Міністерство освіти і науки України
Львівський національний університет природокористування
Факультет механіки, енергетики та інформаційних технологій
Кафедра інформаційних технологій

СИЛАБУС
НАВЧАЛЬНОЇ ДИСЦИПЛІНИ
«Якість програмного забезпечення та тестування»

для усіх освітньо-професійна програм та спеціальностей
перший (бакалаврський) рівень вищої освіти
(вибіркова дисципліна загальноуніверситетського вибору)

ВИКЛАДАЧІ
Шувар Богдан Іванович
Ковалишин Олег Степанович

Електронна пошта:

b.i.shuvar@gmail.com

Телефон

+380974931871

Доцент кафедри інформаційних технологій Львівського національного університету природокористування (з 2022 року), кандидат економічних наук, доцент. Викладач з 13-річним досвідом, автор та співавтор понад 25 наукових статей, 1 монографії, 30 навчально-методичних розробок, фахівець у ВНС Moodle ЛНУП та Microsoft365.

Читає курси: Хмарні технології (Cloud-технології), Комп'ютерні технології з основами програмування.

ЛЬВІВ 2022

Рівень вищої освіти – ***перший (бакалавр)***
Кількість кредитів – 3
Рік підготовки, семестр – ***3 рік***
Компонент освітньої програми: ***вибіркова загальноуніверситетського переліку***
Мова викладання: ***українська***

Опис дисципліни

Мета навчальної дисципліни: розширення та поглиблення теоретичних знань та застосування умінь та навичок у базових поняттях та визначеннях у сфері забезпечення якості тестування програмного забезпечення, критерії відбору тестів, огляд типів тестування, аналіз процесу тестування та технологія промислових випробувань, набуття сучасних прикладних навичок. інформаційні технології для аналізу та тестування інформаційних систем, створення звітної тестової документації

Міждисциплінарні зв'язки: Вивчення дисципліни передбачає наявність систематичних та ґрунтовних знань із суміжних курсів – комп'ютерні мережі, основи інформаційних технологій, алгоритмізація та програмування

Вимоги до знань та умінь визначаються галузевими стандартами вищої освіти України.

Мета вивчення дисципліни. Якість програмного забезпечення та тестування є підготовка спеціаліста, який володіє базовими знаннями про основні види та методи тестування програмного забезпечення (ПЗ) при структурному та об'єктно-орієнтованому підході у програмуванні, знає способи забезпечення якості ПЗ, класи критеріїв тестування, різновиди тестування.

Основні завдання освітньої компоненти є формування сукупності знань щодо прийомів ручного тестування ПЗ, особливостей системного, модульного та інтеграційного тестування, моделей оцінки ступеню тестування програмного продукту, та вмінь оцінювати складність програмного продукту з використанням математичної моделі, використовувати методи ручного та автоматизованого тестування ПЗ, створювати набір тестів для тестування простих та складних систем.

Професійні компетентності, які отримують студенти після вивчення навчальної дисципліни: Інтегральна компетентність: здатність використовувати поглиблені теоретичні та фундаментальні знання, уміння і навички для успішного розв'язування спеціалізованих та практичних задач під час професійної діяльності у галузі інформаційних технологій.

Спеціальні (фахові, предметні) компетентності: здатність ідентифікувати, класифікувати та формулювати вимоги до програмного забезпечення; здатність формулювати та забезпечувати вимоги щодо якості програмного забезпечення у відповідності з вимогами замовника, технічним завданням та стандартами; здатність оцінювати і враховувати економічні, соціальні, технологічні та екологічні чинники, що впливають на сферу професійної діяльності.

Програмні результати навчання: аналізувати, цілеспрямовано шукати і вибирати необхідні для вирішення професійних завдань інформаційно-довідникові ресурси і знання з урахуванням сучасних досягнень науки і техніки; знати кодекс професійної етики, розуміти соціальну значимість та культурні аспекти інженерії програмного забезпечення і дотримуватись їх в професійній діяльності; знати і застосовувати професійні стандарти і інші нормативно-правові документи в галузі інженерії програмного забезпечення; знати та вміти застосовувати методи верифікації та валідації програмного забезпечення; знати підходи щодо оцінки та забезпечення якості програмного забезпечення.

