Міністерство освіти і науки України
Національний аерокосмічний університет
«Харківський авіаційний інститут»

Кафедра інженерії програмного забезпечення (№ 603)

ЗАТВЕРДЖУЮ

Гарант освітньої програми
_____ І.В. Шевченко
(підпис) (ініціали та прізвище)

« 29 » серпня 2025 р.

СИЛАБУС ОБОВ'ЯЗКОВОЇ НАВЧАЛЬНОЇ ДИСЦИПЛІНИ

АРХІТЕКТУРА ТА ПРОЕКТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ .NET

(назва навчальної дисципліни)

Галузь знань: F Інформаційні технології
(шифр і найменування галузі знань)

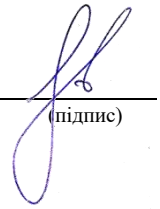
Спеціальність: F2 Інженерія програмного забезпечення
(код і найменування спеціальності)

Освітня програма: Інженерія програмного забезпечення
(найменування освітньої програми)

Рівень вищої освіти: перший (бакалаврський)

Силабус введено в дію з 01.09.2025 р.

Харків – 2025 р.

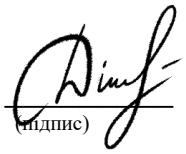
Розробник: Лучшев П.О., доцент каф.603, к.т.н., _____ 
(прізвище та ініціали, посада, науковий ступінь та вчене звання) (підпис)

Силабус навчальної дисципліни розглянуто на засіданні кафедри _____
інженерії програмного забезпечення (№ 603)
(назва кафедри)

Протокол №1 від «29» серпня 2025 р.

Завідувач кафедри д.т.н., професор  І.Б. Туркін
(науковий ступінь та вчене звання) (підпис) (ініціали та прізвище)

Погоджено з представником здобувачів освіти:

Представник студентського самоврядування  Д.В. Дикун
(підпис) (ініціали та прізвище)

1. Загальна інформація про викладача

ПІБ: Лучшев Павло Олександрович
Посада: доцент кафедри Інженерії програмного забезпечення (№603)
Науковий ступінь: кандидат технічних наук
Вчене звання: немає
Перелік дисциплін, які викладає: Комп’ютерна графіка з OpenGL; Програмування віртуальної реальності; Архітектура та проектування програмного забезпечення .Net
Напрями наукових досліджень: інженерія програмного забезпечення, ООП, комп’ютерна графіка
Контактна інформація: *e-mail*: p.luchshev@khai.edu

2. Опис навчальної дисципліни

Форма здобуття освіти	Денна
Семестр	4
Мова викладання	Українська
Тип дисципліни	Обов’язкова
Обсяг дисципліни: кредити ЄКТС/ кількість годин	4,5 кредитів ЄКТС / 135 годин (56 аудиторних, з яких: лекції – 24, практичні – 32; СРЗ – 79)
Види навчальної діяльності	Лекції, практичні заняття, розрахунково-графічна робота, самостійна робота
Види контролю	Поточний контроль, модульний контроль, семестровий контроль – іспит
Пререквізити	«Основи програмування», «Програмування мовою C#» «Алгоритми та структури даних», «Об'єктно-орієнтоване програмування»
Кореквізити	«Git-технологія командної розробки проєктів», «Реляційні бази даних»
Постреквізити	«Аналіз вимог до програмного забезпечення», «Менеджмент IT-проєктів»

3. Мета та завдання навчальної дисципліни, переліки компетентностей та очікуваних результатів навчання

Мета – ознайомлення та засвоєння студентами існуючих архітектур програмного забезпечення, принципами проектування розподілених систем з використанням проміжного/сполучного програмного забезпечення та шаблонів проектування.

Завдання – в результаті навчання студенти матимуть практичні навички з розроблення складних програмних проектів з використанням однієї або декількох добре відомих архітектур та компонентного підходу у середовищі Visual Studio.

Компетентності, які набуваються.

Інтегральна компетентність:

Здатність розв'язувати складні спеціалізовані завдання або практичні проблеми інженерії програмного забезпечення, що характеризуються комплексністю та невизначеністю умов, із застосуванням теорій та методів інформаційних технологій.

Загальні компетентності (ЗК)

Після закінчення цієї програми здобувач освіти буде здатен:

ЗК02. Здатність застосовувати знання у практичних ситуаціях.

ЗК05. Здатність вчитися і оволодівати сучасними знаннями.

ЗК06. Здатність до пошуку, оброблення та аналізу інформації з різних джерел.

