СИЛАБУС НАВЧАЛЬНОЇ ДИСЦИПЛІНИ
«МОДИФІКАЦІЯ ТА ТЕСТУВАННЯ КОМП'ЮТЕРНИХ СИСТЕМ»

Ступінь освіти	магістр
Спеціальність	122 Комп'ютерні науки
Освітня програма	Комп'ютерні науки
Тривалість викладання	Осінній семестр (1, 2 чверть)
Загальний обсяг	4 кредитів ЄКТС (120 годин)
Заняття:	
лекції:	2 години/тиждень
лабораторні заняття:	1 година/тиждень (1чв.) 2 години/тиждень (2чв.)
Мова викладання	українська

Сторінка курсу в СДО НТУ «ДП»: https://do.nmu.org.ua/course/view.php?id=6668

Кафедра, що викладає Програмного забезпечення комп'ютерних систем

Викладач:
Спірінцев В'ячеслав Васильович
доцент, к.т.н.

Персональна сторінка
https://pzks.nmu.org.ua/ua/teachers/teachers.php

E-mail: Spirintsev.V.V@nmu.one

1. Анотація до курсу

Термін експлуатації програмного продукту неможливо передбачити на початку його створення, адже проблеми його використання можуть виникнути при виході нових операційних систем, фреймворків, нового обладнання та апаратно-технічних засобів, які диктують умови використання створюваного програмного забезпечення (ПЗ). Розробники ПЗ стикаються з низкою проблем: зростання тиску з метою швидкої доставки якісної продукції; збільшення розміру і складності програмного забезпечення і систем; зростаючі вимоги до відповідності національним, міжнародним та професійним стандартам; субпідряд та аутсорсинг; розподілені робочі групи; постійна зміна платформ та технологій. Задля збереження конкурентоздатності своєї продукції розробники ПЗ забезпечують постійну підтримку, додавання нових функцій та адаптацію наявних відповідно до нових вимог. Виявлення дефектів та їх усунення під час створення ПЗ впливає на кінцеву вартість програмного продукту та загальний час на розробку, тому що залежить від етапу, на якому їх виявлено, а цей етап залежить від обраної методології командою розробників. Щоб уникати руйнування ПЗ та підтримувати цілісність архітектури, необхідно часто проводити рефакторинг (англ. *refactoring*, процес зміни внутрішньої структури програмного забезпечення, що має на

меті полегшити розуміння його роботи і спростити модифікацію, без зміни зовнішньої поведінки самої системи), щоб бути впевненим, що код залишається чистим та лаконічним та з мінімальними залежностями між модулями. В основі рефакторингу лежить послідовність невеликих еквівалентних (тобто зберігають поведінку) перетворень. Оскільки кожне перетворення невелике, розробнику легше простежити за його правильністю, і в той же час вся послідовність може призвести до суттєвої перебудови програми та покращення її узгодженості та чіткості. Якість коду та метрики проектування пропонують об'єктивні рекомендації щодо визначення ділянок коду в додатку, які є першочерговими кандидатами для рефакторингу. Метрики покриття дозволяють виконувати роботи з рефакторингу. Сучасні методи тестування ПЗ не дозволяють однозначно і повністю виявити всі дефекти і встановити коректність функціонування програми, тому всі існуючі методи тестування діють у рамках формального процесу перевірки ПЗ, що досліджується або розробляється. Такий процес формальної перевірки може довести, що дефекти відсутні з точки зору використовуваного методу. Тестування дозволяє зробити процес розробки ПЗ прозорим і керованим для всіх учасників проєкту. Продуктивність та якість ПЗ суттєво залежить від ефективності модифікації та тестування, тому дане питання є актуальними і потребує подальшого розгляду. Основною метою курсу є формування бази теоретичних та практичних знань в області модифікації та тестування комп'ютерних систем для здобувачів освіти Національного технічного університету «Дніпровська політехніка» спеціальності 122 «Комп'ютерні науки» для вирішення різноманітних практичних задач у їхній діяльності, що сприяє підвищенню якості їхньої підготовки.

В рамках курсу розглядаються такі питання: методи та засоби реінженерії програмних систем, метрики коду програмного забезпечення, рефакторинг та оптимізаційні перетворення ПЗ, основні поняття та підходи до тестування, критерії вибору тестів, оцінка відтестованості проекту: метрики та методики інтегральної оцінки, модульне та інтеграційне тестування, системне та регресивне тестування, автоматизація тестування, документування та метрики індустріального тестування, тестування інтерфейсу кориистувача, формальні інспекції.

