МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
МІЖНАРОДНИЙ ГУМАНІТАРНИЙ УНІВЕРСИТЕТ
Кафедра інформаційних технологій

Стрелковська І.В., Приходько С.Б., Григор'єва Т.І.

РЕІНЖИНІРІНГ ТА ОПТИМІЗАЦІЯ ПРОГРАМНИХ СИСТЕМ

Методичні рекомендації для самостійної роботи здобувачів

Одеса 2023

Затверджено Вченою Радою Міжнародного гуманітарного університету (протокол № 1 від 4 вересня 2023 р.)

Стрелковська І.В., Приходько С.Б., Григор'єва Т.І.

Реінжинірінг та оптимізація програмних систем: методичні рекомендації для самостійної роботи здобувачів [Електронне видання]. / Стрелковська І.В., Приходько С.Б., Григор'єва Т.І. Кафедра інформаційних технологій Міжнародного гуманітарного університету. Одеса, 2023. – 25 с.

Методичні рекомендації з курсу «Реінжинірінг та оптимізація програмних систем» розроблено відповідно до навчального плану, вони складаються з навчальної програми курсу, методичних рекомендацій з проведення практичних занять і завдань для самостійної роботи здобувачів, списку рекомендованої літератури. Матеріали призначено для студентів факультету кібербезпеки, програмної інженерії та комп'ютерних наук Міжнародного гуманітарного університету, які в магістратурі вивчають галузь знань - «Інформаційні технології».

Вивчення дисципліни «Реінжинірінг та оптимізація програмних систем» допоможе студентам зрозуміти принципи та методи реінжинірінгу програмних систем, інструменти та технології, що використовуються для аналізу, оцінки та покращення програмних систем, процеси оптимізації програмних систем з метою підвищення їх продуктивності, швидкості та надійності, а також сприятиме залученню здобувачів до науково-дослідницької діяльності та підготовки ними публікацій, кваліфікаційних й інших наукових робіт.

1. ОПИС НАВЧАЛЬНОЇ ДИСЦИПЛІНИ

<table>
<tr><td rowspan="2">Найменування показників</td><td rowspan="2">Галузь знань, напрям підготовки, освітньо-кваліфікаційний рівень</td><td>Характеристика навчальної дисципліни</td></tr>
<tr><td>денна форма навчання</td></tr>
<tr><td rowspan="5">Кількість кредитів – 6

Загальна кількість годин – 180</td><td>Галузь - 12 «Інформаційні технології»</td><td>обов’язкова</td></tr>
<tr><td rowspan="4">Спеціальність –
121 «Інженерія програмного забезпечення»</td><td>Рік підготовки:</td></tr>
<tr><td>1-й</td></tr>
<tr><td>Семестр</td></tr>
<tr><td>1-й</td></tr>
<tr><td rowspan="11">Мова навчання – українська</td><td rowspan="11">Рівень вищої освіти –
другий (магістерський) рівень</td><td>Лекції</td></tr>
<tr><td>42 год.</td></tr>
<tr><td>Практичні, семінарські</td></tr>
<tr><td>28 год.</td></tr>
<tr><td>Лабораторні</td></tr>
<tr><td>-</td></tr>
<tr><td>Самостійна робота та індивідуальні завдання</td></tr>
<tr><td>110 год.</td></tr>
<tr><td>Вид контролю:</td></tr>
<tr><td>екзамен</td></tr>
</table>

Вивчення дисципліни «**Реінжинірінг та оптимізація програмних систем**» дозволяє сформувати у здобувачів освіти компетенції, необхідні для розв’язання практичних задач наукової діяльності, пов’язаної із роботою з існуючим програмним забезпеченням, його вдосконаленням та масштабуванням для задоволення сучасних потреб супроводження такого програмного забезпечення.

Предметом дисципліни «**Реінжинірінг та оптимізація програмних систем**» є математичне та алгоритмічне забезпечення процесів переробки вихідного коду.

Метою навчальної дисципліни є формування у здобувачів вищої освіти таких компетентностей:

- Здатність розробляти якісне та надійне програмне забезпечення складних програмних комплексів та систем на основі новітніх технологій та стандартів розроблення програмного забезпечення.
- Здатність проводити експериментальні дослідження з оцінювання ефективності та безпечності програмного забезпечення.
- Здатність самостійно виконувати науково-дослідну діяльність в інженерії програмного забезпечення із застосуванням сучасних концепцій, методів, та технологій.
- Здатність критично переосмислювати наявні технології інженерії програмного забезпечення та відстежувати тенденції їх розвитку.

Отримані в результаті засвоєння дисципліни «Реінжинірінг та оптимізація програмних систем» теоретичні знання та практичні уміння є корисними для проведення наукових досліджень за темою дисертації.