ЗК13. Здатність ухвалювати рішення та діяти, дотримуючись принципу неприпустимості корупції та будь-яких інших проявів недоброчесності.

Спеціальні компетентності (СК або ФК)

Після закінчення цієї програми здобувач освіти буде здатен:

ФК02. Здатність брати участь у проектуванні програмного забезпечення, включаючи проведення моделювання (формальний опис) його структури, поведінки та процесів функціонування.

ФК03. Здатність розробляти архітектури, модулі та компоненти програмних систем.

ФК07. Володіння знаннями про інформаційні моделі даних, здатність створювати програмне забезпечення для зберігання, видобування та опрацювання даних.

ФК10. Здатність накопичувати, обробляти та систематизувати професійні знання щодо створення і супроводження програмного забезпечення та визнання важливості навчання протягом всього життя.

ФК11. Здатність реалізовувати фази та ітерації життєвого циклу програмних систем та інформаційних технологій на основі відповідних моделей і підходів розробки програмного забезпечення.

ФК13. Здатність обґрунтовано обирати та освоювати інструментарій з розробки та супроводження програмного забезпечення.

ФК14. Здатність до алгоритмічного та логічного мислення.

Програмні результати навчання (ПРН або РН):

ПРН01. Аналізувати, цілеспрямовано шукати і вибирати необхідні для вирішення професійних завдань інформаційно-довідникові ресурси і знання з урахуванням сучасних досягнень науки і техніки.

ПРН04. Знати і застосовувати професійні стандарти і інші нормативно-правові документи в галузі інженерії програмного забезпечення.

ПРН05. Знати і застосовувати відповідні математичні поняття, методи доменного, системного і об’єктно-орієнтованого аналізу та математичного моделювання для розробки програмного забезпечення.

ПРН06. Уміння вибирати та використовувати відповідну задачі методологію створення програмного забезпечення.

ПРН08. Вміти розробляти людино-машинний інтерфейс.

ПРН09. Знати та вміти використовувати методи та засоби збору, формулювання та аналізу вимог до програмного забезпечення.

ПРН10. Проводити передпроектне обстеження предметної області, системний аналіз об'єкта проектування.

ПРН15. Мотивовано обирати мови програмування та технології розробки для розв’язання завдань створення і супроводження програмного забезпечення.

ПРН17. Вміти застосовувати методи компонентної розробки програмного забезпечення.

ПРН26. Вміти конфігурувати компоненти, організувати робочий процес розгортання та тестувати рішення в його остаточному операційному середовищі.

4. Зміст навчальної дисципліни

МОДУЛЬ 1
Змістовний модуль №1. Технології розробки ПЗ

Тема 1. Об'єкт, предмет, метод і значення дисципліни. Ефективні способи вирішення задач проектування програмного забезпечення

Стисла анотація: Розгляд ключових понять архітектури програмного забезпечення, визначення її ролі у життєвому циклі розробки. Вивчення фундаментальних принципів проектування (SOLID, KISS, DRY), що є основою для створення гнучких, масштабованих та підтримуваних систем. Аналіз відмінностей між архітектурою та дизайном, обговорення ролі архітектора у команді. Огляд основних архітектурних стилів та їх застосування для вирішення типових бізнес-задач.

Тема лекції 1: Вступ до архітектури та проектування ПЗ. Ключові принципи та архітектурні стилі.

Тема практичного заняття: Практичне заняття з цієї теми не передбачено (семінар-обговорення).

Самостійна робота здобувача освіти: Закріплення розуміння основних принципів проектування (SOLID). Аналіз прикладів успішних та невдалих архітектурних рішень. Опрацювання лекційного матеріалу та підготовка питань до викладача.

Тема 2. Розробка та застосування DLL. Створення та налагодження програм декількома засобами розробки

Стисла анотація: Вивчення концепції модульного програмування на платформі .NET через створення та використання динамічно підключаємих бібліотек (DLL). Розгляд процесу розробки бібліотеки класів, її компіляції та підключення до основного проекту. Опанування технік налагодження багатопроектних рішень у середовищі Visual Studio. Практична реалізація рішення, що складається з головного застосунку та зовнішньої бібліотеки, для інкапсуляції та повторного використання коду.

Тема лекції 2: Модульне програмування в .NET. Розробка та використання бібліотек класів (DLL).

Тема практичного заняття: Створення та підключення власної DLL у багатопроектному рішенні Visual Studio.

Самостійна робота здобувача освіти: Закріплення навичок створення та використання бібліотек класів. Дослідження глобального кешу збірок (GAC). Опрацювання лекційного матеріалу та підготовка питань до викладача. Оформлення звіту з практичної роботи.

