МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Львівський національний університет імені Івана Франка
Факультет прикладної математики та інформатики
Кафедра програмування

Затверджено
На засіданні кафедри програмування
факультету прикладної математики
Львівського національного університету
імені Івана Франка
(протокол № 1 від серпня 202 р.)

Зав. кафедри к. ф.-м. н., доц. Ярошко С. А.

Силабус навчальної дисципліни

«Програмна Інженерія»,

що викладається в межах

ОПП «Середня освіта (Інформатика)»

*для здобувачів **першого (бакалаврського) рівня вищої освіти***

з предметної спеціальності **014.09 Середня освіта (Інформатика)**

галузі знань **01 Освіта/Педагогіка**

Львів 202 р.

Назва дисципліни	Програмна інженерія
Адреса викладання дисципліни	Львівський національний університет імені Івана Франка, вул. Університетська 1, м. Львів, Україна, 79000
Факультет та кафедра, за якою закріплена дисципліна	Факультет прикладної математики та інформатики, кафедра програмування
Галузь знань, шифр та назва спеціальності	Галузь знань: 01 Освіта/Педагогіка Спеціальність: 014.09 Середня освіта (Інформатика)
Викладачі дисципліни	Глова Андрій Романович, доктор філософії
Контактна інформація викладачів	Електронна пошта: andrii.hlova@lnu.edu.ua веб-сторінка: https://ami.lnu.edu.ua/employee/hlova-a-r
Консультації з питань навчання по дисципліні відбуваються	Консультації проводять раз на тиждень згідно з оприлюдненим розкладом консультацій викладача. Можливі он-лайн консультації через Zoom чи Microsoft Teams. Для погодження часу он-лайн консультацій слід писати на електронну пошту викладача.
Сторінка курсу	https://ami.lnu.edu.ua/course/prohramna-inzheneriia
Інформація про дисципліну	Курс "Програмна Інженерія" є нормативною дисципліною зі спеціальності 014.09 Середня освіта (Інформатика) для освітньої програми «Середня освіта (Інформатика)», яку викладають у п'ятому і шостому семестрах в обсязі 7,5 кредитів (за Європейською кредитно-трансферною системою ECTS)
Коротка анотація дисципліни	Фокус уваги курсу спрямовано на те, щоб навчити студентів розробляти програмні продукти з урахуванням сучасних підходів до програмної інженерії: планування, проектування, кодування, тестування та підтримки і налагодження програм. Використовуючи мову програмування C# та технології бібліотеки .Net, студенти здобудуть навички розробки десктопних програм під операційну систему Microsoft Windows та веб застосунків. Основні активності, що становлять суть програмної інженерії, у першому семестрі вивчаються та практично застосовуються на прикладі таких технологій, як ADO.Net, Entity FW, WPF, а у другому семестрі -- ASP.NET Core MVC.
Мета та цілі дисципліни	Метою нормативної дисципліни «Програмна інженерія» є навчити студента: ● застосовувати сучасні підходи до планування, дизайну, кодування, тестування та підтримки і налагодження програм ● створювати та аналізувати вимоги до програмних продуктів ● використовувати діаграми UML для відображення вимог до програм та опису архітектури та дизайну програм ● організовувати розробку програми, з допомогою однієї з методологій SDLC ● писати код згідно з основними принципами об'єктно-орієнтованого проектування та підходами 'чистого коду' ● розробляти десктопні програми для Windows з допомогою технології WPF ● розробляти веб програми з допомогою технології ASP.Net ● розробляти програми з доступом до баз даних через технологію ADO.Net та Entity FW ● розробляти ручні та автоматизовані тести для гарантії якості коду ● структуровано передавати здобуті теоретичні знання та практичний досвід іншим
Література для вивчення дисципліни	*Основна література* 1. Matthew MacDonald. Pro WPF 4.5 in C#. Windows Presentation Foundation in .NET 4.5, "Apress", 2012. -1078 2. Adam Freeman. Pro ASP.NET Core MVC 2 7th Edition, "Apress", 2017. -1451 3. Windows Presentation Foundation documentation https://docs.microsoft.com/en-us/dotnet/desktop/wpf/?view=netdesktop-5.0 4. Steve Smith. Architecting Modern Web Applications with ASP.NET Core and Azure. EDITION v5.0 -Updated to ASP.NET Core 5.0 --Redmond, Washington, 98052-6399 – 2021, Microsoft Corporation. 5. Architectural principles https://docs.microsoft.com/en-us/dotnet/architecture/modern-web-apps-azure/architectural-principles#separation-of-concerns 6. Robert Martin. Clean architecture. 7. https://herbertograca.com/2017/07/03/the-software-architecture-chronicles/ 8. ASP .NET Core fundamentals. https://learn.microsoft.com/en-us/aspnet/core/fundamentals/?view=aspnetcore-7.0&tabs=windows 9. Test ASP.NET Core MVC apps https://docs.microsoft.com/en-us/dotnet/architecture/modern-web-apps-azure/test-asp-net-core-mvc-apps

