МІЖРЕГІОНАЛЬНА
АКАДЕМІЯ УПРАВЛІННЯ ПЕРСОНАЛОМ

Навчальна програма
дисципліни

«Якість програмного забезпечення та тестування комп»ютерних систем і мереж»

(для бакалаврів)

Київ 2014

Дуднік А.С. Програма вивчення дисципліни. «*Якість програмного забезпечення та тестування комп»ютерних систем і мереж*». – К.: МАУП, 2014. – 12 с.

Підготовлено доцентом кафедри комп'ютерних інформаційних систем і техгологій Дудніком А.С.

Затверджено на засіданні кафедри комп'ютерних інформаційних систем і техгологій (Протокол № 2 від 16.09. 2014 р.).

Схвалено вченою радою *Міжрегіональної академії управління персоналом*.

Дуднік А.С. Навчальна програма дисципліни «*Якість програмного забезпечення та тестування комп»ютерних систем і мереж*» (для освітньо-кваліфікаційного рівня «спеціаліст»). – К.: МАУП, 2014. – 12 с.

Навчальна програма містить пояснювальну записку, тематичний план, зміст дисципліни «*Якість програмного забезпечення та тестування комп»ютерних систем і мереж»*, а також список літератури.

ПОЯСНЮВАЛЬНА ЗАПИСКА

Сучасні інформаційні технології передбачають широке використання новітніх підходів до організації проектів створення програмних продуктів, як основного засобу менеджменту програмних проектів.

Мета дисципліни *«Якість програмного забезпечення та тестування комп»ютерних систем і мереж»* полягає у вивченні студентами напрямку «Комп'ютерні науки» головних принципів підбору персоналу, організації роботи, розподілу функцій та написання технічних завдань для створення сучасного та конкурентоздатного програмного продукту.

Програма розрахована на спеціалістів з напрямку «Комп'ютерні науки», які володіють, в рамках відповідних навчальних курсів, знаннями з дисциплін: «Основи програмування та алгоритмічні мови», «Об'єктно-орієнтовне програмування та організація баз даних та знань».

Для вивченні конкретних навчальних тем необхідно використовувати рекомендовану літературу з поданого у програмі списку.

Заключна перевірка знань студентів передбачена у вигляді заліку.

ТЕМАТИЧНИЙ ПЛАН
дисципліни «Якість програмного забезпечення та тестування комп»ютерних систем і мереж»

№	**Назва розділу, теми курсу**
	Змістовий модуль 1. Основи Архітектури та проектування програмного забезпечення
1.	Місце КПЗ в життєвому циклі програмної системи
2.	Фундаментальні складові Якість програмного забезпечення та тестування комп»ютерних систем і мереж
3.	Мінімізація складності
4.	Очікування змін
5.	Конструювання з можливістю перевірки
6.	Стандарти у конструюванні
7.	Високоякісне кодування
8.	Правила написання якісного коду. Рівень класів
9.	Принципи використання змінних
10.	Структурне програмування
	Змістовий модуль 2. Удосконалення програмного забезпечення
11.	Рефакторинг
12.	Еволюція програми
13.	Поняття рефакторингу
14.	Ознаки необхідності застосування рефакторингу
15.	Рівні рефакторингу
16.	Безпечний рефакторинг
17.	Стратегії рефакторингу
18.	Якість конструювання
19.	Тестування коду розробником
20.	TDD (Test-Driven Development)
21.	Переваги, які надає TDD
22.	Фреймворк JUnit

ЗМІСТ

ДИСЦИПЛІНИ «ЯКІСТЬ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ТА ТЕСТУВАННЯ КОМП»ЮТЕРНИХ СИСТЕМ І МЕРЕЖ»

Змістовий модуль 1. Основи Архітектури та проектування програмного забезпечення

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.

Література: [2], [4], [8], [12].

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.

Література: [2], [4], [6], [10].

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.

Література: [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.

Література: [4]

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

Література: [6]

Тема 6. Стандарти у конструюванні

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

Література: [7], [9], [11]

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.

Література: [21]

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.

Література: [10], [12], [21].

Тема 9. Принципи використання змінних

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

Література: [12], [14], [23].

Тема 10. Структурне програмування

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

Література: [13], [15], [24].

Змістовий модуль 2. Удосконалення програмного забезпечення

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.

Література: [5], [10], [27].

Тема 12. Еволюція програми

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

Література: [5], [10], [27].