Тема 3. Компоненти. Розробка архітектури складного програмного забезпечення

Стисла анотація: Розгляд компонентного підходу як основи для побудови складних систем. Вивчення принципів багатошарової архітектури (N-Tier Architecture), зокрема, класичного поділу на рівні представлення (UI), бізнес-логіки (BLL) та доступу до даних (DAL). Аналіз переваг слабкої зв'язності та

високої згуртованості компонентів. Проектування архітектури для навчального проекту, визначення відповідальності кожного шару та інтерфейсів взаємодії між ними у середовищі Visual Studio.

Тема лекції 3: Компонентний підхід та проектування багатошарової архітектури ПЗ.

Тема практичного заняття: Проектування структури багатошарового застосунку (UI, BLL, DAL).

Самостійна робота здобувача освіти: Закріплення знань про принципи та переваги багатошарової архітектури. Дослідження поняття "Інверсія залежностей" (Dependency Injection) як механізму для забезпечення слабкої зв'язності між шарами. Розробка детальної схеми архітектури для власного навчального проекту. Опрацювання лекційного матеріалу та підготовка питань до викладача. Оформлення звіту з практичної роботи.

Тема 4. Створення, завершення і безпосереднє управління потоками. Налагодження багатопоточних програм

Стисла анотація: Вивчення основ багатопотокового програмування в .NET. Розгляд класу System.Threading.Thread для створення та управління потоками виконання. Аналіз життєвого циклу потоку: запуск, призупинення, переривання та завершення. Опанування інструментів Visual Studio для налагодження багатопоточних програм (вікна "Threads", "Parallel Stacks"). Моделювання паралельного виконання задач та аналіз впливу на продуктивність застосунку.

Тема лекції 4: Основи багатопоточності в .NET: клас Thread та його життєвий цикл.

Тема практичного заняття: Створення та управління потоками. Налагодження багатопоточних застосунків.

Самостійна робота здобувача освіти: Закріплення знань про життєвий цикл потоків. Вивчення передачі даних у потоки. Самостійне написання програм для паралельних обчислень. Опрацювання лекційного матеріалу та підготовка питань до викладача. Оформлення звіту з практичної роботи.

Тема 5. Об’єкти та методи синхронізації. Проблеми спільного доступу до даних

Стисла анотація: Аналіз типових проблем, що виникають при паралельному доступі до спільних ресурсів: стан гонитви (race condition), взаємні блокування (deadlock), інверсія пріоритетів. Розгляд критичної секції як ділянки коду, що вимагає ексклюзивного доступу. Теоретичний огляд механізмів синхронізації, що запобігають конфліктам потоків. Моделювання проблеми "читачі-письменники" для демонстрації наслідків несинхронізованого доступу до даних.

Тема лекції 5: Проблеми синхронізації потоків: стан гонитви та взаємні блокування.

Тема практичного заняття: Моделювання проблеми спільного доступу до даних та аналіз наслідків.

Самостійна робота здобувача освіти: Закріплення теоретичних знань про проблеми паралельного програмування. Детальне вивчення умов виникнення

взаємних блокувань (умови Коффмана). Пошук та аналіз прикладів програмного коду, що містить потенційні помилки синхронізації. Опрацювання лекційного матеріалу та підготовка питань до викладача. Оформлення звіту з практичної роботи.

Тема 6. Використання об'єктів синхронізації: Event, Criticalsection, Semaphore, Mutex.

Стисла анотація: Детальне вивчення примітивів синхронізації у .NET. Розгляд оператора lock (клас Monitor) для організації критичних секцій. Аналіз об'єктів Mutex для міжпроцесної синхронізації та Semaphore для обмеження доступу до пулу ресурсів. Дослідження сигнальних механізмів AutoResetEvent та ManualResetEvent для координації дій між потоками. Практична реалізація класичної задачі "виробник-споживач" із застосуванням вивчених об'єктів синхронізації для забезпечення потокобезпечної взаємодії.

Тема лекції 6: Механізми синхронізації в .NET: Mutex, Semaphore, Event, Monitor (lock).

Тема практичного заняття: Застосування об'єктів синхронізації для вирішення проблеми "виробник-споживач".

Самостійна робота здобувача освіти: Закріплення розуміння відмінностей між Mutex та lock, Semaphore та SemaphoreSlim. Самостійне вирішення задач на синхронізацію. Опрацювання лекційного матеріалу та підготовка питань до викладача. Оформлення звіту з практичної роботи.

