ДЕРЖАВНА СЛУЖБА СТАТИСТИКИ УКРАЇНИ
НАЦІОНАЛЬНА АКАДЕМІЯ СТАТИСТИКИ, ОБЛІКУ ТА АУДИТУ

КАФЕДРА СТАТИСТИКИ, ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ТА МАТЕМАТИЧНИХ МЕТОДІВ В ЕКОНОМІЦІ

МЕТОДИЧНІ ВКАЗІВКИ

щодо виконання курсової роботи з дисципліни

«Об’єктно-орієнтоване програмування»

для здобувачів **першого (бакалаврського) рівня вищої освіти**
галузі знань **F «Інформаційні технології»**
спеціальності **F3 «Комп'ютерні науки»; F6 «Інформаційні системи і технології»**
освітньо-професійної програми **«Комп'ютерні науки та інформаційні технології»**

Київ 2025 р.

Ставицький О.В., Савченко С.С. Методичні вказівки щодо виконання курсової роботи з дисципліни «Об’єктно-орієнтоване програмування» для студентів денної форми навчання перший (бакалаврський) рівень галузі знань F «Інформаційні технології», спеціальностей F3 «Комп'ютерні науки»; F6 «Інформаційні системи і технології». Київ: НАСОА, 2025. 36 с.

Рецензенти:

Д. О. Байдінов старший виклатач, кафедри статистики, інформаційних технологій та математичних методів в економіці НАСОА.

М. В. Надутенко кандидат технічних наук, в.о. керівника відділу інформатики, заступник директора УМІФ НАН України.

Затверджено на засіданні кафедри статистики, інформаційних технологій та математичних методів в економіці, протокол від «27» серпня 2025 року №1.

Схвалено Вченою радою обліково-статистичного факультету НАСОА, протокол від «28» серпня 2025 року №1.

ЗМІСТ

ЗАГАЛЬНІ ПОЛОЖЕННЯ

Підготовка фахівців високого рівня потребує ґрунтовного вивчення дисциплін циклу професійної підготовки та виконання задач прикладного характеру. Самостійна робота студента передбачає виконання цілої низки дій з розробки програмного забезпечення від постановки задачі до програмної реалізації, випуску програмної документації та захисту роботи. В процесі виконання роботи, студент свої знання поглиблює та набуває практичних навичок використання методів і засобів розробки програмного забезпечення, закріплює раніш пройдений теоретичний матеріал.

Курсова робота є самостійною, творчою практичною роботою, в процесі написання якої студент виявляє своє вміння відбирати, систематизувати і творчо осмислювати технічну документацію, працювати із спеціальною літературою, робити правильні і обґрунтовані висновки, знаходити в них головне, формулювати та логічно викладати свої думки.

Виконання письмових робіт сприяє поглибленому вивченню навчального предмету студентами і є однією з важливих форм перевірки їх знань. Таким чином, курсова робота дає змогу викладачу оцінити якість теоретичних знань студента та його вміння застосовувати їх на практиці.

Згідно з навчальними планами для студентів спеціальності 122 «Комп’ютерні науки та інформаційні технології» передбачена курсова робота з дисципліни «Об’єктно-орієнтоване програмування». Написання курсової роботи є завершальним етапом вивчення дисципліни «Об’єктно-орієнтоване програмування». Для написання курсової роботи студенту необхідно вивчити технічну документацію, методичні, інструктивні матеріали та літературні джерела з обраної теми. Тема курсової роботи обирається згідно рекомендованого переліку тем, що не позбавляє можливості студента за погодженням із кафедрою сформулювати тему самостійно. Курсова робота має бути подана до кафедри до початку екзаменаційної сесії та після проходження формальної перевірки допущена до захисту.

МЕТА ТА ЗАВДАННЯ КУРСОВОЇ РОБОТИ

Мета курсової роботи – поглибити теоретичні знання набуті студентами у процесі вивчення дисципліни і виробити вміння застосувати їх у практичній діяльності.

Завдання курсової роботи – навчити студентів користуватися та опрацьовувати технічну літературу, сприяти поглибленому засвоєнню набутих знань студентам з дисципліни «Об'єктно-орієнтоване програмування», ґрунтовному вивченню обраної мови програмування та популярних інструментів розробки. Виконання курсової роботи повинно сприяти формуванню у студента навичок творчої, дослідницької роботи та компетентностей. Професійні компетентності, яких набувають студенти внаслідок вивчення навчальної дисципліни.

Професійні компетентності, яких набувають студенти в наслідок вивчення навчальної дисципліни:

Загальні компетентності:

ЗК2. Здатність застосовувати знання у практичних ситуаціях.
ЗК7. Здатність до пошуку, оброблення та аналізу інформації з різних джерел.
ЗК11. Здатність приймати обґрунтовані рішення.
ЗК12. Здатність оцінювати та забезпечувати якість виконуваних робіт.

Спеціальні компетентності:

СК8. Здатність проектувати та розробляти програмне забезпечення із застосуванням різних парадигм програмування: узагальненого, об'єктно-орієнтованого, функціонального, логічного, з відповідними моделями, методами й алгоритмами обчислень, структурами даних і механізмами управління.
СК12. Здатність забезпечити організацію обчислювальних процесів в інформаційних системах різного призначення з урахуванням архітектури, конфігурування, показників результативності функціонування операційних систем і системного програмного забезпечення.
СК16. Здатність реалізовувати високопродуктивні обчислення на основі хмарних сервісів і технологій, паралельних і розподілених обчислень при розробці й експлуатації розподілених систем паралельної обробки інформації.

Результати навчання:

ПР13. Володіти мовами системного програмування та методами розробки програм, що взаємодіють з компонентами комп'ютерних систем, знати мережні технології, архітектури комп'ютерних мереж, мати практичні навички технології адміністрування комп'ютерних мереж та їх програмного забезпечення.
ПР15. Розуміти концепцію інформаційної безпеки, принципи безпечного проектування програмного забезпечення, забезпечувати безпеку комп'ютерних мереж в умовах неповноти та не визначеності вихідних даних.

ПР16. Виконувати паралельні та розподілені обчислення, застосовувати чисельні методи та алгоритми для паралельних структур, мови паралельного програмування при розробці та експлуатації паралельного та розподіленого програмного забезпечення.

СТРУКТУРА КУРСОВОЇ РОБОТИ

У вирішенні задачі курсової роботи, потрібно виділяти наступні основні етапи, кожен з яких, у свою чергу, складається з множини взаємозв'язаних задач:

- постановка задачі;
- проектування програми;
- програмування;
- налагодження та тестування програми;
- оформлення пояснювальної записки та захист проекту.

