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

Кафедра цифрових технологій в енергетиці

ОБ'ЄКТНО-ОРІЄНТОВАНЕ ПРОГРАМУВАННЯ

Робоча програма навчальної дисципліни (Силабус)

Реквізити навчальної дисципліни

Рівень вищої освіти	***Перший (бакалаврський)***
Галузь знань	*F Інформаційні технології*
Спеціальність	*F3 Комп'ютерні науки*
Освітня програма	*Цифрові технології в енергетиці*
Статус дисципліни	*Обов'язкова*
Форма навчання	*Очна (денна)*
Рік підготовки, семестр	*2 курс, 1 семестр*
Обсяг дисципліни	*На засвоєння дисципліни передбачено 150 год / 5 кредитів ЄКТС: 30 год лекцій, 28 год практичних занять, 28 год лабораторних робіт, 64 год самостійної роботи.*
Семестровий контроль/ контрольні заходи	*Модульна контрольна робота, екзамен*
Розклад занять	*http://rozklad.kpi.ua/*
Мова викладання	*Українська*
Інформація про керівника курсу / викладачів	Лектор: асистент, доктор філософії, Здор Костянтин Андрійович, email: Kostya9919moonlight@gmail.com Практичні та лабораторні заняття: асистент, доктор філософії, Здор Костянтин Андрійович, email: Kostya9919moonlight@gmail.com, асистент, доктор філософії, Мельниченко Артем Васильович, email: artemxl@gmail.com
Розміщення курсу	https://campus.kpi.ua/ https://classroom.github.com/

Програма навчальної дисципліни

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

Предметом вивчення дисципліни є об'єктно-орієнтована парадигма програмування та засоби її реалізації в мові C# на платформі .NET.

Метою вивчення дисципліни є формування у студентів здатностей:

- опанувати об'єктно-орієнтований підхід для розробки та розвитку програмних систем засобами мови C# та платформи .NET;
- розуміти методологію об'єктно-орієнтованого проектування, оволодіти нею і використовувати її впродовж життєвого циклу програмного забезпечення;
- розробляти програмне забезпечення за допомогою сучасних інструментальних засобів створення програмного забезпечення, систем контролю версій та засобів автоматизованого тестування.

Основні завдання при вивченні дисципліни

Дисципліна «Об'єктно-орієнтоване програмування» забезпечує наступні програмні компетентності і програмні результати освітньо-професійної програми (ОПП):

ЗК 2	Здатність застосовувати знання у практичних ситуаціях
ЗК 8	Здатність генерувати нові ідеї (креативність)
ЗК 11	Здатність приймати обґрунтовані рішення
ФК 3	Здатність до логічного мислення, побудови логічних висновків, використання формальних мов і моделей алгоритмічних обчислень, проектування, розроблення й аналізу алгоритмів, оцінювання їх ефективності та складності, розв’язності та нерозв’язності алгоритмічних проблем для адекватного моделювання предметних областей і створення програмних та інформаційних систем
ФК 8	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

В результаті засвоєння кредитного модуля студенти мають продемонструвати такі програмні результати навчання:

ПР 5	Проектувати, розробляти та аналізувати алгоритми розв’язання обчислювальних та логічних задач, оцінювати ефективність та складність алгоритмів на основі застосування формальних моделей алгоритмів та обчислюваних функцій.
ПР 9	Розробляти програмні моделі предметних середовищ, вибирати парадигму програмування з позицій зручності та якості застосування для реалізації методів та алгоритмів розв’язання задач в галузі комп’ютерних наук.

знання:

- синтаксис і семантика об'єктно-орієнтованої мови програмування C#, будова та можливості платформи .NET;
- патернів об'єктно-орієнтованого проектування та принципів SOLID;
- знання інструментальних засобів, інтегрованих середовищ, систем контролю версій та засобів модульного тестування для створення програмного забезпечення

уміння:

- аналізувати вимоги до програмного забезпечення;
- проектувати архітектуру програмного забезпечення на основі класів, інтерфейсів та узагальнених типів;
- використовувати патерни об'єктно-орієнтованого проектування;
- поєднувати об'єктно-орієнтований підхід з узагальненим та функціональним (generics, LINQ);
- розробляти асинхронний програмний код та коректно обробляти виняткові ситуації;
- виконувати рефакторинг програмного коду та покривати його модульними тестами;
- використовувати потрібні інструментальні засоби та інтегровані середовища для вирішення завдань;

досвід:

- розроблення програмного забезпечення мовою C# на платформі .NET;
- роботи з комп'ютерними інструментальними засобами, інтегрованими середовищами та системою контролю версій Git.