Змістовний модуль №2. Структура та архітектура ПЗ

Тема 7. Поняття шаблона проектування. Типи шаблонів GoF

Стисла анотація: Введення у концепцію шаблонів проектування як формалізованих, перевірених рішень типових проблем в об'єктно-орієнтованому дизайні. Обговорення переваг використання шаблонів: підвищення гнучкості, повторне використання коду та стандартизація комунікації між розробниками. Огляд класифікації шаблонів GoF: породжуючі (creational), структурні (structural) та поведінкові (behavioral). Аналіз структури опису шаблона: назва, проблема, рішення, наслідки.

Тема лекції 7: Вступ до шаблонів проектування. Класифікація GoF.

Тема практичного заняття: Практичне заняття з цієї теми об'єднано з наступними.

Самостійна робота здобувача освіти: Вивчення каталогу шаблонів GoF. Аналіз прикладів коду з реального програмного забезпечення на предмет використання шаблонів. Опрацювання лекційного матеріалу та підготовка питань до викладача.

Тема 8. Породжуючи шаблони

Стисла анотація: Розгляд шаблонів, що абстрагують процес створення об'єктів. Вивчення "Фабричного методу" для делегування створення екземплярів підкласам та "Абстрактної фабрики" для створення сімейств пов'язаних об'єктів. Аналіз шаблона "Одинак" (Singleton) для гарантії єдиного екземпляра класу. Практична реалізація шаблонів "Фабричний метод" та "Одинак" мовою C#.

Моделювання ситуацій, де застосування цих шаблонів дозволяє підвищити гнучкість та розширюваність системи, роблячи її незалежною від конкретних класів об'єктів.

Тема лекції 8: Породжуючі шаблони проектування: Singleton, Factory Method, Abstract Factory.

Тема практичного заняття: Реалізація шаблонів Factory Method та Singleton в C#.

Самостійна робота здобувача освіти: Закріплення розуміння породжуючих шаблонів. Самостійна реалізація шаблонів "Будівник" та "Прототип". Опрацювання лекційного матеріалу та підготовка питань до викладача. Оформлення звіту з практичної роботи.

Тема 9. Структурні шаблони

Стисла анотація: Вивчення шаблонів, що визначають способи композиції класів та об'єктів для утворення більших структур. Розгляд "Адаптера" для узгодження несумісних інтерфейсів, "Декоратора" для динамічного додавання функціональності та "Фасаду" для спрощення взаємодії зі складною підсистемою. Реалізація шаблонів "Адаптер" та "Декоратор" у C# на практичних прикладах. Розробка коду, що демонструє, як ці шаблони дозволяють модифікувати та розширювати систему без зміни існуючого коду.

Тема лекції 9: Структурні шаблони проектування: Adapter, Decorator, Facade.

Тема практичного заняття: Реалізація шаблонів Adapter та Decorator для розширення функціональності класів.

Самостійна робота здобувача освіти: Закріплення розуміння шаблонів "Адаптер", "Декоратор" та "Фасад". Самостійне вивчення шаблонів "Замісник" (Proxy) та "Компонувальник" (Composite), аналіз їх призначення та структури. Порівняння сценаріїв застосування та відмінностей між шаблонами "Адаптер", "Декоратор" та "Замісник". Опрацювання лекційного матеріалу та підготовка питань до викладача. Оформлення звіту з практичної роботи.

Тема 10. Поведінкові шаблони

Стисла анотація: Огляд шаблонів, що визначають алгоритми та розподіл відповідальності між об'єктами. Вивчення "Стратегії" для інкапсуляції сімейства алгоритмів, "Спостерігача" для реалізації механізму підписки-сповіщення та "Команди" для інкапсуляції запиту як об'єкта. Практична реалізація шаблонів "Стратегія" та "Спостерігач" у C#. Розробка гнучких систем, де поведінку об'єктів можна змінювати під час виконання, та створення механізмів слабкозв'язаної взаємодії між компонентами.

Тема лекції 10: Поведінкові шаблони проектування: Strategy, Observer, Command.

Тема практичного заняття: Реалізація шаблонів Strategy та Observer для створення гнучкої системи.

Самостійна робота здобувача освіти: Закріплення знань про поведінкові шаблони. Самостійне вивчення та реалізація шаблонів "Шаблонний метод" та

"Ланцюг обов'язків". Опрацювання лекційного матеріалу та підготовка питань до викладача. Оформлення звіту з практичної роботи.