Заплановані результати навчання за навчальною дисципліною

Знання:

- методи реінжинірингу програмного забезпечення.

− методи поетапного модифікування програмного забезпечення.

− концептуальні та методологічні знання з інженерії програмного забезпечення і на межі предметних галузей, а також дослідницькі навички, достатні для проведення наукових і прикладних досліджень на рівні сучасних світових досягнень з відповідного напряму, отримання нових знань та/або

здійснення інновацій.

− загальні принципи та методи інженерії програмного забезпечення, а також методологію наукових досліджень, застосувати їх у власних дослідженнях у сфері інженерії програмного забезпечення та у викладацькій практиці.

Уміння:

- досліджувати робочі параметри процесів життєвого циклу програмного забезпечення, а також здійснювати аналіз вибраних методів та засобів підтримки цих процесів та бути спроможним обґрунтувати свій вибір.

- застосовувати методи поетапного модифікування програмного забезпечення.

− формулювати та вирішувати задачі оптимізації, адаптації, прогнозування, керування та прийняття рішень щодо процесів, засобів та ресурсів розроблення, впровадження, супроводу та експлуатації програмного забезпечення.виконувати інформаційний пошук, накопичування та обробляти наукову інформацію.

2. ПРОГРАМА НАВЧАЛЬНОЇ ДИСЦИПЛІНИ

Тема 1. Методологія реінжинірингу.

Поняття рефакторінгу, чистого та брудного коду, запаху коду. Актуальні завдання у проблематиці реінжинірингу. Оцінювання запаху коду. Уніфікація і стандартизація поняття «реінжиніринг ІС». Розробка цілісних методологій реінжинірингу ІС. Рівень методології й рівень технології реінжинірингу ІС. Огляд різних методик модернізації успадкованих систем: заміщення, модернізація, еволюція. Моделі модернізації систем. Каркасна модель. Технологія KADS для проектування ІС на основі шаблонів. Реінжиніринг ПЗ у проектуванні Веб-додатків. Технології CMS, CMF, Joomla, Wordpress. Реінжиніринг ПЗ на основі методу побудови оболонок для компонентів

успадкованої системи. Використання технології CORBA. Реінжиніринг ПЗ на основі методів інтеграції з використанням CGI та на основі технології XML.

Тема 2. Дослідження й розробка шаблонів.

Патерни (шаблони), переваги та недоліки їх використання. Історія створення патернів. Класифікація патернів. Породжуючі патерни. Структурні патерни. Поведінкові патерни. Патерни проектування об’єктів інформаційних систем. Архітектурні і системні патерни. Патерни інтеграції інформаційних систем.

Тема 3. Технології та засоби реінжинірингу.

Засоби реінжинірингу в мовах програмування. Темплейти і суперкласи в C++, C#. Призначення і функції суперкласів. Шаблони класів. Реінжиніринг ПЗ на основі методу реплікації баз даних з використанням об’єктно-орієнтованого підходу. Уніфіковані програмні засоби для задач реінжинірингу. Макроси, бібліотеки, репозитарії, сховища даних, оболонки. Технологія NET Framework. Характеристики і функції. Тенденції і напрямки розвитку технологій реінжинірингу. Можливості проблемно-орієнтованих технологій. Хмарні технології для реінжтнірингу. Задачі реінжинірингу бізнес-процесів. Моделі бізнес-процесів. Технології реінжинірингу бізнес-процесів BI та BAM. Технології функціонального моделювання бізнес-процесів - IDEF0, IDEF3. Використання діаграм DFD для опису потоків даних. Засоби Bpwin та Erwin для задач реінжинірингу бізнес-процесів. Архітектура і характеристики. Приклади застосування.

Тема 4. Оптимізація програмних систем.

Оптимізація як бажана ціль у розробці ПЗ. Пошук компромісу між оптимізацією та стабільністю, зручністю обслуговування та переносимістю ПЗ. Загальна постановка задачи оптимізації коду. Продуктивність програми. Артихектура коду та її вплив на продуктивність програми. Підвищення

продуктивності програми за рахунок застосування мов низького рівня. Приклади хороших прийомів оптимізації коду. Усунення розгалуження як важлива проблема для сучасних глибоко конвеєрних архітектур процесорів. Оптимізація при виконанні циклу. Оптимізація при використанні масивів.