Основним змістом першого етапу є формулювання призначення програми та постановка цілей, а також виявлення оригінальних задач, які будуть вирішуватися в процесі виконання роботи. У зв'язку з цим виконання етапу потрібно почати із з'ясування (уточнення) основних понять і визначень, що зустрічаються в завдані і даній предметній області та огляду наявних аналогів. Для цих цілей може бути використана рекомендована література та інші джерела.

Після завершення аналітичного дослідження можна переходити до проектування програми. При цьому необхідно визначити структуру вхідних та вихідних даних, структуру функціональних модулів та взаємозв'язків між ними, розробити алгоритми роботи цих модулів та інтерфейс користувача.

На етапі програмування відбувається реалізація запропонованих архітектурних рішень у вигляді програмного коду. Варто зазначити, що крім формальної реалізації алгоритмів необхідно приділяти увагу стилю, дотриманню загально прийнятих та рекомендованих норм та правил найменування функціональних елементів та оформлення програмного коду загалом. Особливу увагу необхідно приділити розробці інтерфейсу користувача.

Наступним етапом розробки є тестування програми та детальний опис структури головного, допоміжного меню та діалогових вікон, а також рекомендації з використання програми.

Оформлення пояснювальної записки є останнім етапом виконання курсового проекту і має за мету надати студентові навичок документування програмного продукту. Документування є завершальним етапом створення програмного виробу для курсової роботи. Вимоги до оформлення пояснювальної записки наведені в наступному розділі.

Структура курсової роботи

Курсова робота повинна містити:

1. Титульний аркуш.
2. Завдання на курсову роботу та календарний план її виконання.
3. Вступ.
4. Основна частина курсової роботи.
5. Висновки.
6. Список використаних джерел (не менше 15, включаючи нормативні акти).
7. Додатки (при необхідності).

Вимоги до змісту курсової роботи. Після затвердження теми курсової роботи студент за погодженням з науковим керівником розробляє план роботи, підбирає список літератури і опрацьовує необхідні нормативно-правові акти, а також визначає строки і складає орієнтовний графік написання курсової роботи. Обов'язковим правилом вибору теми є те, що йому має передувати належна теоретична підготовка, тобто відповідне оволодіння необхідним матеріалом з предмету даного дослідження.

Курсова робота повинна бути написана відповідно до плану, який розробляється студентом. При роботі над обраною темою, студенту необхідно спочатку вивчити підібрані джерела. При цьому доцільно робити виписки у вигляді цитат або вільного переказу окремих положень. Після вивчення необхідної літератури, зібравши достатній матеріал для висвітлення теми, слід приступити до його повної обробки, упорядкування, аналізу і викладу згідно зі складеним планом. Студент повинен виразно, розбірливо, ясно, зрозуміло та логічно висловлювати свої та запозичені з використаних джерел думки. Необхідно, щоб автор при виконанні роботи висловлював власне ставлення до описуваних подій та оцінок їх у зібраній літературі. Тобто при написанні курсової роботи потрібно уникати поверхневого висвітлення, загальних фраз, а особливо, дослівного переписування з використаних джерел. Останнє призведе до негативної оцінки роботи викладачем і повернення її на доопрацювання.

В загальному, курсова робота виконується акуратно, грамотно, а основні її положення та ідеї мають бути чітко і правильно сформульовані. Структурні елементи курсової роботи повинні відповідати встановленим вимогам:

Титульний аркуш курсової роботи містить найменування міністерства, вищого навчального закладу, факультету та кафедри, де виконана робота; тему курсової роботи; прізвище, ім'я, по батькові автора; вчене звання, наукову ступінь прізвище, ім'я, по батькові наукового керівника; місто і рік **(додаток А)**.

Зміст подають на початку курсової роботи, він містить найменування усіх розділів, підрозділів та пунктів, підпунктів (якщо вони мають заголовки), зокрема: вступу, розділів, підрозділів, загальних висновків, списку використаних джерел і додатків.

Перелік умовних позначень, символів, одиниць, скорочень і термінів, а також використані маловідомі скорочення, нові символи, позначення та їх перелік може бути поданий у курсовій роботі у вигляді окремого списку, який розміщується перед вступом. Перелік треба друкувати двома колонками, в яких зліва за абеткою наводять, наприклад, скорочення, а з права – їх детальну розшифровку. Якщо в курсовій роботі спеціальні терміни, позначення, скорочення повторюються менше трьох разів – перелік не складають, а їх розшифровку наводять у тексті при першому згадуванні.

У вступі коротко викладають оцінку сучасного стану проблеми, актуальність даної роботи і підстави для її проведення, ціль роботи й галузь застосування.

Основна частина складається з розділів і підрозділів (пунктів і підпунктів). Кожен розділ повинен починатися з нової сторінки.

Перший розділ – дається аналітичний огляд можливостей побудови систем, що вирішують поставлену задачу. Розглядаються особливості об'єктноорієнтованого підходу і мов, що підтримують ООП. Наводиться обґрунтування обраного програмного забезпечення.

Другий розділ – проектний. В цьому розділі можна виділити 2–3 відносно самостійних параграфів, що розкривають теоретичні та практичні аспекти розробки програмного забезпечення.

З'ясовується семантика класів і об'єктів — визначаються поведінка і атрибути кожної абстракції, визначається структура вхідних та вихідних даних, структура функціональних модулів та взаємозв'язків між ними, розробляється інтерфейс користувача.

Розділ повинен бути максимально насичений фактичною інформацією (таблиці, схеми, UML діаграми) та всебічно відображати аспекти діяльності програмного забезпечення, що розробляється.

Третій розділ – присвячений реалізації рішень, запропонованих та спроектованих в попередньому розділі. Структурно третій розділ курсової роботи вміщує 2 – 3 параграфи, які містять детальний опис роботи всіх блоків програмного забезпечення, надається опис отриманих результатів і інструкція з використання розробленої програми.

У висновках викладають найбільш важливі практичні результати, одержані в процесі виконання курсової роботи. У висновках необхідно наголосити на якісних та кількісних показниках здобутих результатів: наголосити на архітектурних рішеннях., які було використано, зазначити, які функціональні вимоги вдалось реалізувати, оцінити швидкодію та обчислювальну складність реалізованих алгоритмів.