Тема 11. Шаблони багатозадачності

Стисла анотація: Розгляд спеціалізованих шаблонів для вирішення проблем у багатопотокових середовищах. Аналіз шаблона "Блокування з подвійною перевіркою" (Double-Checked Locking) для ефективної потокобезпечної "лінивої" ініціалізації. Вивчення шаблона "Монітор" (Monitor Object) для гарантії взаємного виключення при доступі до об'єкта. Практичне застосування шаблона Double-Checked Locking для реалізації потокобезпечного "Одинака". Аналіз коду та виявлення переваг цього підходу порівняно з простою синхронізацією.

Тема лекції 11: Шаблони для багатопоточного програмування: Monitor, Double-checked locking.

Тема практичного заняття: Реалізація потоко-безпечного Singleton за допомогою Double-checked locking.

Самостійна робота здобувача освіти: Дослідження шаблона "Виробник-споживач" (Producer-Consumer). Вивчення класу ThreadPool як реалізації шаблона "Пул потоків". Опрацювання лекційного матеріалу та підготовка питань до викладача. Оформлення звіту з практичної роботи.

Змістовний модуль №3. COM-технології

Тема 12. Основи поняття COM-технології. Інтерфейси. Створення і використання ідентифікаторів об'єктів. Засоби реєстрації та обліку об'єктів

Стисла анотація: Теоретичний огляд технології Component Object Model (COM) як бінарного стандарту для взаємодії програмних компонентів. Вивчення фундаментальних концепцій: інтерфейс IUnknown, механізм підрахунку посилань, глобально унікальні ідентифікатори (GUID/CLSID/IID) та повернення результатів через HRESULT. Розгляд процесу реєстрації COM-компонентів у системному реєстрі Windows. Аналіз існуючих COM-інтерфейсів за допомогою системних утиліт (напр., OLE/COM Object Viewer) для розуміння їхньої структури та методів.

Тема лекції 12: Основи COM: інтерфейси, GUID, підрахунок посилань та реєстрація.

Тема практичного заняття: Аналіз COM-інтерфейсів за допомогою системних утиліт.

Самостійна робота здобувача освіти: Закріплення розуміння ролі IUnknown та підрахунку посилань. Дослідження реєстру Windows для пошуку зареєстрованих COM-об'єктів. Опрацювання лекційного матеріалу та підготовка питань до викладача. Оформлення звіту з практичної роботи.

Тема 13. Створення та використання COM-об’єктів

Стисла анотація: Розгляд принципів взаємодії з існуючими COM-об'єктами з сучасних середовищ розробки. Вивчення механізмів пізнього та раннього зв'язування для доступу до функціональності COM-компонентів. Практична демонстрація використання зовнішніх COM-об'єктів на прикладі

автоматизації Microsoft Office (напр., створення та маніпуляція документом Excel або Word) з C# застосунку.

Тема лекції 13: Принципи взаємодії з COM-об'єктами. Раннє та пізнє зв'язування.

Тема практичного заняття: Використання існуючих COM-об'єктів (напр., Microsoft Excel) з C#.

Самостійна робота здобувача освіти: Закріплення знань про механізми раннього та пізнього зв'язування, аналіз їх переваг та недоліків. Дослідження об'єктної моделі іншого COM-сервера (наприклад, Microsoft Word). Вивчення правил коректного звільнення ресурсів при роботі з COM-об'єктами в .NET. Опрацювання лекційного матеріалу та підготовка питань до викладача. Оформлення звіту з практичної роботи.

Тема 14. Застосування COM-об’єктів на платформі .Net

Стисла анотація: Вивчення технології COM Interop, що забезпечує взаємодію між керованим кодом .NET та некерованим кодом COM. Розгляд об'єктів-обгорток: Runtime Callable Wrapper (RCW) для використання COM-компонентів у .NET та COM Callable Wrapper (CCW) для надання .NET-компонентів COM-клієнтам. Практична розробка простого .NET-компонента, його експорт для видимості з COM-середовища за допомогою атрибутів та системних утиліт. Тестування доступу до створеного .NET-компонента з некерованого середовища (напр., VBScript).

Тема лекції 14: Технологія COM Interop в .NET: RCW та CCW.

Тема практичного заняття: Створення .NET-компонента, видимого для COM.

Самостійна робота здобувача освіти: Дослідження атрибута [ComVisible] та утиліти regasm.exe. Вивчення процесу створення бібліотек типів (TLB). Опрацювання лекційного матеріалу та підготовка питань до викладача. Оформлення звіту з практичної роботи.