1. Програма навчальної дисципліни

Тема 1. Місце КПЗ в життєвому циклі програмної системи

Складові частини розробки ПЗ. Місце конструювання при побудові ПЗ. Область знань "Якість програмного забезпечення та тестування комп'ютерних систем і мереж". Задачі, що виникають в процесі розробки ПЗ, пов'язані з конструюванням.

Тема 2. Фундаментальні складові Якість програмного забезпечення та тестування комп'ютерних систем і мереж

Мінімізація складності. Очікування змін. Конструювання з можливістю перевірки. Стандарти у конструюванні.

Тема 3. Мінімізація складності

Зменшення складності у конструюванні програмного забезпечення. Мінімізація складності за рахунок слідування стандартам. Використання низки специфічних технік кодування і підтримкою практик, спрямованих на забезпечення якості в конструюванні.

Тема 4. Очікування змін

Стрімка мінливість програмних систем. Причини мінливості. Прив'язка програмних систем до технологічних процесів. Порядок змін програмних систем, у відповідності до змін процесів.

Тема 5. Конструювання з можливістю перевірки

Огляд, оцінка коду (code review). Модульне тестування (unit-testing). Структурування коду для і спільно з застосуванням автоматизованих засобів тестування (automated testing). Обмежене застосування складних або важких для розуміння мовних структур

Тема 6. Стандарти у конструюванні

Комунікаційні методи. Мови програмування і відповідні стилі кодування. Платформи. Інструменти.

Тема 7. Високоякісне кодування

Можливість приховування реалізації. Більш висока інформативність інтерфейсу. Легкість оптимізації коду. Легкість читання і зрозумілість коду. Обмеження області використання даних рамками одного класу. Можливість роботи з сутностями реального світу, а не низькорівневими деталями реалізації.

Тема 8. Правила написання якісного коду. Рівень класів

Вираження в інтерфейсі класу узгоджений рівень абстракції. Надання методі разом з протилежними до них методами. Перенесення сторонньої інформацію в інші класи. Розгляд абстракції і зв'язності разом.

Тема 9. Принципи використання змінних

Грамотне оголошення змінних. Принципи ініціалізації змінних. Одиничність мети кожної змінної. Принципи вибору імен змінних.

Тема 10. Структурне програмування

Суть структурного програмування. Призначення структурного програмування. Керуючі структури. Складність керуючої логіки.

Тема 11. Рефакторинг

Частота зміни коду у ході роботи над проектом. Зміни коду у відповідності до змін системи. Правила еволюції програмного коду у ході проекту.

Тема 12. Еволюція програми

Фактори еволюції. Покращення якості коду у ході еволюції. Правила внесення змін на тій чи іншій стадії еволюції.

Тема 13. Поняття рефакторингу

Основні правила еволюції програмного коду. Правила Мартіна Фаулера.

Тема 14. Ознаки необхідності застосування рефакторингу

Дублювання коду. Покращення занадто довгого коду. Покращення некоректних імен методів. Покращення недостатнього рівня абстракції.

Тема 15. Рівні рефакторингу

Рефакторинги рівня даних. Рефакторинги рівня операторів. Рефакторинги рівня методів. Рефакторинги рівня реалізації класу. Рефакторинги рівня інтерфейсу класу. Рефакторинги рівня системи

Тема 16. Безпечний рефакторинг

Збереження початкового коду. Обмеження об’єму окремих видів рефакторингу. Виконання окремих видів рефакторингу по одному за раз. Складання списку дій, які програміст збирається виконати. Складання і підтримка списку видів рефакторингу, які потрібно виконати пізніше. Часте створення контрольних точок. Використання попереджень компілятора. Виконання регресивного тестування. Створення додаткових тестів. Виконання оглядів змін. Зміна підходу в залежності від ризикованості рефакторингу.