Тема 13. Поняття рефакторингу

Основні правила еволюції програмного коду. Правила Мартіна Фаулера.

Література: [5], [10], [27].

Тема 14. Ознаки необхідності застосування рефакторингу

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

Література: [5], [10], [27].

Тема 15. Рівні рефакторингу

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

Література: [5], [10], [27].

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.

Література: [5], [10], [27].

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.

Література: [5], [10], [27].

Тема 18. Якість конструювання

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

Література: [21].

Тема 19. Тестування коду розробником

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

Література: [21].

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.

Література: [21].

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.

Література: [21].

Тема 22. Фреймворк JUnit

Анотації. Методи. Приклади застосування методів.

Література: [21].

ПИТАННЯ ДЛЯ САМОКОНТРОЛЮ

1. Що таке життевий цикл програмного забезпечення.
2. Які різновиди моделей життевого циклу програмного забезпечення Вам відомі.
3. Основні характеристики і фази водоспадної моделі.
4. Основні характеристики і фази ітеративної моделі.
5. Основні характеристики і фази спіральної моделі.
6. Основні аспекти планування проекту програмного забезпечення.
7. Основні характеристики планування процесу розробки.
8. Основні характеристики визначення результатів проекту.
9. Основні характеристики оцінки зусиль, розкладу та вартісних очикувань проекту.
10. Основні характеристики розподілу ресурсів проекту.
11. Яким чином визначається критичний шлях проекту.
12. Основні характеристики керування ризиками проекту.
13. Основні типи мов конструювання програмних проектів.
14. Основні можливості конфігураційних мов конструювання.
15. Основні можливості інструментальних мов конструювання.
16. Основні можливості мов програмування.
17. Назвіть методи інтеграції.
18. Назвіть основні характеристики методів інтеграції.
19. Назвіть основні засоби безперервної інтеграції.
20. Що таке побудова, перевірка, розгортання?
21. Що таке зворотній зв'язок?
22. Цілі тестування програмного забезпечення.
23. Цілі тестування програмного забезпечення.
24. Охарактеризуйте техніки що базуються на інтуїції та досвіді інженера.
25. Охарактеризуйте техніки що базуються на специфікаціях.
26. Охарактеризуйте техніки що орієнтовані на код.
27. Охарактеризуйте техніки що орієнтовані на дефекті.
28. Охарактеризуйте техніки що базуються на умовах використання.
29. Охарактеризуйте техніки що базуються на природі застосування.
30. Методи оцінки результатів тестування.
31. Перелічить основні техніки відлагодження програмного забезпечення.
32. Які інструменти відлагодження Вам відомі та які в них переваги та недоліки.
33. Основні типи шаблонів проектування та їх призначення.
34. Фундаментальні шаблони, їх склад та призначення.

35.Опишіть можливості шаблону Незмінний об'єкт.
36.Опишіть можливості шаблону Інтерфейс.
37.Породжуючі шаблони, їх склад та призначення.
38.Опишіть можливості шаблону Абстрактна фабрика
39.Опишіть можливості шаблону Будівник.
40.Опишіть можливості шаблону Фабричний метод.
41.Опишіть можливості шаблону Клас-одинак.
42.Опишіть можливості шаблону Відстрочена ініціалізація.
43.Опишіть можливості шаблону Прототип.
44.Структурні шаблони, їх склад та призначення.
45.Опишіть можливості шаблону Адаптер.
46.Опишіть можливості шаблону Міст.
47.Опишіть можливості шаблону Компоновник.
48.Опишіть можливості шаблону Декоратор.
49.Опишіть можливості шаблону Фасад.
50.Опишіть можливості шаблону Пристосуванець.
51.Опишіть можливості шаблону Заступник.
52.Поведінкові шаблони, їх склад та призначення.
53.Опишіть можливості шаблону Команда.
54.Опишіть можливості шаблону Інтерпретатор.
55.Опишіть можливості шаблону Ітератор.
56.Опишіть можливості шаблону Посередник.
57.Опишіть можливості шаблону Наглядач.

СПИСОК ЛІТЕРАТУРИ

ОСНОВНА

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 справі. Загальні вимоги та правіла складання. - К.: Держстандарт України, 2007. – 47 с.
20. ДСТУ ГОСТ 2.104:2006. ЕСКД. Основні написи. - К.: Держстандарт України, 2006.

ДОДАТКОВА

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 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.