів.
2. Алгоритмічні парадигми.
3. Алгоритми апроксимації.
4. Компроміси: Навчитися враховувати компроміси між складністю часу, складністю простору та іншими факторами при виборі алгоритмів для конкретних завдань.
5. Паралельні та розподілені алгоритми. Зі зростанням паралельних і розподілених обчислень все більш важливим стає розуміння того, як алгоритми можуть бути розроблені для використання переваг цих архітектур.
6. Алгоритмічні зміщення та справедливість.
7. Алгоритми налагодження та профілювання.
8. Бібліотеки алгоритмів і фреймворки.
9. Програми реального світу.

Робочу програму навчальної дисципліни (силабус):

Складено *к.т.н., доцент, Тимошенко Юрій Олександрович*

Ухвалено кафедрою ШІ (*протокол № 14 від "11" червня 2024 р.*)

Погоджено Методичною комісією ННІПСА (*протокол № 10 від "24" червня 2024 р.*)