3. СТРУКТУРА НАВЧАЛЬНОЇ ДИСЦИПЛІНИ

Назви змістових модулів і тем	**Кількість годин**			
	денна форма			
	усього	**у тому числі**		
		лекц.	**практ.**	**сам. роб.**
Тема 1. Методологія реінжинірингу.	46	10	6	30
Тема 2. Дослідження й розробка шаблонів реінжинірингу.	34	8	6	20
Тема 3. Технології та засоби реінжинірингу.	50	12	8	30
Тема 4. Оптимізація програмних систем.	50	12	8	30
Усього годин	**180**	**42**	**28**	**110**
ПІДСУМКОВИЙ КОНТРОЛЬ - ЕКЗАМЕН				

4. ПЛАН – КОНСПЕКТ ЛЕКЦІЙ ДИСЦИПЛІНИ

«РЕІНЖИНІРІНГ ТА ОПТИМІЗАЦІЯ ПРОГРАМНИХ СИСТЕМ»

Тема 1. Методологія реінжинірингу

1.1. Вступ до реінжинірингу та оптимізації програмних систем.

Визначення реінжинірингу: поняття, цілі та об'єктиви.

Роль оптимізації у процесі реінжинірингу: поліпшення продуктивності, якості та вартості програмних систем.

Рефакторінг: визначення, підходи до модифікації коду для поліпшення структури та зрозумілості.

Поняття "чистого коду" та "брудного коду": основні принципи чистого коду та наслідки використання брудного коду.

Виявлення запаху коду: розпізнавання ознак поганої архітектури та дизайну програми.

1.2. Методології реінжинірингу програмних систем.

Уніфікація поняття "реінжинінг ІС": визначення загальних аспектів та завдань цієї області.

Розробка цілісних методологій реінжинірингу ІС: інтеграція аспектів аналізу, моделювання, перетворення та реалізації.

Рівень методології та рівень технології реінжинірингу ІС: від концептуальних підходів до конкретних реалізацій.

Огляд методик модернізації успадкованих систем: підхід "заміщення", модернізація коду, еволюція системи.

Каркасна модель: огляд засобів та підходів для побудови нових систем на основі успадкованих.

1.3. Технології реінжинірингу ПЗ.

Роль технологій у реінжинінгу ПЗ: автоматизація, підтримка прийняття рішень та аналізу.

Використання технологій у реінжинінгу Веб-додатків: CMS (Content Management Systems), CMF (Content Management Frameworks) - призначення, функції, вигоди.

Використання технології CORBA у реінжинінгу: заміна застарілих компонентів, підвищення інтегрованості та швидкодії.

Використання CGI (Common Gateway Interface) та технології XML у реінжинінгу ПЗ: взаємодія з сервером, обробка даних та удосконалення інтерфейсу.

Тема 2. Дослідження й розробка шаблонів.

2.1. Патерни (шаблони) у реінжинінгу.

Класифікація патернів: породжуючі, структурні та поведінкові патерни.

Породжуючі патерни: Factory Method, Abstract Factory, Singleton.

Структурні патерни: Adapter, Bridge, Composite, Decorator, Proxy.

Поведінкові патерни: Chain of Responsibility, Command, Observer, Strategy, Template Method.

2.2. Використання патернів у реінжинінгу.

Роль патернів у поліпшенні структури та функціональності систем.

Використання породжуючих патернів для створення об'єктів та забезпечення їхньої уніфікації.

Застосування структурних патернів для розширення функціональності та забезпечення гнучкості коду.

Використання поведінкових патернів для забезпечення ефективної взаємодії об'єктів та контролю поведінки системи.

Тема 3. Технології та засоби реінжинірингу.

3.1. Засоби реінжинірингу та оптимізації в програмуванні.

Використання темплейтів та суперкласів у C++ та C# для поліпшення коду та забезпечення його перевикористовуваності.

Застосування шаблонів класів для гнучкої реалізації різних функцій та взаємодії об'єктів.

3.2. Уніфіковані програмні засоби для реінжинінгу.

Застосування макросів, бібліотек, репозитаріїв, сховищ даних у реінжинінгу: підвищення продуктивності та швидкодії розробки.

Переваги та обмеження використання технології .NET Framework: стандартизація, підтримка, можливості.

3.3. Тенденції та напрямки розвитку реінжинінгу.

Проблемно-орієнтовані технології та їх вплив на реінжинінг: відповідність специфікаціям, підвищена ефективність.

Використання хмарних технологій для реінжинінгу: забезпечення гнучкості, економія ресурсів.

Задачі реінжинінгу бізнес-процесів: визначення, моделювання, оптимізація та впровадження.

Використання моделей бізнес-процесів, діаграм DFD та технологій Bpwin та Erwin для реінжинінгу бізнес-процесів.

Огляд архітектури та функцій хмарних технологій у контексті реінжинінгу.

Тема 4. Оптимізація програмних систем

4.1. Оптимізація як бажана ціль у розробці ПЗ.

Оптимізація як процес поліпшення продуктивності та ефективності програмних систем.

Важливість балансу між оптимізацією та іншими аспектами, такими як стабільність, зручність обслуговування та переносимість ПЗ.

Вплив оптимізації на загальний успіх проекту та співвідношення між витратами та вигодами.

4.2. Загальна постановка задачі оптимізації коду.