2. Мета та завдання курсу

Мета дисципліни – формування у магістрів глибокого розуміння питань та компетентностей щодо модифікації та тестування комп'ютерних систем з урахуванням реалій сучасного ІТ-бізнесу для підвищення ефективності та надійності інформаційних систем, а також подальшого становлення і вдосконалення інформаційної та програмної культури майбутніх фахівців.

Завданнями дисципліни є:

- опанування теоретико-понятійної бази курсу;
- ознайомлення здобувачів з сучасними методами та засобами реінженерії програмних систем;
- опанування загальних підходів до рефакторингу та оптимізаційних перетворень комп'ютерних систем;
- ознайомлення здобувачів з особливостями та основними підходами до тестування комп'ютерних систем.

3. Результати навчання

Дисциплінарні результати навчання:

- виявляти та усувати проблемні ситуації в процесі експлуатації програмного забезпечення, формулювати завдання для його модифікації або за допомогою методів реінженерії програмного забезпечення;
- модифікувати існуючі програмні продукти за допомогою рефакторингу програмного коду;
- забезпечувати якість програмного забезпечення за допомогою багатофакторного та багаторівневого тестування;
- здійснювати конфігурування та реінжиніринг програмного забезпечення з урахуванням вимог замовника;
- здійснювати тестування програмного забезпечення різними методами і засобами.

Дисциплінарні результати навчання сформовано на основі ПРН освітньо-професійної програми «Комп'ютерні науки» другого (магістерського) рівня вищої освіти (РН05, РН11, РН13, РН14, РН17, РН18).

4. Структура курсу

Види та тематика навчальних занять	**Внесок в загальну оцінку, %**
ЛЕКЦІЇ	**40**
1. Методи та засоби реінженерії програмних систем Введення в курс. Методи еволюційного змінювання компонентів і систем. Поняття реінжинірингу програмних систем. Основний зміст реінжинірингу та його місце в ЖЦ інформаційної системи. Основні етапи процесу реінжинірингу. Рефакторинг компонентів. Реверсна інженерія.	
2. Метрики коду програмного забезпечення Кількісні метрики. Метрики складності потоку керування програми. Метрики складності потоку керування даними. Метрики складності потоку керування і даних програми. Об'єктно-орієнтовані метрики. Гібридні метрики.	
3. Рефакторинг програмного забезпечення Поняття та принципи рефакторингу. Основні цілі та причини проведення рефакторингу. Проблеми рефакторингу. Code Smells (запахи коду). Прийоми рефакторингу: складання методів, переміщення функцій між обєктами, організація даних, спрощення умовних виразів, спрощення викликів методів, рішення завдань узагальнення. Інструментальні засоби проведення рефакторингу.	
4. Патерни проектування Введення та основні поняття патернів проектування. Класифікація патернів. Породжуючі патерни. Структурні патерни. Патерни поведінки.	
Тестова контрольна робота №1 (за темами 1-4)	**20**
5. Основні поняття тестування Основні поняття, види та рівні тестування. Мета, задачі та базова термінологія тестування. Життєвий цикл тестування програмного забезпечення. Підходи до тестування. Класифікація тестування.	

Класифікація помилок. Тестове оточення.	
6. Артефакти тестування Test Plan. Check-list. Test Case & Test suite. Bug Reports / Defects.	
7. Модульне та інтеграційне тестування Цілі і завдання модульного тестування. Принципи структурного тестування. Методи проектування тестових шляхів в структурному тестуванні. Модульне тестування та Test-Driven Development. Інтеграційне тестування. Методи інтеграційного тестування. Інтеграційне тестування для процедурного програмування.	
8. Системне тестування Цілі і завдання системного тестування. Види системного тестування. Функціональне тестування. Тестування продуктивності. Стресове тестування. Тестування конфігурації. Тестування безпеки. Тестування надійності і відновлення після збоїв. Тестування зручності використання.	
9. Регресивне тестування Цілі, задачі і види регресивного тестування. Види регресивного тестування. Умови застосування методів вибіркового регресивного тестування. Класифікація методів вибіркового регресивного тестування. Різновиди методу відбору тестів. Інтеграційне регресивне тестування і регресивне тестування об'єктно-орієнтованих програм. Метод оптимізації програми, методи впорядкування тестів. Методики регресивного тестування.	
10. Автоматизація тестування Структура тестового набору для автоматизованого прогону. Генератори тестів. Структура інструментальної системи автоматизації тестування. Порівняння витрат і ефективності різних методів тестування. Інструментальні засоби автоматизованого тестування.	
11. Тестування інтерфейсу користувача. Формальні інспекції Завдання і цілі тестування інтерфейсу користувача. Функціональне тестування інтерфейсу користувача. Типи вимог до інтерфейсу користувача. Тестування зручності використання інтерфейсу користувача. Завдання і цілі проведення формальних інспекцій. Етапи формальної інспекції та ролі її учасників. Формальні інспекції програмного коду.	
Тестова контрольна робота №2 (за темами 5-11)	**20**
ЛАБОРАТОРНІ ЗАНЯТТЯ	**60**
Лабораторна робота 1 Обчислення метрик програмних систем.	
Звіт з роботи № 1 та захист лабораторної роботи.	15
Лабораторна робота 2 Рефакторинг програмного забезпечення.	
Звіт з роботи № 2 та захист лабораторної роботи.	15
Лабораторна робота 3 Патерни проектування програмних систем.	
Звіт з роботи № 3 та захист лабораторної роботи.	15
Лабораторна робота 4 Модульне тестування програмного забезпечення.	
Звіт з роботи № 4 та захист лабораторної роботи.	15
РАЗОМ	**100**