	10. Gamma Erich, Helm Richard, Johnson Ralph, Vlissides John. Design Patterns: Elements of Reusable Object-Oriented Software. Addison-Wesley, 1995. 11. C# Design Patterns https://www.dofactory.com/net/design-patterns *Додаткова література* 1. Russ Miles, Kim Hamilton. Learning UML 2.0: A Pragmatic Introduction to UML - 2006, 290 p 2. Get started with ASP.NET Core MVC https://docs.microsoft.com/en-us/aspnet/core/tutorials/first-mvc-app/start-mvc?view=aspnetcore-5.0&tabs=visual-studio 3. Robert Martin. Clean Code. 4. Hands-on cloud administration in Azure https://www.amazon.com/Hands-Cloud-Administration-Azure-components/dp/178913496X
Обсяг курсу	7,5 кредитів ЄКТС – 225 годин. З них 32 + 32 годин лекцій, 32 + 32 години лабораторних занять (1-й + 2-й семестр відповідно) та 97 годин самостійної роботи.
Очікувані результати навчання	Після завершення цього курсу студент буде: *знати* ● сучасні підходи до планування, дизайну, кодування, тестування та підтримки і налагодження програм ● способи представлення вимог до програмних продуктів ● різні типи UML діаграм ● життєвий цикл розробки програм ● основні принципи ООД та підходи ‘чистого коду’ *вміти* ● розробляти десктопні програми для Windows з допомогою технології WPF ● розробляти веб програми з допомогою технології ASP.Net ● розробляти програми з доступом до баз даних через технологію ADO.Net та Entity FW ● розробляти ручні та автоматизовані тести для гарантії якості коду ● структуровано передавати здобуті теоретичні знання та практичний досвід іншим
Компетентності	*Інтегральна:* Здатність розв’язувати складні спеціалізовані практичні завдання в галузі середньої освіти, що передбачає застосування концептуальних методів освітніх наук, знань з інформатики, педагогіки, психології, теорії та методики навчання інформатики і характеризується комплексністю та невизначеністю умов організації освітнього процесу в закладах загальної середньої освіти. *Загальні (ЗК):* ЗК4. Здатність орієнтуватися в інформаційному просторі, здійснювати пошук, аналіз та обробку інформації з різних джерел, ефективно використовувати цифрові ресурси та технології в освітньому процесі. ЗК8. Здатність зберігати та примножувати моральні, культурні, наукові цінності і досягнення суспільства на основі розуміння історії та закономірностей розвитку предметної області, її місця у загальній системі знань про природу і суспільство та значення у розвитку суспільства, техніки і технологій. *Спеціальні (фахові, предметні) компетентності (СК):* ФК1. Здатність перенесення системи наукових знань у професійну діяльність та в площину навчального предмету. ФК2. Здатність забезпечувати навчання учнів державною мовою; формувати та розвивати їх мовно-комунікативні уміння і навички в області предметної спеціальності. ПК4. Здатність використовувати програмні засоби загального та спеціального призначення для розв’язання прикладних задач з інформатики. ПК5. Володіння технологіями налагодження, обслуговування та експлуатації комп’ютерної мережі; здатність реалізовувати комплекс заходів, спрямованих на забезпечення захищеності інформації, здатність формувати вміння безпечної роботи школярів у комп’ютерній мережі.
Програмні результати навчання	ПРН4. Здійснює добір і застосовує сучасні освітні технології та методики для формування предметних компетентностей учнів; критично оцінює результати їх навчання та ефективність уроку.