2. Пререквізити та постреквізити дисципліни

У структурно-логічній схемі навчання зазначений кредитний модуль розміщується тоді, коли студенти вже прослухали такі дисципліни, як "Програмування алгоритмічних структур", "Проектування та аналіз обчислювальних алгоритмів" та "Алгоритмізація та програмування", і є

логічним продовженням навчального процесу набуття знань та навичок сучасних систем програмування. Попереднє знайомство з мовою C# не є обов'язковим: базовий синтаксис мови опрацьовується на першій лекції та першому практичному занятті з опорою на алгоритмічну підготовку студентів.

З іншого боку, викладений матеріал може бути використаний при вивченні дисциплін “Геометричне моделювання та комп’ютерна графіка”, “Основи системного аналізу ”, “Технології розробки програмного забезпечення”, які подаються в наступних семестрах.

3. Зміст навчальної дисципліни

В дисципліні вивчаються такі теми:

Розділ 1. Платформа .NET та базові конструкції ООП

Тема 1.1. Парадигми програмування. Платформа .NET. Типи значень і посилальні типи.

Тема 1.2. Класи і об'єкти. Керування пам'яттю та життєвий цикл об'єктів.

Тема 1.3. Перевантаження методів та операцій.

Тема 1.4. Успадкування. Ієрархія класів. Абстрактні класи.

Розділ 2. Абстракції, контракти та узагальнення

Тема 2.1. Поліморфізм. Інтерфейси.

Тема 2.2. Принципи SOLID. Впровадження залежностей.

Тема 2.3. Узагальнене програмування. Колекції та ітератори.

Розділ 3. Надійність, обробка даних та сучасні засоби C#

Тема 3.1. Обробка виняткових ситуацій. Рядки.

Тема 3.2. Файлові потоки та серіалізація.

Тема 3.3. Делегати, події та мова запитів LINQ.

Тема 3.4. Асинхронне програмування.

Тема 3.5. Патерни проектування. Якість програмного коду.

4. Навчальні матеріали та ресурси

Базова література:

1. Albahari J. C# in a Nutshell: The Definitive Reference / J. Albahari. O'Reilly Media (актуальне видання під поточну версію мови). Режим доступу: https://www.albahari.com/nutshell/
2. Price M. J. C# and .NET – Modern Cross-Platform Development Fundamentals / M. J. Price. Packt Publishing (щорічні перевидання).
3. Troelsen A., Japikse P. Pro C# with .NET / A. Troelsen, P. Japikse. Apress, 2022, 1300 p.
4. Freeman E., Robson E. Head First Design Patterns. 2nd edition / E. Freeman, E. Robson. O'Reilly Media, 2020, 672 p.
5. Gamma E., Helm R., Johnson R., Vlissides J. Design Patterns: Elements of Reusable Object-Oriented Software. Addison-Wesley, 1994, 395 p.
6. Алхімова С. М. Об'єктно-орієнтоване програмування / С. М. Алхімова. Київ: КПІ ім. Ігоря Сікорського, 2019, 190 с.
7. Документація з мови C# та платформи .NET [Електронний ресурс]. Microsoft Learn. Режим доступу: https://learn.microsoft.com/dotnet/csharp/

Додаткова:

1. Skeet J. C# in Depth. 4th edition / J. Skeet. Manning Publications, 2019, 528 p.
2. Richter J. CLR via C#. 4th edition / J. Richter. Microsoft Press, 2012, 896 p.
3. Nagel C. Professional C# and .NET / C. Nagel. Wrox, 2021, 880 p.
4. Martin R. C. Clean Code: A Handbook of Agile Software Craftsmanship / R. C. Martin. Prentice Hall, 2008, 464 p.
5. Seemann M., van Deursen S. Dependency Injection Principles, Practices, and Patterns / M. Seemann, S. van Deursen. Manning Publications, 2019, 552 p.
6. Fowler M. Refactoring: Improving the Design of Existing Code. 2nd edition / M. Fowler. Addison-Wesley, 2018, 448 p.

