

Силабус курсу
Екстремальне програмування

Освітній ступінь – магістр
Галузь знань: 01 Освіта/Педагогіка
Спеціальність: 015 Професійна освіта
Спеціалізація: 015.39 Цифрові технології
Освітньо-наукова програма «Професійна освіта (Комп'ютерні технології)»
Кількість кредитів – 5
Рік підготовки, семестр – 1,5 та 2 рік, 1 семестр
Компонент освітньої програми: вибірковий, професійна підготовка
Дні занять: за розкладом, ауд. 233
Консультації: за розкладом, ауд. 233

Мова викладання: українська



Керівник курсу

	канд. пед. наук, доцент Сіткарь Тарас Вікторович
Контактна інформація	sitkar@tnpu.edu.ua ; 0969415876

Опис дисципліни

Мета дисципліни «Екстремальне програмування» полягає у вивченні студентами головних принципів підбору персоналу, організації роботи, розподілу функцій та написання технічних завдань для створення сучасного та конкурентоздатного програмного продукту.

Завданням дисципліни є навчити студентів правилам написання правильного коду та якісного здійснення рефакторингу, а також методиці екстремального програмування при створенні програмних продуктів.

По закінченню курсу студент повинен:

знати:

- Основні принципи екстремального програмування
- Основні шаблони екстремального програмування

вміти:

- Варіювати правилами написання правильного коду
- Якісно здійснювати рефакторинг коду
- Тестувати програмні продукти у період їх розробки
- Застосовувати методику екстремального програмування при створенні програмних продуктів

продуктів

Курс передбачає лекційні, практичні та лабораторні аудиторні заняття, а також самостійну роботу студента за межами навчального закладу.

Загальний обсяг дисципліни – 150 годин (5 кредити ЕКТС)

З них: 20 год лекції, 40 год практичні та лабораторні заняття, 90 год самостійна робота.

Дана дисципліна є вибірковою для вивчення.

Структура курсу

Години (лек. / практ.)	Тема	Результати навчання	Завдання
	Змістовий модуль 1. Основи Архітектури та проектування програмного забезпечення		
1/0	Тема 1. Місце КПЗ в життєвому циклі програмної системи	Складові частини розробки ПЗ. Місце конструювання при побудові ПЗ. Область знань "Екстремальне програмування". Задачі, що виникають в процесі розробки ПЗ, пов'язані з конструюванням.	Питання, кейси, ІНДЗ
1/0	Тема 2. Фундаментальні складові екстремального програмування	Мінімізація складності. Очікування змін. Конструювання з можливістю перевірки. Стандарти у конструюванні.	Питання, кейси, ІНДЗ
1/2	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.	Питання, кейси, ІНДЗ
1/2	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.	Питання, кейси, ІНДЗ
1/2	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	Питання, кейси, ІНДЗ
1/2	Тема 6. Стандарти у конструюванні	Комунікаційні методи. Мови програмування і відповідні стилі кодування. Платформи. Інструменти.	Питання, кейси, ІНДЗ
1/2	Тема 7. Високоякісне кодування	Можливість приховування реалізації. Більш висока	Питання, кейси, ІНДЗ

		інформативність інтерфейсу. Легкість оптимізації коду. Легкість читання і зрозумілості коду. Обмеження області використання даних рамками одного класу. Можливість роботи з сутностями реального світу, а не низькорівневими деталями реалізації.	
1/2	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 інформації в інші класи. Розгляд абстракції і зв'язності разом.	Питання, кейси, ІНДЗ
1/2	Тема 9. Принципи використання змінних	Грамотне оголошення змінних. Принципи ініціалізації змінних. Одиначність мети кожної змінної. Принципи вибору імен змінних.	Питання, кейси, ІНДЗ
1/2	Тема 10. Структурне програмування	Суть структурного програмування. Призначення структурного програмування. Керуючі структури. Складність керуючої логіки.	Питання, кейси, ІНДЗ
Змістовий модуль 2. Удосконалення програмного забезпечення			
1/2	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.	Питання, кейси, ІНДЗ
1/2	Тема 12. Еволюція програми	Фактори еволюції. Покращення якості коду у ході еволюції. Правила внесення змін на тій чи іншій стадії еволюції.	Питання, кейси, ІНДЗ
1/2	Тема 13. Поняття рефакторингу	Основні правила еволюції програмного коду. Правила Мартіна Фаулера.	Питання, кейси, ІНДЗ
1/2	Тема 14. Ознаки необхідності застосування рефакторингу	Дублювання коду. Покращення занадто довгого коду. Покращення некоректних імен методів. Покращення недостатнього рівня абстракції.	Питання, кейси, ІНДЗ
1/2	Тема 15. Рівні рефакторингу	Рефакторинги рівня даних. Рефакторинги рівня операторів. Рефакторинги рівня методів. Рефакторинги рівня реалізації класу. Рефакторинги рівня	Питання, кейси, ІНДЗ