ПРАВИЛА ОФОРМЛЕННЯ КУРСОВОЇ РОБОТИ

Загальні принципи оформлення

Як правило, курсова робота спочатку виконується у чорновому варіанті. Лише після узгодження з керівником дискусійних і сумнівних моментів, врахування зауважень і усунення недоліків, удосконалення структури викладу матеріалу приступають до чистового варіанту. Чистовий варіант курсової роботи потрібно акуратно підшити у папку.

Курсова робота виконується державною (українською) мовою.

Курсова робота виконується на стандартному папері формату А4 (210×297 мм) білого кольору до тридцяти рядків на сторінці. При наборі використовують шрифт Times New Roman 14 з міжрядковим інтервалом 1,5.

Текст необхідно друкувати, залишаючи поля таких розмірів: ліве – не менше 25 мм, праве – не менше 10 мм, верхнє – не менше 20 мм, нижнє – не менше 20 мм.

Обсяг основного тексту курсової роботи повинен становити не менше 30 та не більше 50 сторінок (без врахування таблиць та рисунків).

Кількість використаних джерел у курсовій роботі повинна складати не менше 15 найменувань.

У викладі матеріалу важливо дотримуватись принципу пропорційності, який полягає у дотриманні пропорцій між обсягами окремих частин роботи. При цьому на вступ відводиться 1–2 сторінки, на основну частину роботи 30– 40 сторінок, на титульний лист і зміст роботи – по одній сторінці, висновки та пропозиції – 2 сторінки.

Заголовки структурних частин дипломної роботи «ЗМІСТ», «ВСТУП», «РОЗДІЛ», «ВИСНОВКИ ТА ПРОПОЗИЦІЇ», «СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ», «ДОДАТКИ» (14 шрифт, жирно) пишуть великими літерами симетрично до тексту. Заголовки параграфів друкують маленькими літерами з абзацного відступу. Крапку в кінці заголовка не ставлять. Всі сторінки курсової роботи повинні бути пронумеровані арабськими цифрами без знака № в правому верхньому куті.

Першою сторінкою курсової роботи є титульний лист (приклад подано у **додатку А**), який включають до загальної нумерації сторінок. На ньому номер сторінки не ставлять.

На другій сторінці розміщують завдання на курсову роботу та календарний план її виконання (бланк завдання подано у **додатку Б**). зміст курсової роботи (приклад подано у **додатку В**). Зверху потрібно написати: **ЗМІСТ**.

З наступної сторінки розпочинається виклад тексту роботи. Кожний розділ пишуть з нової сторінки і обов'язково вказують його найменування (виділяють заголовок). Номер розділу ставлять після слова «РОЗДІЛ», після номера крапку не ставлять, потім з нового рядка друкують заголовок розділу.

Параграфи нумерують у межах кожного розділу. Номер параграфу складається з номера розділу і порядкового номера параграфу, між якими ставлять крапку. В кінці номера параграфу повинна стояти крапка, наприклад: «2.1» (перший параграф другого розділу). Далі у тому ж рядку друкується заголовок параграфу.

Всі аркуші, на яких розміщені такі структурні частини курсової роботи як зміст, вступ, висновки, список використаних джерел, додатки нумерують звичайним чином. Не нумерують лише їх заголовки, тобто не можна друкувати «1. ВСТУП» або «4. ВИСНОВКИ».

Оформлення таблиць та ілюстрацій

Для аналізу необхідно використовувати таблиці та ілюстрації: схеми, графіки, діаграми тощо.

Цифровий матеріал, як правило, оформляють у вигляді таблиць. Кожна таблиця повинна мати назву, яку розміщують над таблицею симетрично до тексту. Назву і слово «Таблиця» починають з великої літери. Назву не підкреслюють. В дужках над змістом таблиці вказують одиниці виміру. Заголовки граф повинні починатися з великих літер, підзаголовки – з маленьких (якщо вони складають одне речення із заголовком) і з великих (якщо вони **є** самостійними).

Таблиці нумерують послідовно в межах розділу. В правому верхньому куті над відповідним заголовком таблиці розміщують напис «Таблиця» із зазначенням номера. Номер таблиці повинен складатися з номера розділу і порядкового номера таблиці, між якими ставиться крапка, наприклад: «Таблиця 2.5» (п'ята таблиця другого розділу).

При переносі частини таблиці на інший аркуш слово «Таблиця» і її номер вказують один раз справа над першою частиною таблиці, над іншими частинами пишуть «продовження таблиці» і вказують номер таблиці, наприклад: «Продовження таблиці 2.5». Таблицю з великою кількістю граф можна ділити на частини і розміщувати одну частину під іншою в межах однієї сторінки.

Ілюстрації позначають словом «Рис.» і нумерують послідовно в межах розділу (за винятком ілюстрацій, поданих у додатках). Номер ілюстрації повинен складатися з номера розділу і порядкового номера ілюстрації, між якими ставиться крапка. Наприклад, **Рис. 1.1** (перший рисунок першого розділу). Номер

ілюстрації, її назва і пояснювальні підписи розміщують послідовно під ілюстрацією. Якщо у роботі подано ілюстрацію, то її нумерують за загальними правилами.

Посилання на таблиці та ілюстрації не варто оформляти як самостійні фрази, в яких лише повторюється те, що міститься у підписі. У тому місці, де викладається матеріал, пов’язаний з таблицею чи ілюстрацією, на якому необхідно зосередити увагу читача, розміщують посилання у вигляді виразу у круглих дужках «(табл. 2.5)» або зворот типу: «... як це видно з рис. 1.1». (приклади оформлення таблиць, рисунків наведено у **додатках Г, Д**).

Написання формул

При необхідності використовують формули. їх виділяють з тексту вільними рядками. Над і під кожною формулою потрібно залишити не менше одного вільного рядка. Якщо рівняння чи формула не вміщаються в одному рядку, їх слід перенести після знака рівності (=) або після інших розділових, знаків (+, –, :, ×). Невеликі і нескладні формули, що не мають самостійного значення, вписують всередині рядків тексту.

Пояснення значень символів і числових коефіцієнтів подають безпосередньо під формулою в тій послідовності, в якій вони наведені у формулі. Значення кожного символу і числового коефіцієнта треба подавати з нового рядка, який починають зі слова «де» без двокрапки. Нумерувати слід лише ті формули, на які є посилання у наступному тексті. Формули у курсовій роботі (якщо їх більше однієї) нумерують у межах розділу. Номер формули складається з номера розділу і порядкового номера формули в розділі, між якими ставлять крапку. Номери формул пишуть біля правого поля аркуша на рівні відповідної формули в круглих дужках, наприклад: (2.3) (третя формула другого розділу) (**додаток Е**).