Тема 17. Стратегії рефакторингу

Виконувати рефакторинг при створенні нових методів. Виконувати рефакторинг при створенні нових класів. Виконувати рефакторинг при виправленні дефектів. Виконувати рефакторинг модулів, в яких велика ймовірність виникнення помилок. Виконувати рефакторинг складних модулів. При супроводженні програми покращувати фрагменти, які доводиться виправляти. Визначити інтерфейс між акуратним і поганим
кодом та перенести поганий код на інший бік цього інтерфейсу.

Тема 18. Якість конструювання

Тестування коду розробником. TDD (Test-Driven Development).. Переваги, які надає TDD. Фреймворк JUnit.

Тема 19. Тестування коду розробником.

Unit-тестування. Компонентне тестування. Інтеграційне тестування.. Регресійне тестування. Системне тестування.

Тема 20. TDD (Test-Driven Development).

Написання тесту для визначення поведінки програмної одиниці.. Створення програмної одиниці якомога простішими засобами так, щоб вона. пройшла тест. Здійснення рефакторингу коду, для покращення якості.. Тестування після кожної зміни.

Тема 21. Переваги, які надає TDD.

Спрощена, інкрементна розробка. Можливість постійного регресійного. тестування. Покращена комунікація і централізація знань. Покращене. розуміння вимог до програми, і, відповідно, покращений дизайн програми.. Покращена інкапсуляція і модульність. Зменшення складності за рахунок не. внесення надлишкового коду.

Тема 22. Фреймворк JUnit.

Анотації. Методи. Приклади застосування методів.

Літературні джерела

1. ДСТУ 2873-94. Системи обробки інформації. Програмування. Терміни та визначення. К.: Держстандарт України, 1994.
2. ДСТУ 2941-94. Системи оброблення інформації. Розроблення систем. Терміни та визначення. - К.: Держстандарт України, 1994.
3. ДСТУ 4302:2004. Інформаційні технології. Настанови щодо документування комп’ютерних програм. К.: Держстандарт України, 2004.
4. ДСТУ ISO/IEC 12119:2003. Інформаційні технології. Пакети програм тестування і вимоги до якості. К.: Держстандарт України, 2003.
5. ДСТУ ISO/IEC 14764:2002. Інформаційні технології. Супроводження програмного забезпечення. К.: Держстандарт України, 2002.
6. ДСТУ ISO/IEC 90003:2006. Програмна інженерія. Настанови щодо застосування ISO 9001:2000 до програмного забезпечення (ISO/IEC 90003:2004, IDT) К.: Держстандарт України, 2006.
7. ДСТУ ISO/IEC TR 12182:2004. Інформаційні технології. Класифікація програмних засобів (ISO/IEC TR 12182:1998, IDT) К.: Держстандарт України, 2004.
8. ДСТУ ISO/IEC 14598-1:2004. Інформаційні технології. Оцінювання програмного продукту. Частина 1. Загальний огляд (ISO/IEC 14598-1:1999, IDT) К.: Держстандарт України, 2004.
9. ДСТУ ISO/IEC 15288:2005. Інформаційні технології. Процеси життєвого циклу системи (ISO/IEC 15288:2002, IDT) - К.: Держстандарт України, 2005.
10. ДСТУ ISO/IEC 15939:2008. Інженерія систем і програмних засобів. Процес вимірювання. К.: Держстандарт України, 2008.
11. ДСТУ 3327-96. Методика випробування процесорів мов програмування. Загальні вимоги. К.: Держстандарт України, 1996.