Визначення завдання оптимізації коду: зростання продуктивності, зменшення витрат ресурсів, прискорення роботи програми.

Процес виявлення місць для оптимізації та вибір підходів для досягнення цілей.

4.3. Продуктивність програми та архітектура коду

Розуміння поняття продуктивності програми: час виконання, використання пам'яті, швидкість реакції.

Вплив архітектури коду на продуктивність програми: зв'язність, залежності, оптимізація обчислень та доступу до даних.

4.4. Підвищення продуктивності програми за рахунок застосування мов низького рівня.

Огляд мов низького рівня: асемблер, C/C++.

Вигоди від використання мов низького рівня у контексті оптимізації.

Важливість балансу між продуктивністю та зрозумілістю коду.

4.5. Приклади хороших прийомів оптимізації коду.

Використання кешування даних: локальність, попереднє завантаження.

Паралельна обробка та конкурентність: використання багатопотоковості.

Лінійний алгоритм: уникнення зайвих операцій та залежностей.

4.6. Усунення розгалуження як важлива проблема для сучасних глибоко конвеєрних архітектур процесорів.

Огляд розгалужень у коді: умовні оператори, цикли, рекурсія.

Вплив розгалужень на продуктивність та використання ресурсів процесора.

Застосування векторизації та інших технік для зниження розгалужень.

4.7. Оптимізація при виконанні циклу та використанні масивів.

Важливість оптимізації обробки циклів: зменшення кількості операцій та викликів.

Використання кешування та принципів локальності під час обходу масивів.

Застосування векторизації та паралельних обчислень для оптимізації циклічних операцій.

5. ПИТАННЯ ДО ПРАКТИЧНИХ ЗАНЯТЬ

Тема 1. Методологія реінжинірингу.

Поняття рефакторінгу. Актуальні завдання у проблематиці реінжинірингу. Оцінювання запаху коду. Уніфікація і стандартизація поняття «реінжиніринг ІС». Розробка цілісних методологій реінжинірингу ІС. Рівень методології й рівень технології реінжинірингу ІС. Огляд різних методик модернізації успадкованих систем: заміщення, модернізація, еволюція. Моделі модернізації систем. Каркасна модель. Технологія KADS для проектування ІС на основі шаблонів. Реінжиніринг ПЗ у проектуванні Веб-додатків. Технології CMS, CMF, Joomla, Wordpress. Реінжиніринг ПЗ на основі методу побудови оболонок для компонентів успадкованої системи. Використання технології CORBA. Реінжиніринг ПЗ на основі методів інтеграції з використанням CGI та на основі

технології XML.

Тема 2. Дослідження й розробка шаблонів реінжинірингу.

Патерни (шаблони), переваги та недоліки їх використання. Історія створення патернів. Класифікація патернів. Породжуючі патерни. Структурні патерни. Поведінкові патерни. Патерни проектування об’єктів інформаційних систем. Архітектурні і системні патерни. Патерни інтеграції інформаційних систем.

Тема 3. Технології та засоби реінжинірингу.

Засоби реінжинірингу в мовах програмування. Темплейти і суперкласи в C++, C#. Призначення і функції суперкласів. Шаблони класів. Реінжиніринг ПЗ на основі методу реплікації баз даних з використанням об’єктно-орієнтованого підходу. Уніфіковані програмні засоби для задач реінжинірингу. Макроси, бібліотеки, репозитарії, сховища даних, оболонки. Технологія NET Framework. Характеристики і функції. Тенденції і напрямки розвитку технологій реінжинірингу. Можливості проблемно-орієнтованих технологій. Хмарні технології для реінжтнірингу. Задачі реінжинірингу бізнес-процесів. Моделі бізнес-процесів. Технології реінжинірингу бізнес-процесів BI та BAM. Технології функціонального моделювання бізнес-процесів - IDEF0, IDEF3. Використання діаграм DFD для опису потоків даних. Засоби Bpwin та Erwin для задач реінжинірингу бізнес-процесів.

Тема 4. Оптимізація програмних систем.

Оптимізація як бажана ціль у розробці ПЗ. Пошук компромісу між оптимізацією та стабільністю, зручністю обслуговування та переносимістю ПЗ. Загальна постановка задачи оптимізації коду. Продуктивність програми. Артихектура коду та її вплив на продуктивність програми. Підвищення продуктивності

програми за рахунок застосування мов низького рівня. Приклади хороших прийомів оптимізації коду. Усунення розгалуження як важлива проблема для сучасних глибоко конвеєрних архітектур процесорів. Оптимізація при виконанні циклу. Оптимізація при використанні масивів.