Посилання на використані джерела

При написанні курсової роботи студент повинен робити посилання на використані літературні джерела. Посилання робляться тоді, коли в тексті використовують цитати чи вислови, переповідається своїми словами думка того чи іншого автора, наводяться цифри і факти, використовуються таблиці або ілюстрації з певного джерела тощо. Вони дають необхідну інформацію щодо процитованого матеріалу, допомагають з’ясувати його зміст, мову тексту, а також дають змогу відшукати документи і перевірити їх достовірність.

Загальні вимоги до цитування:

– текст цитати береться в лапки і наводиться дослівно;

– цитування повинно бути повним (у випадку пропуску слів в цитаті ставляться три крапки), без перекручень думок автора;

– кожна цитата повинна мати посилання на джерело;

– при непрямому цитуванні слід також посилатись на джерело, а думка автора повинна передаватись максимально точно.

Посилання в тексті курсової роботи роблять згідно їх переліку у квадратних дужках, наприклад [2, с. 25], де перша цифра – номер джерела у списку використаних джерел, друга – сторінка, з якої взята цитата. Якщо посилання робиться лише на джерело (без використання цитати), – у квадратних дужках вказується тільки одна цифра, яка відображає порядковий номер джерела у списку використаних джерел, наприклад [25]. Посилатися слід на останні видання публікації.

Приклад.

Цитата в тексті: Економісти Е.С. Хандріксен та М.Ф. Ван Бреда зазначають, що «витрати є несприятливим рухом ресурсів, який зменшує прибуток фірми. Проте не кожен несприятливий рух є витратами. Більше того, витрати – це вико ристання або споживання товарів і послуг у процесі отримання доходу» [26, с. 243].

Відповідне джерело у переліку посилань:

26. Хандриксен Е.С., Ван Бреда М.Ф. Теорія бухгалтерського обліку / гл. ред. Я.В. Соколов. К. : Фінанси та статистика, 2023. 576 с.

На всі ілюстрації, формули та таблиці у роботі повинні бути посилання в тексті. Посилання на них вказують порядковим номером. Наприклад:

– ілюстрації – «рис. 1.1»;

– «... у формулі 2.5»; –

«...в табл. 2.5».

У повторних посиланнях на таблиці та ілюстрації пишуть скорочено слово «дивись», наприклад: «(табл. 3.2)». При цьому слово «таблиця» в тексті пишуть скорочено. Посилаючись на джерело, слід мати на увазі, що кількість посилань має бути оптимальною, тобто не надто надмірною чи мізерною.

Формування списку використаних джерел

Список використаних джерел розміщують після висновків. У списку повинні наводитись лише ті джерела, які дійсно використовувалися автором при написанні курсової роботи.

Список використаних джерел формується одним в алфавітному порядку прізвищ перших авторів або заголовків.

Слід наголосити, що назва джерел подається тією мовою, якою вони видані. Не потрібно перекладати їх на українську.

Приклад оформлення списку використаних джерел наведено в **додатку Ж**. Бібліографічний опис джерел складають відповідно до чинних стандартів з бібліотечної та видавничої справи, міжнародних і державного стандартів з обов'язковим наведенням назв праць.

Оформлення додатків

Додатки оформляються як продовження курсової роботи на наступних її сторінках або у вигляді окремої частини, розміщуючи їх у порядку появи посилань у тексті.

Додаток повинен мати заголовок. Він пишеться угорі малими літерами з першої великої симетрично відносно тексту сторінки. Посередині рядка над заголовком малими літерами з першої великої пишуть (друкують) слово «Додаток» і велика літера, що позначає додаток.

Позначаючи послідовно додатки великими літерами української абетки із нумерації слід виключити літери Ґ, Є, З, І, Ї, Й, О, Ч, Ь. Наприклад, Додаток А, Додаток Б.

При оформленні додатків окремою частиною на окремому аркуші пишуть (друкують) великими літерами слово «ДОДАТКИ»

ЗАХИСТ ТА КРИТЕРІЇ ОЦІНЮВАННЯ КУРСОВОЇ РОБОТИ

Захист курсових робіт відбувається у дні та години, вказані у графіку захисту курсових робіт або відповідно до розкладу занять. Під час захисту студент коротко висловлює зміст роботи, відповідає на запитання викладача та зауваження, висловлені у рецензії. Оцінка за курсову роботу залежить від повноти, викладу теми правильності висвітлення питань, рівня використання теоретичного, методологічного та фактологічного матеріалу; належного оформлення та рівня захисту роботи. Підсумкову, диференційовану оцінку курсової роботи визначає комісія, що призначається кафедрою.

Курсова робота не допускається до захисту і повертається на доопрацювання, якщо:

- роботу представлено до комісії (на перевірку) для рецензування з порушенням термінів, установлених комісією (викладачем, який викладає дану дисципліну);
- роботу написано на тему, що не включена до переліку тем курсових робіт з даної дисципліни або не погоджена з викладачем;
- роботу виконано не самостійно;
- структура і логіка побудови плану роботи не відповідає вимогам та темі курсової роботи;
- курсову роботу не зброшуровано (тобто аркуші не скріплені).

У процесі оцінювання враховується низка важливих показників якості курсової роботи, а саме:

- чіткість формулювання мети і завдання курсової роботи, складність досліджуваних у роботі проблем, відповідність логічної побудови роботи поставленим цілям і завданням;
- якість і глибина теоретико-методичного аналізу проблеми; широта й адекватність методологічного апарату;
- якість критичного огляду літературних джерел, наявність наукової полеміки, посилань на літературні джерела та визначення власної думки студента-автора курсової роботи;
- системність і глибина аналізу статистичних та фактичних матеріалів, наявність і переконливість узагальнень і висновків з аналізу, наявність і якість ілюстративних матеріалів у тексті роботи, використання економікоматематичних методів, соціологічних та інших досліджень;
- наявність та логічний зв'язок заходів, що пропонуються для вирішення проблеми, що досліджується у роботі, з проведеним у роботі аналізом фактичних та статистичних матеріалів, їх актуальність, реальність та економіко-

соціальна обґрунтованість, наявність альтернативних підходів до вирішення визначених проблем;

- володіння культурою презентації, вміння стисло (в межах регламенту), послідовно й чітко викласти сутність і результати дослідження, здатність аргументовано захищати свої пропозиції, думки, погляди; повнота і ґрунтовність відповідей на запитання членів комісії, що приймає захист курсових робіт, на зауваження і пропозиції, що містяться у рецензії на курсову роботу.