12. ДСТУ ISO/IEC TR 14369:2003. Інформаційні технології. Мови програмування, їхнє середовище та системний інтерфейс. Настанова щодо підготовки незалежних від мов специфікацій послуг. К.: Держстандарт України, 2003.
13. ДСТУ 4072:2001. Інформаційні технології. Мови програмування, їхнє середовище та системний інтерфейс. Настанова щодо підготовки незалежних від мов виклик процедур. К.: Держстандарт України, 2001.
14. ДСТУ ISO/IEC 2382-15:2005. Інформаційні технології. Словник термінів. Мови програмування (ISO/IEC 2382-15:1999, IDT) - К.: Держстандарт України, 2005.
20.ДСТУ ГОСТ 2.104:2006. ЕСКД. Основні написи. К.: Держстандарт України, 2006. ДОДАТКОВА
22. Bohm, Corrado; and Giuseppe Jacopini (May 1966). "Flow Diagrams, Turing Machines and Languages with Only Two Formation Rules". Communications of the ACM 9 (5): 366–371. doi:10.1145/355592.365646
23. Dijkstra, E. W. (Aug 1972). "The Humble Programmer". Communications of the ACM 15 (10): 859–866. doi:10.1145/355604.361591. http://www.cs.utexas.edu/~EWD/transcriptions/EWD03xx/EWD340.html. (E WD340) PDF, 1972 ACM Turing Award lecture
24. Dijkstra, E.W., "Structured Programming," Software Engineering Techniques, Buxton, J.N., and Randell, B., eds. Brussels, Belgium, NATO Science Committee, 1969.
25. B. Meyer, Object-Oriented Software Construction, second ed., Prentice Hall, 1997, Chap. 6, 10, 11.
26. Guide to the Software Engineering Body of Knowledge (SWEBOK). CHAPTER 4. SOFTWARE CONSTRUCTION. http://www.computer.org/portal/web/swebok/html/ch4K. Beck, Test-Driven Development: By Example, Addison-Wesley, 2002.
27. McCabe : Complexity Measure, IEEE Transacions on Software Engineering, Volume 2, No 4, pp 308-320, December 1976
28. M. Fowler and al., Refactoring: Improving the Design of Existing Code, Addison-Wesley, 2002.
29. Russell Gold, Thomas Hammell, Tom Snyder. Test Driven Development: A J2EE Example. Apress, 2005. 296 pages.

Політика оцінювання

Політика щодо дедлайнів та перескладання: Роботи, які здаються із порушенням термінів без поважних причин, оцінюються на нижчу оцінку (75% від можливої максимальної кількості балів за вид діяльності балів). Перескладання модулів відбувається за наявності поважних причин (наприклад, лікарняний).

Політика щодо академічної доброчесності: Списування під час контрольних робіт заборонені (в т.ч. із використанням мобільних девайсів). Мобільні пристрої дозволяється використовувати лише під час он-лайн тестування та підготовки практичних завдань під час заняття.

Політика щодо відвідування: Відвідування занять є обов’язковим компонентом оцінювання. За об’єктивних причин (наприклад, хвороба, працевлаштування, міжнародне стажування) навчання може відбуватись в он-лайн формі за погодженням із ведучим викладачем курсу.

Оцінювання

Остаточна оцінка за курс розраховується наступним чином: поточний контроль оцінюється в 50 балів, та складається із двох модулів по 25 балів кожен. В суму балів кожного модуля входять бали за підготовку, виконання та захисту 10 практичних робіт по 4 бали за кожну роботу (10 х 4 = 40) та 1 бал за самостійну роботу, яка оцінюється усна компонента під час здачі модуля (співбесіда із лектором) (10 х 1 = 10).

Поточне тестування та самостійна робота (разом 50 балів)				Підсумковий контроль	Сума
Модуль 1 (25 балів)		**Модуль 2** (25 балів)		екзамен	
П1- П5	СР	П6- П10	СР		
5 х 4 =20	5	5 х 4 =20	5	**50**	**100**

П1, П2 ... П10 – практичні роботи; СР – самостійна робота.

До Силабусу також готуються матеріали навчально-методичного комплексу:

1) навчальний контент (розширений план лекцій)
2) тематика та зміст практичних робіт
3) завдання для підсумкової роботи, питання на іспит
4) електронне навчання у ВНС ЛНУП MODLE.