	ПРН7. Демонструє знання основ фундаментальних і прикладних наук інформатики та програмування, оперує базовими категоріями та поняттями предметної області спеціальності. ПРН8. Генерує обґрунтовані думки в галузі професійних знань як для фахівців, так і для широкого загалу державною та іноземною мовами. ПРН11. Виявляє навички роботи в команді, адаптації та дії у новій ситуації, пояснює необхідність забезпечення рівних можливостей і дотримання гендерного паритету у професійній діяльності. ПРН12. Аналізує власну педагогічну діяльність та її результати, здійснює об'єктивну самооцінку і самокорекцію своїх професійних якостей. ПРН15. Використовує інформаційно-комунікаційні технології для подання, редагування, збереження та перетворення текстової, числової, графічної, звукової та відеоінформації. ПРН20. Створює інформаційні моделі, реалізує їх засобами інформаційно-комунікаційних технологій, здійснює дослідження, інтерпретує, аналізує та узагальнює його результати. ПРН21. Уміє реалізувати алгоритми розв'язання задач мовами програмування, вибирати й застосовувати інформаційно-комунікаційні технології; розв'язує задачі шкільного курсу інформатики різного рівня складності.
Ключові слова	Програмування мовою C#, .Net технології, WPF, ADO.Net, Entity FW, ASP.Net, MVC, десктопні програми, веб застосунки.
Формат курсу	Очний
Теми	

Тижд.	Тема, план, короткі тези	Форма заняття	Тривалість, год (сам.роб.)	Термін виконання
1.1	Життєвий цикл розробки програм (SDLC). Ролі в команді	Лекція	2 (3)	
	Формування команд, вибір теми проекту, вибір методології розробки проекту	Лабораторна робота	2	Наступне лабораторне заняття
1.2	Вимоги до програмних продуктів. Збір, типи, аналіз вимог	Лекція	2 (3)	
	Збір вимог щодо проекту, їх документування	Тест	2	Наступне лабораторне заняття
1.3	UML, основні види діаграм, які використовуються при проектуванні та розробці програм	Лекція	2 (3)	
	Застосування UML діаграм для опису проекту та його розробки	Лабораторна робота	2	Наступне лабораторне заняття
1.4	Принципи ООД - SOLID, KISS	Лекція	2 (3)	
	Використання StyleCop та Code Analysis для підтримки конвенцій коду C#	Лабораторна робота	2	Наступне лабораторне заняття
1.5	Аналіз коду інструментами StyleCop та Code Analysis	Лекція	2 (3)	
	Опис вимог інтерфейсу користувача - мокапи, вайфрейми	Лабораторна робота	2	Наступне лабораторне заняття
1.6	Технології ADO.NET доступу до даних. Провайдери даних	Лекція	2 (3)	
	Проектування бази даних, архітектурні діаграми	Тест	2	
1.7	Під'єднаний режим роботи з базою даних. Рядок з'єднання. команди, DataReader Від'єднаний режим роботи з базою даних. DataAdapter.	Лекція	2 (3)	

	Генерація бази даних, доступ через ADO.Net	Лабораторна робота	2	Наступне лабораторне заняття
1.8	Entity Framework, підходи DBFirst, ModelFirst, Code First Патерн Репозиторій та UnitOfWork	Лекція	2 (3)	
	Створення трьох рівневої архітектури проекту. Розробка рівня доступу до бази даних з допомогою Entity FW	Лабораторна робота	2	Наступне лабораторне заняття
1.9	Принципи та інструменти розробки інтерфейсу користувача, UX	Лекція	2 (3)	
	Розробка презентаційного рівня проекту, Графічний інтерфейс за допомогою бібліотеки WPF	Лабораторна робота	2	Наступне лабораторне заняття
1.10	Архітектура WPF. Ієрархія основних класів	Лекція	2 (3)	
	Додавання в програму необхідної графіки, анімації	Лабораторна робота	2	Наступне лабораторне заняття
1.11	XAML для розробки GUI. Обробка подій	Лекція	2 (3)	
	Розробка рівня бізнес логіки проекту	Лабораторна робота	2	Наступне лабораторне заняття
1.12	Елементи керування WPF. Data Binding. Стилі, шаблони WPF	Лекція	2 (4)	
	Покриття коду юніт тестами	Лабораторна робота	2	Наступне лабораторне заняття
1.13	Розробка WPF програм з використанням патерну MVVM	Лекція	2 (3)	
	Застосування патерну MVVM в WPF частині програми	Тест	2	
1.14	Inversion Of Control. Dependency Injection. Unity	Лекція	2 (3)	
	Логування подій, зауважень та помилок	Лабораторна робота	2	Наступне лабораторне заняття
1.15	Тестування програмного забезпечення	Лекція	2 (3)	
	Презентація та захист проекту	Лабораторна робота	2	Наступне лабораторне заняття
1.16	Logger. Логування подій, зауважень та помилок	Лекція	2 (3)	
	Презентація та захист проекту	Лабораторна робота	2	
2.1	Підходи до проектування складних систем. Класифікація вимог до ПЗ та відношення між ними.	Лекція	2 (3)	
	Створення проектів. Налаштування середовища	Лабораторна робота	2	Наступне лабораторне заняття
2.2	Процес розробки вимог:виявлення, аналіз, специфікація , валідація	Лекція	2 (3)	