АКАДЕМІЧНА ДОБРОЧЕСНІСТЬ

Студенти при виконанні курсової роботи з дисципліни «Об'єктноорієнтоване програмування» повинні дотримуватись академічної доброчесності, тобто самостійно виконувати курсову роботу, здійснювати посилання на джерела інформації у разі використання певних тверджень й відомостей, надавати достовірну інформацію про результати власних розробок та досліджень. В курсовій роботі доцільно уникати плагіату, фальсифікації та фабрикування матеріалів дослідження.

Академічний плагіат – це оприлюднення (частково або повністю) наукових результатів, отриманих іншими особами, як результатів власного дослідження та/або відтворення опублікованих текстів інших авторів без зазначення авторства.

Академічним плагіатом є:

1) відтворення в тексті роботи без змін, з незначними змінами, або в перекладі тексту іншого автора (інших авторів), обсягом від речення і більше, без посилання на автора (авторів) відтвореного тексту;

2) відтворення в тексті роботи повністю або частково, тексту іншого автора (інших авторів) через його перефразування чи довільний переказ без посилання на автора (авторів) відтвореного тексту;

3) відтворення в тексті роботи наведених в іншому джерелі цитат з третіх джерел без вказування, за яким саме безпосереднім джерелом наведена цитата;

4) відтворення в тексті роботи наведеної в іншому джерелі науковотехнічної інформації (крім загальновідомої) без вказування на те, з якого джерела взята ця інформація.

Фальсифікація розуміється, як свідома зміна чи модифікація вже наявних даних, що стосуються власної діяльності або діяльності інших учасників процесу реалізації державної політики у сфері якості освіти, зокрема підробка підписів в офіційних документах.

Під **фабрикуванням** мається на увазі вигадування даних чи фактів, що використовуються у власній діяльності чи діяльності інших учасників в процесі реалізації державної політики в сфері якості освіти.

ТЕМИ КУРСОВИХ РОБОТ

Основна частина курсової роботи має містити виконання наступних завдань:

1. Створити класовий тип згідно із варіантом.
2. Для створення об'єкту динамічного типу передбачити відповідні конструктори (конструктор по замовчанню, конструктор із параметрами, конструктор копіювання).
3. Реалізувати компонентні методи відповідно до варіанту.
4. Розробити демонстраційно - тестуючу програму. Виконати тестування розробленого програмного коду.

Перелік тем:

1. Класовий тип для обробки текстової інформації
2. Класові типи для роботи з чисельними матрицями та векторами
3. Бібліотечні засоби для розв’язування задач лінійної алгебри
4. Класовий тип для роботи з структурами типу «Вектор»
5. Класовий тип для роботи з структурами типу «Бітовий вектор»
6. Клас для роботи з структурами типу «Зв’язний список»
7. Клас для обслуговування структур типу «Черга»
8. Клас для обслуговування структур типу «Дек»
9. Клас для обслуговування структур типу «Черга з пріоритетами»
10. Клас для роботи з множинами
11. Клас для роботи з мультимножинами
12. Клас для роботи з асоціативними масивами (словниками)
13. Клас для роботи з словниками з повтореннями (MultiMap)
14. Клас для роботи з структурами типу «Стек»
15. Клас для роботи з структурами типу «Дерево»

16. Клас для роботи з збалансованими деревами (AVL-дерева)
17. Класи для роботи з неорієнтованими графами
18. Класи для роботи з орієнтованими графами
19. Класи для компактного представлення графів
20. Клас для роботи з дводольними графами
21. Клас для роботи з мультіграфами
22. Клас для роботи з гіперграфами
23. Клас для роботи з табличними інваріантами графів
24. Реалізація алгоритму ізоморфізму графів
25. Модель лінії затримки на основі структури типу «Черга»
26. Побудова засобів обробки числових даних
27. Обчислення з довільною точністю
28. Комп’ютерна модель системи терморегулювання
29. Алгебра бінарних відношень
30. Оптимізація пошуку у великих наборах даних
31. Використання паралельних алгоритмів сортування
32. Алгоритми для роботи з великими графами

ВАРІАНТИ ЗАВДАНЬ ДО КУРСОВОЇ РОБОТИ

Варіант 1. Класовий тип для обробки текстової інформації

1) Створити клас String – символьний масив динамічного типу із змінною кількістю символів. Максимальний розмір масиву – у межах вільного простору оперативної пам'яті.
2) Реалізувати компонентні методи:
- length() – довжина рядка;
- findSubstring() – пошук підрядка;
- replaceSubstring() – пошук та заміна підрядка;
- toUpperCase() – приведення всіх літер до верхнього регістру;
- toLowerCase() – приведення всіх літер до нижнього регістру;
- concat() – конкатенація рядків;
- swap() – обмін значеннями двох рядків.

Література: [1, 9, 10]

Варіант 2. Класові типи для роботи з чисельними матрицями та векторами

1) Створити клас Matrix – двовимірний числовий масив динамічного типу із змінними розмірами.
2) Створити клас Vector – одновимірний числовий масив динамічного типу.
3) Використати шаблони (template) для гнучкості типів матриці та вектора.
4) Реалізувати компонентні методи:
- transpose() – транспонування матриці;
- multiply(Matrix) – множення матриць;
- add(Matrix), subtract(Matrix) – додавання та віднімання матриць;
- compare(Matrix) – порівняння матриць;
- multiply(Vector) – множення матриці на вектор;
- dotProduct(Vector) – скалярний добуток векторів;
- norm() – обчислення норми матриці;
- findSaddlePoints() – пошук сідлових точок матриці;
- findMinMaxElements() – пошук мінімальних та максимальних елементів.

Література: [1-3, 9].

Варіант 3. Бібліотечні засоби для розв'язування задач лінійної алгебри

1) Створити клас LinearAlgebra для обчислення операцій над числовими структурами.

2) Реалізувати перевантажені оператори для класів Matrix і Vector.
3) Реалізувати методи:
- add(Matrix), subtract(Matrix) – додавання та віднімання матриць;
- add(Vector), subtract(Vector) – додавання та віднімання векторів;
- multiply(Matrix, Vector) – множення матриці на вектор;
- rotate(Matrix) – обертання матриці;
- compare(Matrix) – порівняння матриць;
- solveLinearEquations(Matrix, Vector) – розв'язування системи лінійних рівнянь;
- determinant(Matrix) – обчислення визначника;
- vectorLength(Vector) – довжина вектора.

Література: [1, 4, 5, 10].