6. САМОСТІЙНА РОБОТА

До самостійної роботи студентів включаються:

1. Знайомство з науковою та навчальною літературою відповідно зазначених у програмі тем.
2. Опрацювання лекційного матеріалу.
3. Підготовка до практичних занять.
4. Консультації з викладачем протягом семестру.
5. Самостійне опрацювання окремих питань навчальної дисципліни.
6. Підготовка до підсумкового контролю.

Тематика та питання до самостійної підготовки та індивідуальних завдань

Тема 1. Методологія реінжинірингу.

Поняття рефакторінгу, чистого та брудного коду, запаху коду. Актуальні завдання у проблематиці реінжинірингу. Оцінювання запаху коду. Уніфікація і стандартизація поняття «реінжиніринг ІС». Розробка цілісних методологій реінжинірингу ІС. Рівень методології й рівень технології реінжинірингу ІС. Огляд різних методик модернізації успадкованих систем: заміщення, модернізація, еволюція. Моделі модернізації систем. Каркасна модель. Технологія KADS для проектування ІС на основі шаблонів. Реінжиніринг ПЗ у проектуванні Веб-додатків.

Тема 2. Дослідження й розробка шаблонів реінжинірингу.

Патерни (шаблони), переваги та недоліки їх використання. Історія створення патернів. Класифікація патернів. Породжуючі патерни. Структурні патерни. Поведінкові патерни. Патерни проектування об’єктів інформаційних систем. Архітектурні і системні патерни. Патерни інтеграції інформаційних систем.

Тема 3. Технології та засоби реінжинірингу.

Засоби реінжинірингу в мовах програмування. Темплейти і суперкласи в C++, C#. Призначення і функції суперкласів. Шаблони класів. Реінжиніринг ПЗ на основі методу реплікації баз даних з використанням об’єктно-орієнтованого підходу. Уніфіковані програмні засоби для задач реінжинірингу. Макроси, бібліотеки, репозитарії, сховища даних, оболонки. Технологія NET Framework. Характеристики і функції. Тенденції і напрямки розвитку технологій реінжинірингу. Можливості проблемно-орієнтованих технологій. Хмарні технології для реінжтнірингу. Задачі реінжинірингу бізнес-процесів. Моделі бізнес-процесів. Технології реінжинірингу бізнес-процесів BI та BAM. Технології функціонального моделювання бізнес-процесів - IDEF0, IDEF3. Використання діаграм DFD для опису потоків даних.

Тема 4. Оптимізація програмних систем.

Оптимізація як бажана ціль у розробці ПЗ. Пошук компромісу між оптимізацією та стабільністю, зручністю обслуговування та переносимістю ПЗ. Загальна постановка задачи оптимізації коду. Продуктивність програми. Артихектура коду та її вплив на продуктивність програми. Підвищення продуктивності програми за рахунок застосування мов низького рівня. Приклади хороших прийомів оптимізації коду. Усунення розгалуження як важлива проблема для сучасних глибоко конвеєрних архітектур процесорів.

Оптимізація при виконанні циклу. Оптимізація при використанні масивів.

7. ВИДИ ТА МЕТОДИ КОНТРОЛЮ

Робоча програма навчальної дисципліни передбачає наступні види та методи контролю:

Види контролю	**Складові оцінювання**
поточний контроль, який здійснюється у ході: проведення практичних занять, виконання індивідуального завдання; проведення консультацій та відпрацювань.	**50%**
підсумковий контроль, який здійснюється у ході проведення іспиту.	**50%**

Методи діагностики знань (контролю)	фронтальне опитування; наукова доповідь, усне повідомлення, індивідуальне опитування, робота у групах, розв`язання задач і практичних завдань, іспит

8. Питання до іспиту

1. Поняття рефакторінгу, чистого та брудного коду, запаху коду.
2. Актуальні завдання у проблематиці реінжинірингу.
3. Уніфікація і стандартизація поняття «реінжиніринг ІС».
4. Розробка цілісних методологій реінжинірингу ІС.
5. Рівень методології й рівень технології реінжинірингу ІС.
6. Оцінювання запаху коду.
7. Огляд різних методик модернізації успадкованих систем: заміщення, модернізація, еволюція. Моделі модернізації систем. Каркасна модель.
8. Технологія KADS для проектування ІС на основі шаблонів. Реінжиніринг ПЗ у проектуванні Веб- додатків.