	Формування переліку вимог до Web-аплікацій	Лабораторна робота	2	Наступне лабораторне заняття
2.3	Архітектура ПЗ: моделювання, архітектурні стилі	Лекція	2 (3)	
	Специфікація та валідація вимог. Use-case моделі	Лабораторна робота	2	Наступне лабораторне заняття
2.4	Архітектура ASP Core MVC Web-аплікацій. Контролери.	Лекція	2 (3)	
	Виокремлення підсистем, затвердження архітектури	Лабораторна робота	2	Наступне лабораторне заняття
2.5	Особливості розробки методів контролера	Лекція	2 (3)	
	Розробка моделі предметної області	Лабораторна робота	2	Наступне лабораторне заняття
2.6	Основи проектування Views	Лекція	2 (3)	
	Unit-тестування. Імплементація контролерів для окремої підсистеми	Лабораторна робота	2	Наступне лабораторне заняття
2.7	Проектування моделей предметної області. Використання БД	Лекція	2 (3)	
	Імплементація контролерів для окремої підсистеми	Лабораторна робота	2	Наступне лабораторне заняття
2.8	Проектування типізованих Views	Лекція	2 (3)	
	Імплементація методів та їхніх тестів для основної функціональності. Вдосконалення Views.	Лабораторна робота	2	Наступне лабораторне заняття
2.9	Оптимізація методів контролера	Лекція	2 (3)	
	Завершення імплементації методів та їхніх тестів для основної функціональності	Лабораторна робота	2	Наступне лабораторне заняття
2.10	ASP.NET Core Identity	Лекція	2 (3)	
	Рефакторинґ з врахуванням Identity	Лабораторна робота	2	Наступне лабораторне заняття
2.11	Класифікація атрибутів якості ПЗ. Техніки забезпечення якості (QA).	Лекція	2 (3)	
	Завершення імплементації повного набору юскейсів	Лабораторна робота	2	Наступне лабораторне заняття
2.12	Якість процесу розробки ПЗ	Лекція	2 (3)	
	Вдосконалення UI/UX	Лабораторна робота	2	Наступне лабораторне заняття
2.13	Патерни проектування GoF (DP). Структурні DP.	Лекція	2 (3)	
	Вдосконалення UI/UX	Лабораторна робота	2	Наступне лабораторне заняття
2.14	Твірні DP	Лекція	2 (3)	
	Демонстрація командами Web-аплікацій та рев'ю стосовно параметрів якості.	Лабораторна робота	2	Наступне лабораторне заняття
2.15	Поведінкові DP.	Лекція	2 (3)	