Варіант 4. Класовий тип для роботи з структурами типу «Вектор»

1) Створити параметризований клас Vector<T> – одновимірний динамічний масив.
2) Передбачити конструктори для ініціалізації вектора стандартними масивами.
3) Реалізувати методи:
- get(int index), set(int index, T value) – доступ до елементів;
- concat(Vector) – конкатенація двох векторів;
- equals(Vector) – порівняння векторів;
- size() – розмір вектора;
- push_back(T), pop_back() – додавання та видалення останнього елемента;
- insert(int index, T value), erase(int index) – вставка та видалення елемента;
- accumulate() – накопичення суми або добутку;
- sort() – сортування.

Література: [1, 3, 6, 9].

Варіант 5. Класовий тип для роботи з структурами типу «Бітовий вектор»

1) Створити параметризований клас BitVector<T>, який є одновимірним динамічним масивом бітових значень.
2) Реалізувати компонентні методи:
- get(int index), set(int index, bool value) – доступ до елемента;
- concat(BitVector) – конкатенація двох векторів;
- equals(BitVector) – порівняння векторів;

- size() – кількість елементів;
- push_back(bool), pop_back() – додавання та видалення останнього бітового елемента;
- swap(BitVector) – обмін значеннями двох векторів;
- bitwiseAnd(BitVector), bitwiseOr(BitVector), bitwiseXor(BitVector), bitwiseNot() – бітові операції.

Література: 1, 6, 10, 13].

Варіант 6. Клас для роботи з структурами типу «Зв'язний список»

1) Створити клас LinkedList<T> для роботи із структурами типу «Двонапрямлений зв'язний список».
2) Реалізувати компонентні методи:

- size() – кількість елементів;
- push_front(T), push_back(T), pop_front(), pop_back() – додавання та вилучення елементів;
- insert(int index, T value), erase(int index) – вставка та видалення елементів;
- swap(LinkedList) – обмін значеннями між списками;
- find(T value) – пошук елемента;
- accumulate() – підсумовування значень у списку;
- sort(), removeDuplicates() – сортування та видалення повторень.

Література: [1, 3, 6, 9, 14].

Варіант 7. Клас для обслуговування структур типу «Черга»

1) Створити клас Queue<T> для роботи із структурами типу «Черга» (FIFO - first-in, first-out).
2) Реалізувати компонентні методи:

- enqueue(T value) – додавання елемента в кінець черги;
- dequeue() – видалення елемента з початку черги;
- front() – доступ до першого елемента;
- back() – доступ до останнього елемента;
- size() – кількість елементів у черзі;
- isEmpty() – перевірка на порожність;
- swap(Queue<T>) – обмін вмістом черг.

Література: [1, 3, 9, 10].

Варіант 8. Клас для обслуговування структур типу «Дек»

1) Створити клас Deque<T> для роботи із структурами типу «Дек» (double-ended queue).
2) Реалізувати компонентні методи:

- push_front(T value), push_back(T value) – додавання елемента на початок і в кінець дека;
- pop_front(), pop_back() – видалення елемента з початку і кінця дека;
- front(), back() – доступ до першого та останнього елементів;
- size(), isEmpty() – отримання розміру та перевірка на порожність;
- swap(Deque<T>) – обмін вмістом деків.

Література: [1, 3, 9, 10].

Варіант 9. Клас для обслуговування структур типу «Черга з пріоритетами»

1) Створити клас PriorityQueue<T> для реалізації черги з пріоритетами.
2) Реалізувати компонентні методи:
- push(T value, int priority) – додавання елемента з пріоритетом;
- pop() – видалення елемента з найвищим пріоритетом;
- top() – доступ до елемента з найвищим пріоритетом;
- size(), isEmpty() – отримання розміру черги та перевірка на порожність.

Література: [1, 3, 9, 10].

Варіант 10. Клас для роботи з множинами

1) Створити клас Set<T> для роботи із множинами.
2) Реалізувати компонентні методи:
- insert(T value), erase(T value) – додавання та видалення елемента;
- contains(T value) – перевірка наявності елемента;
- size(), isEmpty() – отримання розміру множини та перевірка на порожність;
- union(Set<T>), intersection(Set<T>), difference(Set<T>) – операції над множинами.

Література: [1, 3, 9, 10, 12].

Варіант 11. Клас для роботи з мультимножинами

1) Створити клас MultiSet<T>, який дозволяє зберігати повторювані елементи.
2) Реалізувати методи:
- insert(T value), erase(T value) – додавання та видалення елемента;
- count(T value) – підрахунок кількості входжень елемента;
- size(), isEmpty() – отримання розміру та перевірка на порожність;
- union(MultiSet<T>), intersection(MultiSet<T>), difference(MultiSet<T>) – операції над мультимножинами.

Література: [1, 3, 10, 12]

Варіант 12. Клас для роботи з асоціативними масивами (словниками)

1) Створити клас Map<K, V> для реалізації структури «словник» (key-value).
2) Реалізувати методи:
- insert(K key, V value), erase(K key) – додавання та видалення пари ключ-значення;
- find(K key) – отримання значення за ключем;
- size(), isEmpty() – отримання розміру та перевірка на порожність.

Література: [1, 3, 10, 12, 14].

Варіант 13. Клас для роботи з словниками з повтореннями (MultiMap)

1) Створити клас MultiMap<K, V> для збереження множини значень для одного ключа.
2) Реалізувати методи:
- insert(K key, V value), erase(K key) – додавання та видалення пари ключ-значення;
- find(K key) – отримання всіх значень за ключем;
- size(), isEmpty() – отримання розміру та перевірка на порожність.

Література: [1, 3, 10, 12, 14].

Варіант 14. Клас для роботи з структурами типу «Стек»

1) Створити клас Stack<T> для реалізації структури LIFO.
2) Реалізувати компонентні методи:
- push(T value), pop() – додавання та видалення елемента;
- top() – отримання верхнього елемента;
- size(), isEmpty() – перевірка на порожність та отримання розміру.

Література: [1, 3, 10, 12, 14].

Варіант 15. Клас для роботи з структурами типу «Дерево»

1) Створити клас Tree<T> для реалізації деревоподібної структури.
2) Реалізувати методи:
- insert(T value), erase(T value) – додавання та видалення елемента;
- find(T value) – пошук елемента;
- traverseInOrder(), traversePreOrder(), traversePostOrder() – обходи дерева;
- size(), height(), isEmpty().

Література: [1, 3, 10, 11].