Змістовний модуль №4. Аналіз якості та оцінка програмного дизайну

Тема 15. Методи та засоби вимірювання внутрішніх параметрів і складності програмного забезпечення

Стисла анотація: Введення в програмну метрику як інструмент кількісної оцінки якості коду. Розгляд базових метрик: кількість рядків коду (LOC), цикломатична складність за МакКейбом для оцінки складності логіки, метрики Холстеда для оцінки об'єму та зусиль на розробку. Практичне застосування вбудованих інструментів статичного аналізу коду в Visual Studio. Розрахунок та інтерпретація метрик для існуючого коду, виявлення надто складних методів та модулів, що потребують рефакторингу.

Тема лекції 15: Вступ до метрик ПЗ. Цикломатична складність та метрики Холстеда.

Тема практичного заняття: Використання інструментів Visual Studio для аналізу коду та розрахунку метрик.

Самостійна робота здобувача освіти: Закріплення розуміння цикломатичної складності. Самостійний аналіз власного проекту за допомогою інструментів статичного аналізу. Опрацювання лекційного матеріалу та підготовка питань до викладача. Оформлення звіту з практичної роботи.

Тема 16. Попередня оцінка складності проекту

Стисла анотація: Огляд методів для прогнозування трудовитрат та складності програмних проектів. Вивчення базових принципів алгоритмічних моделей оцінки, таких як COCOMO (Constructive Cost Model). Розгляд концепції функціональних точок (Function Points) як методу оцінки обсягу проекту на основі його функціональності. Демонстрація розрахунку приблизної складності та трудовитрат для навчального проекту за спрощеною моделлю. Обговорення факторів, що впливають на точність оцінки, та обмежень існуючих моделей.

Тема лекції 16: Моделі оцінки складності програмних проектів (COCOMO, Function Points).

Тема практичного заняття: Розрахунок приблизної складності для навчального проекту.

Самостійна робота здобувача освіти: Закріплення розуміння моделей COCOMO та методу функціональних точок. Детальне вивчення факторів-множників (cost drivers) моделі COCOMO та їх впливу на кінцеву оцінку. Огляд альтернативних методів оцінки проектів, зокрема, експертних та agile-методик (наприклад, Planning Poker). Опрацювання лекційного матеріалу та підготовка питань до викладача. Оформлення звіту з практичної роботи.

Тема 17. Метрики об’єктно-орієнтованого проектування програмного забезпечення

Стисла анотація: Вивчення спеціалізованих метрик для оцінки якості об'єктно-орієнтованого дизайну, зокрема набору метрик Чідамбера та Кемерера (CK metrics). Аналіз метрик зв'язності (CBO - Coupling Between Objects) та згуртованості (LCOM - Lack of Cohesion in Methods) та їх впливу на якість архітектури. Практичний аналіз ООП-дизайну існуючого проекту. Розрахунок метрик CBO та LCOM, інтерпретація результатів для виявлення проблемних місць у дизайні (надмірна зв'язність класів, низька згуртованість методів).

Тема лекції: Метрики об'єктно-орієнтованого дизайну (метрики Чідамбера та Кемерера).

Тема практичного заняття: Аналіз ООП-дизайну за допомогою метрик зв'язності та згуртованості.

Самостійна робота здобувача освіти: Закріплення розуміння метрик зв'язності (CBO) та згуртованості (LCOM). Самостійне вивчення інших метрик з набору Чідамбера та Кемерера: DIT (глибина дерева наслідування), NOC (кількість нащадків) та RFC (Response for a Class). Аналіз того, як екстремальні значення цих метрик можуть вказувати на потенційні проблеми в ООП-дизайні. Опрацювання лекційного матеріалу та підготовка питань до викладача. Оформлення звіту з практичної роботи.

Тема 18. Еволюція та реінжиніринг програмного забезпечення

Стисла анотація: Розгляд життєвого циклу програмного забезпечення після його випуску: супроводження, еволюція та модернізація. Вивчення поняття рефакторингу як процесу покращення внутрішньої структури коду без зміни його зовнішньої поведінки. Аналіз індикаторів проблем у дизайні. Вивчення відмінностей між реінжинірингом (перепроектуванням) та реверс-інжинірингом. Практичне застосування технік рефакторингу в навчальному проекті з метою підвищення його читабельності, гнучкості та підтримуваності.

Тема лекції: Супроводження, еволюція та реінжиніринг ПЗ. Основи рефакторингу.

Тема практичного заняття: Рефакторинг програмного коду.

Самостійна робота здобувача освіти: Вивчення технік рефакторингу за Мартіном Фаулером. Опрацювання лекційного матеріалу та підготовка питань до викладача. Оформлення звіту з практичної роботи.