9. Технології CMS, CMF, Joomla, Wordpress.
10. Реінжиніринг ПЗ на основі методу побудови оболонок для компонентів успадкованої системи.
11. Реінжиніринг ПЗ на основі методів інтеграції з використанням CGI та на основі технології XML. Дослідження й розробка шаблонів реінжинірингу.
12. Патерни. Класифікація патернів. Породжуючі патерни. Структурні патерни. Поведінкові патерни.
13. Патерни проектування об'єктів інформаційних систем.
14. Архітектурні і системні патерни.
15. Патерни інтеграції інформаційних систем.
16. Засоби реінжинірингу в мовах програмування.
17. Темплейти і суперкласи в C++, C#.
18. Призначення і функції суперкласів. Шаблони класів.
19. Реінжиніринг ПЗ на основі методу реплікації баз даних з використанням об'єктно-орієнтованого підходу.
20. Уніфіковані програмні засоби для задач реінжинірингу.
21. Макроси, бібліотеки, репозитарії, сховища даних, оболонки.
22. Задачі реінжинірингу бізнес-процесів.
23. Моделі бізнес-процесів. Технології реінжинірингу бізнес-процесів BI та BAM. Технології функціонального моделювання бізнес-процесів-IDEF0, IDEF3.
24. Використання діаграм DFD для опису потоків даних.
25. Засоби Bpwin та Erwin для задач реінжинірингу бізнес-процесів. Архітектура і характеристики. Приклади застосування.
26. Застосування компонентних технологій для вирішення задач реінжинірингу.
27. Технологія NET Framework. Характеристики і функції.
28. Тенденції і напрямки розвитку технологій реінжинірингу.
29. Можливості проблемно-орієнтованих технологій.

30. Хмарні технології для реінжтніринґу.
31. Оптимізація як бажана ціль у розробці ПЗ.
32. Пошук компромісу між оптимізацією та стабільністю, зручністю обслуговування та переносимістю ПЗ.
33. Загальна постановка задачи оптимізації коду. Продуктивність програми.
34. Артихектура коду та її вплив на продуктивність програми.
35. Підвищення продуктивності програми за рахунок застосування мов низького рівня.
36. Усунення розгалуження як важлива проблема для сучасних глибоко конвеєрних архітектур процесорів.
37. Оптимізація при виконанні циклу.
38. Оптимізація при використанні масивів.
39. Проєктування архітектури програмного забезпечення. Моделювання процесів функціонування окремих підсистем і модулів.
40. Розвиток і реалізація нових конкурентоспроможних ідей в інженерії програмного забезпечення.
41. Розробка моделей та засобів інтелектуальної обробки даних в розподілених системах. Оптимізація програмного забезпечення з урахуванням вимог до їх надійності.
42. Як розробляти і оцінювати стратегії проєктування програмних засобів; обґрунтовувати, аналізувати і оцінювати варіанти проєктних рішень з точки зору якості кінцевого програмного продукту, ресурсних обмежень та інших факторів.
43. Як розробляти і модифікувати архітектуру програмного забезпечення для реалізації вимог замовника.
44. Як модифікувати існуючі та розробляти нові алгоритмічні рішення детального проєктування програмного забезпечення.
45. Як забезпечувати якість на всіх стадіях життєвого циклу програмного забезпечення, у тому числі з використанням релевантних моделей та

методів оцінювання, а також засобів автоматизованого тестування і верифікації програмного забезпечення.

46. Як приймати ефективні організаційно-управлінські рішення в умовах невизначеності та зміни вимог, порівнювати альтернативи, оцінювати ризики.
47. Як конфігурувати програмне забезпечення, керувати його змінами та розробленням програмної документації на всіх етапах життєвого циклу.
48. Як здійснювати реінжиніринг програмного забезпечення відповідно до вимог замовника.
49. Як планувати, організовувати та здійснювати тестування, верифікацію та валідацію програмного забезпечення.
50. Як розробляти моделі та засоби обробки даних в розподілених системах та здійснювати оптимізацію програмного забезпечення з урахуванням вимог до надійності.

9. КРИТЕРІЇ ПІДСУМКОВОЇ ОЦІНКИ ЗНАНЬ СТУДЕНТІВ

Рівень знань оцінюється:

- «відмінно» / «зараховано» А - від 90 до 100 балів. Студент виявляє особливі творчі здібності, вміє самостійно знаходити та опрацьовувати необхідну інформацію, демонструє знання матеріалу, проводить узагальнення і висновки. Був присутній на лекціях та семінарських заняттях, під час яких давав вичерпні, обґрунтовані, теоретично і практично правильні відповіді, має конспект з виконаними завданнями до самостійної роботи, проявляє активність і творчість у науково-дослідній роботі;

- «добре» / «зараховано» В - від 82 до 89 балів. Студент володіє знаннями матеріалу, але допускає незначні помилки у формуванні термінів, категорій, проте за допомогою викладача швидко орієнтується і знаходить правильні відповіді. Був присутній на лекціях та семінарських заняттях, має конспект з виконаними завданнями до самостійної роботи, проявляє активність і творчість у науково-дослідній роботі;

- «добре» / «зараховано» C - від 74 до 81 балів. Студент відтворює значну частину теоретичного матеріалу, виявляє знання і розуміння основних положень, з допомогою викладача може аналізувати навчальний матеріал, але дає недостатньо обґрунтовані, невичерпні відповіді, допускає помилки. При цьому враховується наявність конспекту з виконаними завданнями до самостійної роботи та активність у науково-дослідній роботі;