Варіант 16. Клас для роботи з збалансованими деревами (AVL-дерева)

1) Створити клас AVLTree<T> – реалізація самобалансованого дерева пошуку.
2) Реалізувати:
- insert(T value), erase(T value), find(T value);
- balance() – балансування дерева після вставки/видалення;
- rotateLeft(), rotateRight() – операції балансування.

Література: [1, 3, 10-12].

Варіант 17. Класи для роботи з неорієнтованими графами

1) Створити клас Graph для представлення неорієнтованого графа.
2) Реалізувати методи:
- addVertex(int v), removeVertex(int v) – додавання та видалення вершини;
- addEdge(int v1, int v2), removeEdge(int v1, int v2) – додавання та видалення ребра;
- getDegree(int v) – визначення степеня вершини;
- traverseDFS(int start), traverseBFS(int start) – обходи графа (DFS, BFS).

Література: [1, 7, 10, 15].

Варіант 18. Класи для роботи з орієнтованими графами

1) Створити клас DirectedGraph для представлення орієнтованого графа.
2) Реалізувати методи:
- addVertex(int v), removeVertex(int v);
- addEdge(int v1, int v2), removeEdge(int v1, int v2);
- getInDegree(int v), getOutDegree(int v) – вхідний та вихідний степені вершини;
- traverseDFS(int start), traverseBFS(int start).

Література: [1, 7, 10, 15].

Варіант 19. Класи для компактного представлення графів

1) Створити класи AdjacencyMatrixGraph та AdjacencyListGraph для різних уявлень графів.
2) Реалізувати методи:
- convertToAdjacencyList(), convertToAdjacencyMatrix() – конвертація між представленнями;
- printGraph() – візуалізація графа.

Література: [1, 7, 10, 15].

Варіант 20. Клас для роботи з дводольними графами

1) Створити клас BipartiteGraph.
2) Реалізувати метод isBipartite(), який перевіряє, чи є граф дводольним.
3) Реалізувати алгоритм розфарбовування вершин.

Література: [1, 7, 10, 15].

Варіант 21. Клас для роботи з мультіграфами

1) Створити клас MultiGraph.
2) Реалізувати можливість додавання декількох ребер між вершинами.

Література: [1, 7, 10, 15].

Варіант 22. Клас для роботи з гіперграфами

1) Створити клас HyperGraph.
2) Реалізувати методи:
- addHyperEdge(vector<int> vertices) – додавання гіперребра;
- removeHyperEdge(vector<int> vertices) – видалення гіперребра.

Література: [1, 7, 10, 15].

Варіант 23. Клас для роботи з табличними інваріантами графів

1) Створити клас GraphInvariants.
2) Реалізувати методи:
- calculateDegreeSequence() – вектор степенів вершин;
- calculateDiameter() – обчислення діаметра графа;
- isBipartite() – перевірка на дводольність.

Література: [1, 7, 10, 15].

Варіант 24. Реалізація алгоритму ізоморфізму графів

1) Створити клас GraphIsomorphism.
2) Реалізувати алгоритм перевірки ізоморфності методом повного перебору.

Література: [1, 7, 10, 15].

Варіант 25. Модель лінії затримки на основі структури типу «Черга»

1) Створити клас DelayLineQueue, що моделює лінію затримки.
2) Врахувати ефекти затримки, ослаблення сигналу та його спотворення.

Література: [1, 6, 7, 10].

Варіант 26. Побудова засобів обробки числових даних

1) Створити клас DataProcessor.
2) Реалізувати методи апроксимації:
- linearApproximation() – лінійна апроксимація;
- polynomialApproximation(int degree) – поліноміальна апроксимація.

Література: [1, 6, 7, 10].

Варіант 27. Обчислення з довільною точністю

1) Створити клас BigNumber для роботи з великими числами.
2) Реалізувати основні операції:
- add(BigNumber), subtract(BigNumber), multiply(BigNumber), divide(BigNumber).

Література: [1, 6, 7, 10].

Варіант 28. Комп'ютерна модель системи терморегулювання

1) Створити класи TemperatureObject, Environment, Controller.
2) Реалізувати регулятори:
- proportionalControl(), integralControl(), derivativeControl().

Література: [1, 6, 7, 10].

Варіант 29. Алгебра бінарних відношень

1) Створити клас BinaryRelation.
2) Реалізувати методи:
- isReflexive(), isSymmetric(), isTransitive().

Література: [1, 7, 8, 10, 15].

Варіант 30. Оптимізація пошуку у великих наборах даних

1) Створити клас SearchAlgorithms.
2) Реалізувати:
- binarySearch(), fibonacciSearch(), hashSearch().

Література: [1, 7, 8, 10, 15].

Варіант 31. Використання паралельних алгоритмів сортування

1) Створити клас ParallelSort.
2) Реалізувати:
- parallelQuickSort(), multiThreadedMergeSort().

Література: [1, 7, 8, 10, 15].

Варіант 32. Алгоритми для роботи з великими графами

1) Створити клас GraphAlgorithms.
2) Реалізувати:
- dijkstra(), bellmanFord(), prim(), kruskal().

Література: [1, 7, 8, 10, 15].

ЛІТЕРАТУРА

1. Страуструп Б. *Мова програмування C++*. Київ: Наукова думка, 2015. 752 с.
2. Майстренко В.І. *Об'єктно-орієнтоване програмування на C++: Навчальний посібник.* Київ: КНТ, 2013. 280 с.
3. Бублик В.В. *Алгоритми та структури даних.* Київ: Видавничий дім "Києво-Могилянська академія", 2017. 368 с.
4. Пилипчук В.В., Іванюк П.О. *Проектування алгоритмів та їх аналіз.* Львів: ЛНУ ім. І. Франка, 2019. 312 с.
5. Павлов О.М. *Структури даних та алгоритми на C++*. Харків: Фоліо, 2020. 356 с.
6. Ляшенко О.О. *Дискретна математика: навчальний посібник.* Львів: Видавництво ЛНУ ім. І. Франка, 2018. 294 с.
7. Глинський Я.М., Тарасов В.М. *Теорія графів у програмуванні.* Одеса: ОНПУ, 2016. 278 с.
8. Тарасенко О.М. *Чисельні методи та їх застосування в інформатиці.* Дніпро: НМетАУ, 2021. 288 с.
9. Stroustrup B. *The C++ Programming Language, 4th Edition.* Addison-Wesley, 2013. – 1376 p.
10. Meyers S. *Effective C++: 55 Specific Ways to Improve Your Programs and Designs.* Addison-Wesley, 2005. – 320 p.
11. Sutter H. *Exceptional C++: 47 Engineering Puzzles, Programming Problems, and Solutions.* Addison-Wesley, 1999. – 240 p.
12. Josuttis N.M. *The C++ Standard Library: A Tutorial and Reference, 2nd Edition.* Addison-Wesley, 2012. – 1128 p.
13. Alexandrescu A. *Modern C++ Design: Generic Programming and Design Patterns Applied.* Addison-Wesley, 2001. – 352 p.
14. Prata S. *C++ Primer Plus, 6th Edition.* Addison-Wesley, 2011. – 1080 p.
15. Sedgewick R. *Algorithms in C++, 3rd Edition.* Addison-Wesley, 2002. – 752 p.