5.Технічне обладнання та/або програмне забезпечення

Використовуються лабораторії кафедри програмного забезпечення комп'ютерних систем (комп’ютерне та мультимедійне обладнання). Дистанційна платформа Moodle, MS Office 365, Microsoft Teams, Visual Studio, C#.

6. Система оцінювання та вимоги

6.1. Навчальні досягнення здобувачів вищої освіти за результатами вивчення курсу оцінюватимуться за шкалою, що наведена нижче:

Рейтингова шкала	Інституційна шкала
90 – 100	відмінно
74 – 89	добре
60 – 73	задовільно
0 – 59	незадовільно

6.2. Здобувачі вищої освіти можуть отримати **підсумкову оцінку** з навчальної дисципліни **на підставі поточного оцінювання знань** за умови, якщо набрана кількість балів з поточного тестування та виконання і захисту лабораторних робіт складатиме не менше 60 балів.

Теоретична частина оцінюється за результатами здачі двох контрольних тестових робіт, кожна з яких містить тестові закриті запитання з однією вірною відповіддю (максимальна кількість – 20 балів за кожною тестовою роботою). Загалом за дві контрольні тестові робіти отримується **максимум 40 балів**, тобто 40% від оцінки за дисципліну.

Лабораторні роботи (4 роботи – у вигляді індивідуального завдання з кожної, розподіл % див. в таблиці розділу 4) виконуються у письмовому вигляді (звіт з кожної роботи оцінюється в межах балів, представлених в таблиці розділу 4, загалом лабораторні враховуються як 60% (максимум 60 балів). При несвоєчасному здаванні роботи оцінка знижується вдвічі. Отримані бали за теоретичну частину та лабораторні роботи додаються і є підсумковою оцінкою за вивчення навчальної дисципліни. Максимально за поточною успішністю здобувач вищої освіти може набрати 100 балів.

6.3. Критерії оцінювання підсумкової роботи. У випадку якщо здобувач вищої освіти за поточною успішністю отримав менше 60 балів та/або прагне поліпшити оцінку проводиться **підсумкове оцінювання**.

Диференційований залік проводиться у вигляді комплексної контрольної роботи, яка включає запитання з теоретичної та практичної частини курсу. Білет складається з **30 тестових завдань** з чотирма варіантами відповідей, одна правильна відповідь оцінюється в 2 бали **(разом 60 балів)** та **2 завдань** з практичної частини, кожне з запитань оцінюється максимум у 20 балів **(разом 40 балів**), причому:

- 20 балів – відповідність еталону;
- 15 балів – відповідність еталону з незначними помилками;
- 10 балів – часткова відповідність еталону, питання повністю не розкрите;
- 5 балів – невідповідність еталону, але відповідність темі запитання;
- 0 балів – відповідь не наведена або не відноситься до теми запитання.

Отримані бали за тестові завдання та завдання з практичної частини додаються і є підсумковою оцінкою за вивчення навчальної дисципліни. Максимально за

підсумковою роботою здобувач вищої освіти може набрати 100 балів.

7. Політика курсу

7.1. Політика щодо академічної доброчесності. Академічна доброчесність здобувачів вищої освіти є важливою умовою для опанування результатами навчання за дисципліною і отримання задовільної оцінки з поточного та підсумкового контролів. Академічна доброчесність базується на засудженні практик списування (виконання письмових робіт із залученням зовнішніх джерел інформації, крім дозволених для використання), плагіату (відтворення опублікованих текстів інших авторів без зазначення авторства), фабрикації (вигадування даних чи фактів, що використовуються в освітньому процесі). Політика щодо академічної доброчесності регламентується положенням "Положення про систему запобігання та виявлення плагіату у Національному технічному університеті "Дніпровська політехніка" https://cutt.ly/MCfh5kv

У разі порушення здобувачем вищої освіти академічної доброчесності (списування, плагіат, фабрикація), робота оцінюється незадовільно та має бути виконана повторно. При цьому викладач залишає за собою право змінити тему завдання.