		інтерфейсу класу. Рефакторинги рівня системи	
1/2	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.	Питання, кейси, ІНДЗ
1/2	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.	Питання, кейси, ІНДЗ
1/2	Тема 18. Якість конструювання	Тестування коду розробником. TDD (Test-Driven Development). Переваги, які надає TDD. Фреймворк JUnit.	Питання, кейси, ІНДЗ
1/2	Тема 19. Тестування коду розробником	Unit-тестування. Компонентне тестування. Інтеграційне тестування. Регресійне тестування. Системне тестування.	Питання, кейси, ІНДЗ
1/2	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	Питання, кейси, ІНДЗ

		рефакторингу коду, для покращення якості. Тестування після кожної зміни.	
1/2	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.	Питання, кейси, ІНДЗ
1/2	Тема 22. Фреймворк JUnit	Анотації. Методи. Приклади застосування методів.	Питання, кейси, ІНДЗ

Формування програмних компетентностей

Індекс в матриці ОП	Програмні компетентності
ЗК 11	Здатність до розробки й застосування програмного забезпечення виробничого або освітнього процесів.
ФК 12	Здатність до розробки, тестування програмного забезпечення, адміністрування і налаштування інформаційних систем та їх інтеграції у науково-педагогічні дослідження.
ПРН 2	Ефективно використовувати сучасні цифрові інструменти, інформаційні технології та ресурси у професійній, інноваційній та/або дослідницькій діяльності.
ПРН 14	Вміти розробляти вимоги та специфікації компонентів інформаційних систем, проєктувати та імплементувати компоненти програмного забезпечення, людино-машинний інтерфейс інформаційних систем, інтегрувати їх компоненти у навчальну та науково-дослідну діяльність.

Літературні джерела

ОСНОВНІ

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. Частина 15. Мови програмування (ISO/IEC 2382-15:1999, IDT) - К.: Держстандарт України, 2005.
15. ДСТУ 3008-95. "Документація. Звіти у сфері науки і техніки Структура і правила оформлення". К.: Держстандарт України, 1995. – 75 с.
16. ГОСТ 2.106-96. Единая система конструкторской документации. Текстовые документы. Изд. Офиц – К.: Госстандарт Украины, 1998. – 47 с.
17. ГОСТ 2.109-73 ЕСКД. Основные требования к чертежам – М., 1978.
18. ГОСТ 2.105-95. Единая система конструкторской документации. Общие требования к текстовым документам. Изд. Офиц – К.: Госстандарт Украины, 1996.
19. ДСТУ ГОСТ 7.1:2006. Система стандартів з інформації, бібліотечної та видавничої справ. Загальні вимоги та правила складання. - К.: Держстандарт України, 2007. – 47 с.
20. ДСТУ ГОСТ 2.104:2006. ЕСКД. Основні написи. - К.: Держстандарт України, 2006.

ДОДАТКОВІ

21. Макконнелл С. Совершенный код. Мастер-класс / Пер. с англ.- М.: Издательско-торговый дом "Русская Редакция"; СПб.: Питер, 2005.- 896 стр.: ил.
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>. (EWD340) 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 Transactions 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 із дозволу навчальної частини за наявності поважних причин (наприклад, лікарняний).
- **Політика щодо академічної доброчесності:** Списування під час контрольних робіт та екзаменів заборонені (в т.ч. із використанням мобільних девайсів). Мобільні пристрої дозволяється використовувати лише під час он-лайн тестування та підготовки практичних завдань в процесі заняття.
- **Політика щодо відвідування:** Відвідування занять є обов'язковим компонентом оцінювання, за яке нараховуються бали. За об'єктивних причин (наприклад, хвороба, працевлаштування, міжнародне стажування) навчання може відбуватись в он-лайн формі за погодженням із керівником курсу.

Оцінювання

Остаточна оцінка за курс розраховується наступним чином:

Види оцінювання		% від остаточної оцінки
Модуль 1	усне опитування, тести, завдання	30
Модуль 2	усне опитування, тести, завдання	30
ІНДЗ		10
Підсумковий контроль – тести		30

Шкала оцінювання студентів:

ECTS	Бали	Зміст
A	90-100	відмінно
B	85-89	добре
C	75-84	добре
D	65-74	задовільно
E	60-64	достатньо
FX	35-59	незадовільно з можливістю повторного складання
F	1-34	незадовільно з обов'язковим повторним курсом