Навчальний контент

5. Методика опанування навчальної дисципліни (освітнього компонента)

№ з/п	Назва теми лекції та перелік основних питань (перелік дидактичних засобів, посилання на літературу та завдання на СРС)	К-сть ауд.год.
Розділ 1. Платформа .NET та базові конструкції ООП		
Тема 1.1. Парадигми програмування. Платформа .NET. Типи значень і посилальні типи.		
1.	ЛЕКЦІЯ 1. Мови та парадигми програмування. Платформа .NET: середовище виконання CLR, проміжна мова IL, JIT-компіляція, збірки, пакети NuGet. Структура проєкту та рішення. Типи значень і посилальні типи, стек і керована купа. Структури (struct) та класи: відмінності й критерії вибору.	2
Тема 1.2. Класи і об'єкти. Керування пам'яттю та життєвий цикл об'єктів.		
2.	ЛЕКЦІЯ 2. Класи. Опис класу. Поля, властивості (автоматичні та обчислювані), методи. Модифікатори доступу. Конструктори, у тому числі первинні (primary constructors). Ініціалізатори об'єктів. Посилання this. Статичні члени, readonly та const.	2
3.	ЛЕКЦІЯ 3. Керування пам'яттю: збирач сміття та покоління об'єктів. Фіналізатори, інтерфейс IDisposable та конструкція using. Пакування і розпакування (boxing/unboxing). Порівняння з детермінованим руйнуванням об'єктів у C++.	2
Тема 1.3. Перевантаження методів та операцій.		
4.	ЛЕКЦІЯ 4. Перевантаження методів. Іменовані та необов'язкові параметри, params. Перевантаження операцій. Явні та неявні перетворення типів. Способи передавання параметрів ref, out, in. Кортежі.	2
Тема 1.4. Успадкування. Ієрархія класів. Абстрактні класи.		
5.	ЛЕКЦІЯ 5. Принцип успадкування. Модифікатори virtual, override, new, sealed. Ключове слово base. Конструктори в похідних класах. Ієрархія класів, тип object та його методи ToString, Equals, GetHashCode. Абстрактні класи й абстрактні методи.	2
Розділ 2. Абстракції, контракти та узагальнення		
Тема 2.1. Поліморфізм. Інтерфейси.		
6.	ЛЕКЦІЯ 6. Поліморфізм. Статичне і динамічне зв'язування. Приведення типів, операції is та as. Зіставлення зі зразком (pattern matching), switch-вирази.	2
7.	ЛЕКЦІЯ 7. Інтерфейси: оголошення, реалізація, явна реалізація, реалізації за замовчуванням. Чому в C# немає множинного успадкування класів. Стандартні інтерфейси IComparable<T>, IEquatable<T>, IEnumerable<T>. Композиція проти успадкування.	2
Тема 2.2. Принципи SOLID. Впровадження залежностей.		
8.	ЛЕКЦІЯ 8. Принципи SOLID на прикладах. Зв'язність і зачеплення модулів. Впровадження залежностей (dependency injection) через конструктор, контейнер Microsoft.Extensions.DependencyInjection, життєві цикли служб.	2
Тема 2.3. Узагальнене програмування. Колекції та ітератори.		
9.	ЛЕКЦІЯ 9. Узагальнені (generic) класи, структури та методи. Обмеження where. Коваріантність і контраваріантність. Порівняння узагальнень C# із шаблонами C++: конкретизація під час компіляції та під час виконання.	2