- «задовільно» / «зараховано» D - від 64 до 73 балів. Студент був присутній не на всіх лекціях та семінарських заняттях, володіє навчальним матеріалом на середньому рівні, допускає помилки, серед яких є значна кількість суттєвих. При цьому враховується наявність конспекту з виконаними завданнями до самостійної роботи;

- «задовільно» / «зараховано» E - від 60 до 63 балів. Студент був присутній не на всіх лекціях та семінарських заняттях, володіє навчальним матеріалом на рівні, вищому за початковий, значну частину його відтворює на репродуктивному рівні, на всі запитання дає необґрунтовані, невичерпні відповіді, допускає помилки, має неповний конспект з завданнями до самостійної роботи.

- «незадовільно з можливістю повторного складання» / «не зараховано» FX – від 35 до 59 балів. Студент володіє матеріалом на рівні окремих фрагментів, що становлять незначну частину навчального матеріалу.

- «незадовільно з обов'язковим повторним вивченням дисципліни» / «не зараховано» F – від 1 до 34 балів. Студент не володіє навчальним матеріалом.

Таблиця відповідності результатів контролю знань за різними шкалами

100-бальною шкалою	**Шкала за ECTS**	**За національною шкалою**	
		екзамен	**залік**
90-100 (10-12)	A	Відмінно	зараховано
82-89 (8-9)	B	Добре	
74-81(6-7)	C		

64-73 (5)	D	Задовільно	
60-63 (4)	E		
35-59 (3)	Fx	незадовільно	не зараховано
1-34 (2)	F		

10. РЕКОМЕНДОВАНА ЛІТЕРАТУРА

Основна

1. Чистий код. Створення і рефакторинг за допомогою Agile / Роберт Сесіл Мартін. – «Фабула», 2019. – 448 с.
2. Програмна інженерія: підруч. / К.М. Лавріщева; Ін-т програм. систем НАН України. — К.: Академперіодика, 2008. — 320 с.
3. Fowler, M. Refactoring: Improving the Design of Existing Code Edition Unstated / Martin Fowler. – Addison-Wesley Professional, 2018. – 448 p.
4. Michael McCool, James Reinders, Arch Robison. Structured Parallel Programming: Patterns for Efficient Computation. 2012. – 432 p.

Допоміжна

1. Підхід до оптимізації програмного забезпечення для аналізу великих даних / А.М. Лавренюк, С.І. Лавренюк, П.Г. Тульчинський // Компьютерная математика. — 2017. — № 1. — С. 121-125.
2. Данченко О.Б. Практичні аспекти реінжинірингу бізнес-процесів / О. Б. Данченко. Київ: Університет економіки та права «КРОК», 2017. – 238 с.
3. Michael Feathers. Working Effectively with Legacy Code. – Person, 2004. – 464 p.
4. G. Szóke, C. Nagy, R. Ferenc and T. Gyimóthy, "Designing and Developing Automated Refactoring Transformations: An Experience Report," *2016 IEEE 23rd International Conference on Software Analysis, Evolution, and Reengineering (SANER)*, Osaka, Japan, 2016, pp. 693-697, doi: 10.1109/SANER.2016.17.
5. F. Coelho, T. Massoni and E. L.G. Alves, "Refactoring-Aware Code Review: A Systematic Mapping Study," *2019 IEEE/ACM 3rd International Workshop on Refactoring (IWoR)*, Montreal, QC, Canada, 2019, pp. 63-66, doi: 10.1109/IWoR.2019.00019.
6. N. Yoshida, S. Numata, E. Choiz and K. Inoue, "Proactive Clone Recommendation System for Extract Method Refactoring," *2019 IEEE/ACM*

3rd International Workshop on Refactoring (IWoR), Montreal, QC, Canada, 2019, pp. 67-70, doi: 10.1109/IWoR.2019.00020.