Додаток А
Приклад оформлення титульної сторінки курсової роботи

ДЕРЖАВНА СЛУЖБА СТАТИСТИКИ УКРАЇНИ

НАЦІОНАЛЬНОЇ АКАДЕМІЇ СТАТИСТИКИ, ОБЛІКУ ТА АУДИТУ

КАФЕДРА СТАТИСТИКИ, ІТ ТА МАТЕМАТИЧНИХ МЕТОДІВ В ЕКОНОМІЦІ

КУРСОВА РОБОТА

з Об’єктно-орієнтованого програмування

на тему:

«_____
_____»

Студента (ки) _____ курсу _____ групи галузі знань F «Інформаційні технології»
спеціальності F3 «Комп'ютерні науки»; F6 «Інформаційні системи і технології»

_____
(прізвище та ініціали)

Керівник _____ст. вик., Савченко С.С.

Національна шкала _____

Кількість балів _____ Оцінка: ECTS _____

м. Київ 202_ р.

Додаток Б

Бланк завдання на курсовий проект (роботу)

_____

назва вищого навчального закладу

Кафедра _____

Дисципліна _____

Спеціальність _____

Курс_____ Група _____ Семестр _____

ЗАВДАННЯ

на курсовий проект (роботу) студента

_____

Прізвище, ім’я, по-батькові

1. Тема проекту (роботи)

_____

_____

_____

2. Строк здачі студентом закінченого проекту (роботи)

_____

_____

_____

3. Вихідні дані до проекту (роботи)

_____

_____

_____

4. Зміст роботи (перелік питань по розділах, які потрібно розробити)

_____

_____

_____

5. Перелік графічного матеріалу (з точним зазначенням обов’язкових креслень)

_____

_____

_____

6. Дата видачі завдання _____

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Назва етапів курсового проекту (роботи)	Строк виконання	Примітки

Студент _____ _____

Підпис Прізвище, ініціали

Керівник _____ _____

Підпис Прізвище, ініціали

«____» _____ 202___ р.

Додаток В

Приклад оформлення змісту курсової роботи

ЗМІСТ

Додаток Г

Приклад оформлення таблиць у курсовій роботі

Таблиця 2.1

Показники отримані підчас першого експерименту у несприятливих для GPS навігатора умовах

№	довгота	широта	№	довгота	широта
1	24.9218540	48.9218540	12	24.7003985	48.9218805
2	24.7003580	48.9218699	13	24.7003982	48.9218851
3	24.7003644	48.9218615	14	24.7003961	48.9218894
4	24.7003615	48.9218710	15	24.7003970	48.9218939
5	24.7003694	48.9218783	16	24.7004020	48.9218970
6	24.7003807	48.9218781	17	24.7004060	48.9219006

Додаток Д
Приклад оформлення рисунків у курсовій роботі

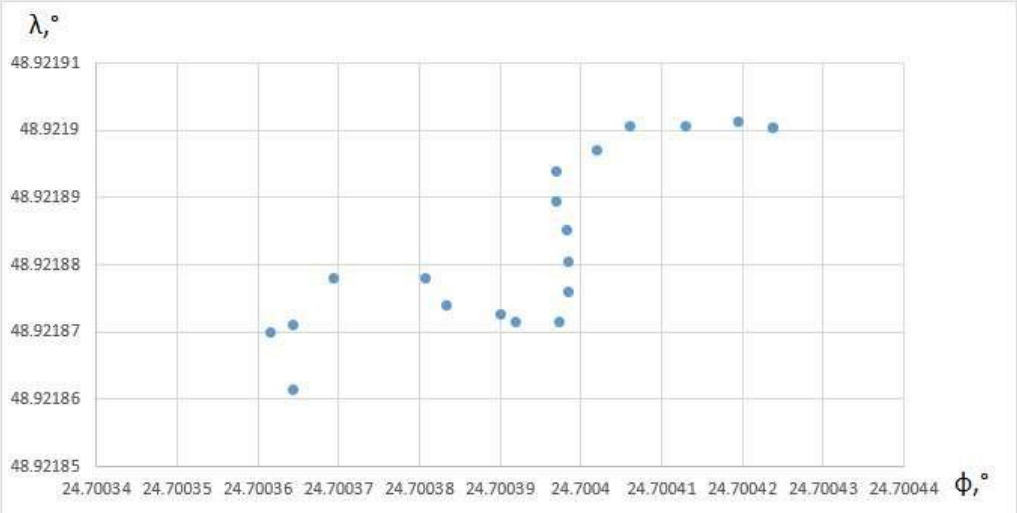

Рисунок 2.7. Діаграма множин значень отриманих від GPS модуля на одній позиції у несприятливих умовах

Додаток Е
Взірець подання формул у курсовій роботі

Коефіцієнти передавальної функції цифрового фільтру нижніх частот першого порядку повинні відповідати формулі 2.2.

$$b_0 = b_1 = (1+a)/2. \quad (2.2)$$

Щоб забезпечити адекватну роботу фільтра нижніх частот необхідно щоб значення коефіцієнта а знаходилось у межах |a| < 1.

Додаток Ж
Взірець оформлення списку використаних джерел

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Differential GPS [Електронний ресурс] – Режим доступу до ресурсу: https://www.trimble.com/gps_tutorial/dgps-how.aspx.

2. Дичківська О. О. Інноваційний менеджмент : конспект лекцій. Київ : ДІА, 2018. 82 с.

3. Мороз І. С., Василенко Н. Ю. Маркетинг : конспект лекцій. Київ : Молодь, 2016. 102 с.

4. Денисенко М. П., Догмачов В. М., Кабанов В. Г. Кредитування та ризики : навч. посіб. Київ, 2010. 213 с.

5. Вища математика : конспект лекцій / Ткачук Т.С. та ін. Київ, 2015. 82 с.

6. NET documentation. / Microsoft Learn. URL: https://learn.microsoft.com/en-us/dotnet/ (дата звернення: 13.02.2023).