10.	ЛЕКЦІЯ 10. Колекції платформи .NET: List<T>, Dictionary<TKey,TValue>, HashSet<T>, Queue<T>, Stack<T>. Оцінка обчислювальної складності операцій. Ітератори, yield return, побудова власних перелічуваних типів.	2
Розділ 3. Надійність, обробка даних та сучасні засоби C#		
Тема 3.1. Обробка виняткових ситуацій. Рядки.		
11.	ЛЕКЦІЯ 11. Винятки: ієрархія Exception, конструкція try/catch/finally, фільтри when. Власні типи винятків. Відмінність throw і throw ex. Типові антипатерни обробки помилок. Рядки: незмінність типу string, StringBuilder, інтерполяція, форматування та культури.	2
Тема 3.2. Файлові потоки та серіалізація.		
12.	ЛЕКЦІЯ 12. Простір імен System.IO. Потоки Stream, FileStream, StreamReader і StreamWriter. Текстовий та двійковий режими. Довільний доступ до файлів, аналіз помилок введення/виведення. Серіалізація у форматі JSON засобами System.Text.Json.	2
Тема 3.3. Делегати, події та мова запитів LINQ.		
13.	ЛЕКЦІЯ 13. Делегати, анонімні методи та лямбда-вирази. Узагальнені делегати Func, Action, Predicate. Події: реалізація патерну «Спостерігач» засобами мови. LINQ to Objects: синтаксис запитів і методів, відкладене виконання, операції проєкції, фільтрації, групування та агрегування.	2
Тема 3.4. Асинхронне програмування.		
14.	ЛЕКЦІЯ 14. Потоки виконання та задачі (Task). Ключові слова async і await. Скасування операцій через CancellationToken. Асинхронні послідовності IAsyncEnumerable<T>. Типові помилки асинхронного коду та способи їх уникнення.	2
Тема 3.5. Патерни проектування. Якість програмного коду.		
15.	ЛЕКЦІЯ 15. Патерни об'єктно-орієнтованого проектування у .NET: Factory Method, Strategy, Observer, Decorator, Adapter, Singleton, Repository. Записи (record) та незмінні типи. Якість коду: модульне тестування (xUnit), посилальні типи з підтримкою null, статичні аналізатори, рефакторинг. Система контролю версій Git.	2
	Всього годин	30

Лабораторні роботи

N	**Назва лабораторних робіт**	К-сть ауд.год.
1.	Розроблення консольного застосунку на платформі .NET: типи значень і посилальні типи, структури та класи, колекції, форматований вивід. Репозиторій Git та файл .gitignore.	4
2.	Побудова класів предметної області: властивості з валідацією, конструктори, звільнення ресурсів через IDisposable, перевантаження операцій та методів ToString, Equals, GetHashCode.	4
3.	Створення ієрархії класів та інтерфейсів: абстрактний базовий клас, поліморфізм, реалізація IComparable<T> та IEquatable<T>, сортування й пошук.	4
4.	Рефакторинг застосунку згідно з принципами SOLID із впровадженням	4

	залежностей через конструктор.	
5.	Узагальнене програмування та LINQ: розроблення власної узагальненої колекції з реалізацією IEnumerable<T> та yield return, обчислення статистичних характеристик засобами LINQ.	4
6.	Обробка виняткових ситуацій, файлові потоки та серіалізація: збереження й відновлення стану предметної області у текстовому, двійковому та JSON форматах.	4
7.	Делегати, події та асинхронне програмування: асинхронне оброблення даних із відображенням поступу та можливістю скасування. Модульні тести засобами xUnit.	4
	Всього годин	28

6. Самостійна робота

Самостійна робота студента (64 години) передбачає підготовку до лекційних, практичних та лабораторних занять, самостійне опрацювання документації Microsoft Learn за темами лекцій, доопрацювання програмного коду лабораторних робіт поза аудиторними заняттями та підготовку до контрольних заходів.

Результати самостійної роботи над лабораторними роботами фіксуються в персональному репозиторії Git: історія комітів є частиною матеріалу, що перевіряється під час захисту.

Розподіл годин СРС: підготовка до лекцій – 15 годин (по 1 годині на лекцію); підготовка до практичних занять – 7 годин; підготовка та доопрацювання лабораторних робіт – 21 година (по 3 години на роботу); підготовка до модульної контрольної роботи – 6 годин; підготовка до екзамену – 15 годин. Разом – 64 години.

Політика та контроль

7. Політика навчальної дисципліни (освітнього компонента)