Модульний контроль

Форма занять: тестування в аудиторії (за рішенням лектора допускається проведення у дистанційній формі).

Самостійна робота здобувача освіти:

Підготовка до модульного контролю 1:

– опрацювання теоретичного матеріалу за темами змістовних модулів;

– перегляд та аналіз звітів з виконаних практичних робіт з урахуванням рекомендацій та зауважень викладача;

– підготовка відповідей на можливі контрольні запитання, підготовка питань до викладача для уточнення складних моментів;

– опрацювання додаткових джерел для поглиблення знань з дисципліни.

5. Індивідуальні завдання

Виконання розрахунково-графічної роботи за темою «Патерни проектування».

Розрахункова робота включає:

1. Розгляд та створення програмних шаблонів проектування чотирьох типів: породжуючого, структурного, поведінкового, та багатозадачності.

2. Опис шаблонів проєктування за допомогою UML та розміщення їх програмної реалізації на GitHub.

Самостійна робота здобувача освіти:

Вибір та аналіз чотирьох шаблонів проектування (по одному з кожної категорії); створення UML-діаграм та програмна реалізація обраних шаблонів на C#; розміщення коду у власному репозиторії на GitHub; написання та оформлення пояснювальної записки з описом, діаграмами та лістингами; підготовка до захисту роботи та відповідей на запитання; оформлення пояснювальної записки та підготовка до здачі.

6. Методи навчання

Словесні, наочні, практичні.

7. Методи контролю

Поточний контроль (теоретичне опитування й виконання практичних робіт), модульний контроль (тестування за розділами курсу) та підсумковий (семестровий) контроль (іспит у виді загального тесту за усіма розділами курсу).

8. Критерії оцінювання та розподіл балів, які отримують здобувачі освіти

Таблиця 8.1 – Розподіл балів, які отримують здобувачі освіти

Складові навчальної роботи	Бали за одне заняття (завдання)	Кількість занять (завдань)	Сумарна кількість балів
Змістовний модуль 1,2			
Робота на лекціях	0…1	8	0…8
Виконання і захист практичних робіт	1...5	3	3...15
Модульний контроль	0…20	1	0…20
Змістовний модуль 3,4			
Робота на лекціях	0…1	8	0…8
Виконання і захист практичних робіт	1...5	3	3...15
Модульний контроль	0…20	1	0…20
Виконання і захист РГР (РР, РК)			1…20
Усього за семестр			**60…100**

Допуском до семестрового контролю є отримання позитивної оцінки з щонайменше з чотирьох практичних занять і розрахункової роботи.

Додатково може бути оцінено як дострокове виконання кожної з практичних робіт (до 2 балів), так і елементи самостійної роботи в ній (до 3 балів)). Сумарно основна та додаткова оцінка за усі види робіт не може перевищувати 100 балів.

Семестровий контроль (іспит) проводиться у вигляді комп’ютерного тестування (питання відкритого типу) у разі відмови студента від балів поточного тестування та за наявністю допуску до іспиту (виконання усіх практичних робіт та РГР). Під час складання семестрового іспиту студент має можливість отримати максимум 60 балів, які замінюють результати поточного модульного контролю.

Таблиця 8.2 – Шкали оцінювання: бальна і традиційна

Сума балів	Оцінка за традиційною шкалою	
	Іспит, диференційний залік	Залік
90 – 100	Відмінно	Зараховано
75 – 89	Добре	
60 – 74	Задовільно	
0 – 59	Незадовільно	Не зараховано

Критерії оцінювання роботи здобувача протягом семестру

Задовільно (60-74). Здати всі практичні роботи (зі значною корекцією тексту практичної роботи викладачем), захистити розрахункову роботу (зі значною корекцією тексту розрахункової роботи викладачем), мати необхідних мінімум знань за всіма темами та мінімум вмінь щодо застосування отриманих знань.

Добре (75-89). Здати всі практичні роботи (з мінімальною корекцією тексту практичної роботи викладачем), захистити розрахункову роботу (з мінімальною корекцією тексту розрахункової роботи викладачем), знати всі теми та уміти застосовувати їх.

Відмінно (90-100). Здати всі практичні роботи (без корекції тексту практичної роботи викладачем), захистити розрахункову роботу (без корекції тексту розрахункової роботи викладачем), досконально знати всі теми та уміти застосовувати їх.

9. Політика навчального курсу

Відпрацювання пропущених занять відбувається відповідно до розкладу консультацій, за попереднім погодженням з викладачем. Питання, що стосуються академічної доброчесності, розглядає викладач або за процедурою, визначеною у Положенні про академічну доброчесність.