		Демонстрація командами Web-аплікацій та рев'ю стосовно параметрів якості.	Лабораторна робота	2	Наступне лабораторне заняття
	2.16	Патерни рефакторингу	Лекція	2 (3)	
		Демонстрація командами Web-аплікацій та рев'ю стосовно параметрів якості.	Лабораторна робота	2	
Підсумковий контроль, форма	іспит в кінці 6 семестру				
Пререквізити	Для вивчення курсу студенти потребують базових знань з дисциплін "Програмування", "Англійської мови" (термінологія), Одночасно з вивченням цього курсу, студенти в команді розробляють проекти з використанням сучасних підходів до розробки програм, тому доцільними є навички з навчальної (обчислювальної) практики.				
Навчальні методи та техніки, які використовують під час викладання курсу	Лекції з мультимедійними презентаціями та з демонстрацією прийомів практичного використання середовища програмування; лабораторні заняття у вигляді розробки проектів у команді з 3-5 осіб; самостійне опрацювання навчальних матеріалів: підручників, конспектів лекцій, готових програм мовою C#, додаткових навчальних посібників, розміщених у хмарному сховищі (Moodle, Microsoft Teams, Google Classroom). Обговорення теоретичного та практичного матеріалу в онлайн сервісах, формулювання творчих завдань для студентів, виконання яких готує до вивчення нового теоретичного матеріалу.				
Необхідне обладнання	Для проведення лекцій: комп'ютер, проектор, доступ до мережі інтернет. Для проведення лабораторних та виконання завдань: комп'ютер, ОС Windows, доступ до інтернету, середовище розробки програм технологіями .Net C# (Microsoft Visual Studio, Code Blocks тощо). Уся література, яку студенти не зможуть знайти самостійно, буде надана викладачем виключно в освітніх цілях без права її передачі третім особам. Студенти заохочуються до використання також й іншої літератури та джерел, яких немає серед рекомендованих.				
Критерії оцінювання (окремо для кожного виду навчальної діяльності)	**Оцінювання** проводиться за 100-бальною шкалою. У п'ятому семестрі до 70 балів нараховується за розробку проекту, ще до 30 балів за знання теоретичної частини курсу. У шостому семестрі до 50 балів нараховують за розробку проекту, ще до 50 балів – за виконання екзаменаційного завдання. Розробка проекту поділена на 6-8 частин, кожна з яких оцінюється 5-10 балів залежно від складності. Для кожного проекту визначено термін виконання: зазвичай до закінчення навчального тижня. Вчасно виконані проекти оцінюють так (у відсотках від максимальної оцінки): • 100% – умови завдання виконано повністю, алгоритми складено правильно, програма містить належні коментарі, роботу програми перевірено на достатньому наборі тестових даних, автор відповідає на всі запитання щодо використаних підходів, чітко інтерпретує отримані результати, немає ознак недоброчесності; • 80% – наведено логічно правильну послідовність розв'язування, алгоритми складено правильно, бракує окремих коментарів чи тестів, автор не досить повно пояснює використані підходи, немає ознак недоброчесності; • 60% – у правильній послідовності розв'язування допущено окремі помилки, які автор уміє виправити після зауваження викладача, бракує коментарів чи тестів, на запитання щодо використаних підходів автор відповідає з помилками, немає ознак недоброчесності; • 40% – у правильній послідовності розв'язування пропущено окремі етапи, завдання виконано частково, автор не розуміє недоліків поданої роботи, не вміє їх виправити, немає ознак недоброчесності; • 20% – завдання виконано частково, немає тестів, програма працює правильно для окремих наборів вхідних даних, автор не може самостійно інтерпретувати отримані результати, виправити помилки, немає ознак недоброчесності; • 0% – завдання не виконано, написана програма не відповідає умові, або ж виявлено ознаки недоброчесності: запозичення, фраґменти коду, дію яких автор пояснити не може, автор не володіє відповідним теоретичним матеріалом тощо; • можуть бути нараховані додаткові бали за повністю виконане завдання, яке містить кілька способів розв'язування, використовує особливо ефективний спосіб, демонструє креативність автора тощо. Запізнення зменшує максимальну оцінку за проект: кожного наступного після терміну виконання тижня оцінка зменшується удвічі.				

	Оцінка за екзаменаційне завдання може бути поділена на дві частини: до 30 балів за розробку нової функціональності програми з оновленням відповідних моделей та специфікацій і 20 балів за засвоєння матеріалу усіх розділів курсу, з яких частина може бути виставлена в процесі опитувань упродовж семестру (у формі тестувань, колоквіумів тощо). **Відвідання занять** є важливою складовою навчання. Очікується, що всі студенти відвідають усі лекції і лабораторні заняття курсу. Активність під час проведення лекцій і лабораторних заохочується балами. У будь-якому випадку студенти зобов'язані дотримуватися усіх строків визначених для виконання усіх видів письмових робіт, передбачених курсом. Виконані роботи завантажують у відповідне хмарне сховище. За відповідних обставин альтернативою відвідування лабораторних занять в університеті може бути дистанційна онлайнова робота за розкладом проведення занять. Активність на лекціях і лабораторних ураховують при оцінюванні відповідного лабораторного завдання. **Академічна доброчесність**: роботи студентів мають бути їхніми оригінальними дослідженнями, розробками чи міркуваннями. Відсутність посилань на використані джерела, фабрикування джерел, списування, втручання в роботу інших студентів, здавання чужих комп'ютерних програм як своїх становлять, але не обмежують, приклади можливої академічної недоброчесності. Виявлення ознак академічної недоброчесності в роботі студента є підставою для її незарахування викладачем, незалежно від масштабів плагіату чи обману.
Опитування	Анкету-оцінку з метою оцінювання якості курсу буде надано після завершення курсу.