- Відвідування лекцій окремо не оцінюється, проте на лекціях та практичних заняттях проводиться усне опитування, за яке нараховуються бали за активність. Відвідування практичних та лабораторних занять є обов'язковою складовою вивчення матеріалу;
- При захисті лабораторних робіт студент має продемонструвати розроблений програмний код та результати його виконання на тестах, як заздалегідь підготованих, так і запропонованих викладачем. Програмний код лабораторних робіт розміщується у персональному репозиторії Git, здача виконується через Pull Request. У випадку дистанційної форми навчання захист відбувається на відповідній конференції шляхом демонстрації екрана.
- Використання систем генеративного штучного інтелекту (GitHub Copilot, ChatGPT, Claude тощо) під час виконання лабораторних робіт дозволяється за умови, що студент декларує факт їх застосування у звіті, здатен пояснити призначення кожного фрагмента коду та самостійно модифікувати його на вимогу викладача. Нездатність пояснити власний код розглядається як порушення академічної доброчесності незалежно від працездатності програми.
- Політика та принципи академічної доброчесності визначені у розділі 3 Кодексу честі Національного технічного університету України «Київський політехнічний інститут імені Ігоря Сікорського». Детальніше: https://kpi.ua/code.
- Норми етичної поведінки Норми етичної поведінки студентів і працівників визначені у розділі 2 Кодексу честі Національного технічного університету України «Київський політехнічний інститут імені Ігоря Сікорського». Детальніше: https://kpi.ua/code.
-

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

- Рейтинг студента з освітнього компонента розраховується зі 100 балів, з них 60 балів складає стартова шкала. Стартовий рейтинг (протягом семестру) складається з балів, що студент отримує за виконання лабораторних робіт (7 робіт), модульну контрольну роботу та активність на лекційних і практичних заняттях;
- Критерії нарахування балів за виконання лабораторних робіт.

Ваговий бал за виконання завдань лабораторної роботи складає 6 балів. Максимальна кількість балів за всі лабораторні роботи дорівнює

6 балів х 7 = 42 бали.

Мінімальна кількість балів для зарахування лабораторної роботи складає 4 бали (60%)

Максимальна кількість балів за модульну контрольну роботу дорівнює 10 балів. На модульну контрольну роботу виноситься одне теоретичне питання та одне практичне завдання.

Контрольна робота оцінюється наступним чином:

1. правильність відповіді на теоретичне питання – 4 бали;
2. наведення прикладу програмного коду до теоретичного питання – 2 бали;
3. правильність розв'язання практичного завдання – 4 бали.

- За активність на лекційних і практичних заняттях та виконання домашніх робіт нараховується максимум – 8 балів.

- Умови допуску до екзамену: зарахування всіх лабораторних робіт та стартовий рейтинг не менше 36 балів (60% від 60).

- Календарний контроль проводиться двічі на семестр як моніторинг поточного стану виконання вимог силабусу. Перший календарний контроль (тиждень 7–8): умова позитивної оцінки – не менше 13 балів. Другий календарний контроль (тиждень 14–15): умова позитивної оцінки – не менше 32 балів.

На екзамені студенти виконують письмову контрольну роботу. Екзаменаційний білет складається з двох теоретичних питань та одного практичного завдання. Ваговий бал кожного теоретичного питання – 10. Ваговий бал практичного завдання – 20.

Максимальна кількість балів за складання екзамену дорівнює

10 балів х 2+ 20 балів = 40 балів.

Теоретична частина оцінюється наступним чином:

- «відмінно» , правильна чітко викладена, повна відповідь – (не менше 90% потрібної інформації) – 9-10 балів;
- «добре», достатньо повна відповідь (не менше 70% потрібної інформації) – 7-8 балів;
- «задовільно», неповна відповідь (не менше 50% потрібної інформації) – 5-6 балів;
- «незадовільно», незадовільна відповідь – 0 балів.

Практичне завдання оцінюється наступним чином:

- «відмінно», повне, безпомилкове розв’язування завдання – 18-20 балів;
- «добре», повне, розв’язування завдання із несуттєвими неточностями – 15-17 балів;
- «задовільно», завдання виконане з певними недоліками – 12-14 балів;
- «незадовільно», завдання не виконано.

5. Сума стартових балів і балів за екзаменаційну контрольну роботу переводиться за освітній компонент згідно з таблицею.

Бали: *лабораторні роботи, МКР, активність +* *екзаменаційна контрольна робота*	*Оцінка*
100-95	Відмінно
94-85	Дуже добре
84-75	Добре
74-65	Задовільно

64-60	Достатньо
Менше 60	Незадовільно
Є не зараховані лабораторні роботи	Не допущено

Робочу програму навчальної дисципліни (силабус):

Складено асистентом, доктором філософії, Здором Костянтином Андрійовичем

Ухвалено кафедрою ЦТЕ (протокол № 22 від 17.06.2026)

Погоджено науково-методичною комісією НН ІАТЕ (протокол № 9 від 26.06.2026 р.)