Дотримання вимог академічної доброчесності здобувачами освіти під час вивчення навчальної дисципліни. Під час вивчення навчальної дисципліни здобувачі освіти мають дотримуватися загальноприйнятих морально-етичних норм і правил поведінки, вимог академічної доброчесності, передбачених Положенням про академічну доброчесність Національного аерокосмічного університету «Харківський авіаційний інститут» (https://khai.edu/assets/files/polozhennya/polozhennya-pro-akademichnu-dobrochesnist.pdf). Очікується, що роботи здобувачів освіти будуть їх оригінальними дослідженнями або міркуваннями. Відсутність посилань на використані джерела, фабрикування джерел, списування, втручання в роботу інших здобувачів освіти становлять, але не обмежують, приклади можливої академічної недоброчестності. Виявлення ознак академічної недоброчесності в письмовій роботі здобувача освіти є підставою для її незарахування викладачем незалежно від масштабів плагіату чи обману.

Вирішення конфліктів. Порядок і процедури врегулювання конфліктів, пов’язаних із корупційними діями, зіткненням інтересів, різними формами

дискримінації, сексуальними домаганнями, міжособистісними стосунками та іншими ситуаціями, що можуть виникнути під час навчання, а також правила етичної поведінки регламентуються Кодексом етичної поведінки в Національному аерокосмічному університеті «Харківський авіаційний інститут» (https://khai.edu/ua/university/normativna-baza/ustanovchi-dokumenti/kodeks-etichnoi-povedinki/).

10. Методичне забезпечення

Для забезпечення дистанційного доступу до навчально-методичний комплекс дисципліни використовується електронний ресурс,: https://mentor.khai.edu/course/view.php?id=6968 Для реєстрації студентів у системах дистанційного доступу використовуються почтові адреси виключно у домені ***student.khai.edu.***

Реєстрація на http://github.com виконується студентом самостійно. Додаткові навчальні посібники, навчально-методичні посібники, конспекти лекцій, методичні рекомендації з проведення практичних робіт тощо, які видані в Університеті знаходяться за посиланням: https://library.khai.edu/catalog

11. Рекомендована література

Базова

1. Табунщик Г. В. Проектування та моделювання програмного забезпечення сучасних інформаційних систем / Г. В. Табунщик, Т.І. Каплієнко, О.А. Петрова − Запоріжжя : Дике Поле, 2020. – 250 с. http://eir.zntu.edu.ua/bitstream/123456789/1824/1/Tabunshchik_Software_Design.pdf
2. Designing Applications on the .NET Platform. Application Architecture Guide 2.0 patterns & practices / J.D. Meier, Alex Homer, David Hill, Jason Taylor, Prashant Bansode, Lonnie Wall, Rob Boucher Jr, Akshay Bogawat – Microsoft press, – 381 p.
3. Опорний конспект лекцій. Архітектура та проектування програмного забезпечення.
4. «Дизайн-патерни — просто, як двері» by Andriy Buday is licensed under a Creative Commons Attribution-NonCommercial 3.0 Unported License http://designpatterns.andriybuday.com. 2022. – 90 с.

Допоміжна

1. Мартін Роберт Чиста архітектура: мистецтво розробки програмного забезпечення / Роберт Мартін – К.: Фабула, 2020. – 368 с. https://fabulabook.com/product/chysta-arhitektura/
2. Design Patterns: Elements of Reusable Object-Oriented Software. Erich Gamma, Richard Helm, Ralph Johnsonand John Vliss. Addison-Wesley Professional, 1994.
3. Мартін Роберт Чистий код: створення, аналіз, рефакторинг / Роберт Мартін – К.: Фабула, 2019. – 416 с. https://fabulabook.com/product/chystyj-kod/

4. E. Gamma, R. Helm, R. Johnson, J. Willsides. Tricks of object-oriented design. Design Patterns. Peter 2017, - 366 pp.
5. Martin Fowler. Templates for corporate applications. Williams, 2019, - 544 pp.

12. Інформаційні ресурси

1. ASP.NET MVC: TheOfficial Microsoft ASP.NET Site http://www.asp.net/mvc
2. http://www.aivosto.com/project/help/pm-oo-mood.html MOOD & MOOD2 metrics.
3. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.48.2017&rep=rep1&type=ps Fernando Brito e Abreu. Toward the Design Quality Evaluation of Object-Oriented / Software Systems. Fernando Brito e Abreu, Miguel Goulão, Rita Esteves/