7.2. Комунікаційна політика. Здобувачі вищої освіти повинні мати активовану корпоративну університетську пошту.

Усі письмові запитання до викладачів стосовно курсу мають надсилатися на університетську електронну пошту.

7.3. Політика щодо перескладання. Роботи, які здаються із порушенням термінів без поважних причин оцінюються на нижчу оцінку. Перескладання підсумкового оцінювання відбувається із дозволу деканату за наявності поважних причин (наприклад, лікарняний).

7.4 Політика щодо оскарження оцінювання. Якщо здобувач вищої освіти не згоден з оцінюванням його знань він може опротестувати виставлену викладачем оцінку у встановленому порядку.

7.5. Відвідування занять

Для здобувачів вищої освіти денної форми відвідування занять є обов’язковим. Поважними причинами для неявки на заняття є хвороба, участь в університетських заходах, академічна мобільність, які необхідно підтверджувати документами. Про відсутність на занятті та причини відсутності здобувач вищої освіти має повідомити викладача або особисто, або через старосту.

За об’єктивних причин (наприклад, міжнародна мобільність) навчання може відбуватись в он-лайн формі за погодженням з керівником курсу.

7.6. Участь в анкетуванні

Наприкінці вивчення курсу та перед початком сесії здобувачам вищої освіти буде запропоновано анонімно заповнити електронні анкети (Microsoft Forms Office 365), які буде розіслано на їхні університетські поштові скриньки. Заповнення анкет є важливою складовою їхньої навчальної активності, що дозволить оцінити дієвість застосованих методів викладання та врахувати надані пропозиції стосовно покращення змісту навчальної дисципліни.

8. Рекомендовані джерела інформації

1. С.В. Баран. Розробка програмного забезпечення з використанням патернів проектування: Навчальний посібник. – Кривий Ріг: Державний університет економіки і технологій, 2023. –203 с.
2. Авраменко А.С., Авраменко В.С., Косенюк Г.В. Тестування програмного забезпечення. Навчальний посібник/ А.С. Авраменко, В.С. Авраменко, Г.В. Косенюк. – Черкаси: ЧНУ імені Богдана Хмельницького, 2017. – 284 с.
3. Табунщик Г.В. Інженерія якості програмного забезпечення: навчальний посібник, 2-ге видання./ Г.В Табунщик, Р.К. Кудерметов, Т.І. Каплієнко // Запоріжжя: Дике Поле, 2016. – 176 с.
4. Visual Studio [Electronic resourse]. – Access mode: https://learn.microsoft.com/ru-ru/visualstudio/code-quality/code-metrics-values?view=vs-2022
5. SonarQube [Electronic resourse]. – Access mode: https://docs.sonarsource.com/sonarqube/latest/analyzing-source-code/overview/
6. Refactoring. Improving the Design of Existing Code Second Edition Martin Fowler with contributions by Kent Beck [Electronic resourse]. – Access mode: https://dl.ebooksworld.ir/motoman/Refactoring.Improving.the.Design.of.Existing.Code.2nd.edition.www.EBooksWorld.ir.pdf
7. Robert C. Martin. Clean code. Creating and refactoring with Agile [Electronic resourse]. – Access mode: https://github.com/martinmurciego/good-books/blob/master/Clean%20Code_%20A%20Handbook%20of%20Agile%20Software%20Craftsmanship%20-%20Robert%20C.%20Martin.pdf
8. Рефакторинг // Refactoring.Guru [Electronic resourse]. – Access mode: URL: https://refactoring.guru/ru/refactoring
9. Clausen Ch. Five Lines of Code: How and when to refactor // Manning, 2021 – 338 p. – ISBN 9781617298318
10. Андрій Будай. Дизайн-патерни — просто, як двері [Електронний ресурс]. – Режим доступу: https://programming.in.ua/programming/basisprogramming/268-design-patterns-book-andriy-buday.html
11. Патерни проектування [Електронний ресурс]. – Режим доступу: https://abitap.com/category/paterny-proektuvannya/
12. Е. Гамма Р. Хелм Р. Джонсон Дж. Вліссідес. Design Patterns: Elements of Reusable Object-Oriented Software [Електронний ресурс]. – Режим доступу: http://www.uml.org.cn/c++/pdf/DesignPatterns.pdf
13. Svyatoslav Kulikov. Software testing. Base course, 3rd Edition/ EPAM Systems, 2022.-278p.