7. R. S. Bashir, S. P. Lee, C. C. Yung, K. A. Alam and R. W. Ahmad, "A Methodology for Impact Evaluation of Refactoring on External Quality Attributes of a Software Design," *2017 International Conference on Frontiers of Information Technology (FIT)*, Islamabad, Pakistan, 2017, pp. 183-188, doi: 10.1109/FIT.2017.00040.
8. N. A. Nagy and R. Abdalkareem, "On the Co-Occurrence of Refactoring of Test and Source Code," *2022 IEEE/ACM 19th International Conference on Mining Software Repositories (MSR)*, Pittsburgh, PA, USA, 2022, pp. 122-126, doi: 10.1145/3524842.3528529.
9. M. Mohan, D. Greer and P. McMullan, "Maximizing Refactoring Coverage in an Automated Maintenance Approach Using Multi-Objective Optimization," *2019 IEEE/ACM 3rd International Workshop on Refactoring (IWoR)*, Montreal, QC, Canada, 2019, pp. 31-38, doi: 10.1109/IWoR.2019.00014.
10. A. Brito, A. Hora and M. T. Valente, "Refactoring Graphs: Assessing Refactoring over Time," *2020 IEEE 27th International Conference on Software Analysis, Evolution and Reengineering (SANER)*, London, ON, Canada, 2020, pp. 367-377, doi: 10.1109/SANER48275.2020.9054864.
11. M. Agnihotri and A. Chug, "Understanding Refactoring Tactics and their Effect on Software Quality," *2022 12th International Conference on Cloud Computing, Data Science & Engineering (Confluence)*, Noida, India, 2022, pp. 41-46, doi: 10.1109/Confluence52989.2022.9734141.
12. J. Pérez, "Refactoring Planning for Design Smell Correction: Summary, Opportunities and Lessons Learned," *2013 IEEE International Conference on Software Maintenance*, Eindhoven, Netherlands, 2013, pp. 572-577, doi: 10.1109/ICSM.2013.98.
13. A. Pizzini, "Behavior-based test smells refactoring : Toward an automatic approach to refactoring Eager Test and Lazy Test smells," *2022 IEEE/ACM 44th International Conference on Software Engineering: Companion Proceedings (ICSE-Companion)*, Pittsburgh, PA, USA, 2022, pp. 261-263, doi: 10.1145/3510454.3517059.
14. S. Jindal and G. Khurana, "The statistical analysis of source-code to determine the refactoring opportunities factor (ROF) using a machine learning algorithm," *Fifth International Conference on Advances in Recent Technologies in Communication and Computing (ARTCom 2013)*, Bangalore, 2013, pp. 396-403, doi: 10.1049/cp.2013.2244.
15. Prykhodko S. A Statistical Evaluation of the Depth of Inheritance Tree Metric for Open-Source Applications Developed in Java / S. Prykhodko, N. Prykhodko, T. Smykodub // Foundations of Computing and Decision Sciences. – 2021. – Vol. 46. – No. 2. – P. 159-172. – ISSN 0867-6356, e-ISSN 2300-3405. DOI: https://doi.org/10.2478/fcds-2021-0011

16. Prykhodko S. A Joint Statistical Estimation of the RFC and CBO Metrics for Open-Source Applications Developed in Java / S. Prykhodko, N. Prykhodko, T. Smykodub // 2022 IEEE 17th International Conference on Computer Sciences and Information Technologies (CSIT), 2022. – P. 442-445. DOI: https://doi.org/10.1109/CSIT56902.2022.10000457
17. Prykhodko S.B. A statistical estimation of the coupling between object metric for open-source apps developed in Java / S. B. Prykhodko, K. S. Prykhodko, T. G. Smykodub // Herald of Advanced Information Technology. – 2022. – No. 3 (5). – P. 175-184. DOI: https://doi.org/10.15276/hait.05.2022.13

Інформаційні ресурси

1. Національна бібліотека України імені В. І. Вернадського [Електронний ресурс]: [Веб-сайт]. – Електронні дані. – Київ: НБУВ, 2013-2015. – Режим доступу: www.nbuv.gov.ua
2. Електронний каталог Національної парламентської бібліотеки України [Електронний ресурс]: [політемат. база даних містить відом. про вітчизн. та зарубіж. кн., брош., що надходять у фонд НПБ України]. – Електронні дані (803 438 записів). – Київ: Нац. парлам. б-ка України, 2002-2015. – Режим доступу: catalogue.nplu.org
3. Український інститут інтелектуальної власності [Електронний ресурс]: [Веб-сайт]. – Електронні дані. – Київ: УІІВ, 2017. – Режим доступу: http://www.uipv.org
4. Paul Hsieh. Programming Optimization. – [Електронний ресурс]: [Веб-сайт]. – Режим доступу: http://www.azillionmonkeys.com/qed/optimize.html
5. An introduction to the owl web ontology language. Режим доступу: http://www.cse.lehigh.edu/~heflin/IntroToOWL.pdf
6. Idiomatic Ways to Refactor Your Python Code / Yong Cui. – [Електронний ресурс]: [Веб-сайт]. – Режим доступу: https://towardsdatascience.com/10-idiotic-ways-to-refactor-your-python-code-cbb05bb0c820

ЗМІСТ

Навчальне видання

Стрелковська Ірина Вікторівна
Приходько Сергій Борисович
Григор'єва Тетяна Ігорівна

РЕІНЖИНІРІНГ ТА ОПТИМІЗАЦІЯ ПРОГРАМНИХ СИСТЕМ

Методичні рекомендації для самостійної роботи здобувачів

Українською мовою