Рівненський державний гуманітарний університет

Кафедра інформаційних технологій та моделювання

РОБОЧА ПРОГРАМА НАВЧАЛЬНОЇ ДИСЦИПЛІНИ

КОНСТРУЮВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Спеціальність **121 Інженерія програмного забезпечення**

Освітня програма **«Інженерія програмного забезпечення»**

Рівень вищої освіти **перший (бакалаврський)**

Факультет **математики та інформатики**

2024 – 2025 навчальний рік

Робоча програма навчальної дисципліни «Конструювання програмного забезпечення» для здобувачів першого (бакалаврського) рівня вищої освіти зі спеціальності 121 Інженерія програмного забезпечення за освітньою програмою «Інженерія програмного забезпечення»

Мова навчання: українська

Розробник: Копелюк В. О., викладач кафедри інформаційних технологій та моделювання

Робоча програма затверджена на засіданні кафедри інформаційних технологій та моделювання.
Протокол від 27 серпня 2024 року № 8.

Завідувач кафедри  Мороз І. П.

Робочу програму схвалено навчально-методичною комісією факультету математики та інформатики.

Протокол від 3 вересня 2024 року № 7.

Голова навчально-методичної комісії  Гнедко Н. М.

1. ОПИС НАВЧАЛЬНОЇ ДИСЦИПЛІНИ

<table>
<tr><th rowspan="2">Найменування показників</th><th rowspan="2">Галузь знань, спеціальність, ступінь вищої освіти</th><th colspan="2">Характеристика навчальної дисципліни</th></tr>
<tr><th>денна форма навчання</th><th>заочна форма навчання</th></tr>
<tr><td rowspan="20">Кількість кредитів – 4

Модулів – 1

Змістових модулів – 2

Індивідуальне науково-дослідне завдання –

Загальна кількість годин – 120

Тижневих годин для денної форми навчання:

аудиторних – 3

самостійної роботи – 6</td><td rowspan="20">Галузь знань:
12 Інформаційні технології

Спеціальність:
121 Інженерія програмного забезпечення

Освітня програма:
«Інженерія програмного забезпечення»

Рівень вищої освіти:
перший (бакалаврський)</td><td colspan="2">Обов’язкова</td></tr>
<tr><td colspan="2">Рік підготовки:</td></tr>
<tr><td>4-й</td><td>4-й</td></tr>
<tr><td colspan="2">Семестр:</td></tr>
<tr><td>7-й</td><td>7-й</td></tr>
<tr><td colspan="2">Лекції:</td></tr>
<tr><td>20 год.</td><td>6 год.</td></tr>
<tr><td colspan="2">Практичні:</td></tr>
<tr><td>-</td><td>-</td></tr>
<tr><td colspan="2">Лабораторні:</td></tr>
<tr><td>20 год.</td><td>6 год.</td></tr>
<tr><td colspan="2">Самостійна робота:</td></tr>
<tr><td>80 год.</td><td>108 год.</td></tr>
<tr><td colspan="2">Індивідуальні завдання:</td></tr>
<tr><td>-</td><td>-</td></tr>
<tr><td colspan="2">Вид контролю:</td></tr>
<tr><td>екзамен</td><td>екзамен</td></tr>
</table>

Передумови для вивчення дисципліни: «Об’єктно-орієнтоане програмування», «Моделювання та проєктування програмного забезпечення», «Якість і тестування програмного забезпечення»

2. МЕТА ТА ЗАВДАННЯ НАВЧАЛЬНОЇ ДИСЦИПЛІНИ

Навчальна дисципліна «Конструювання програмного забезпечення» відноситься до обов’язкових компонентів професійної підготовки здобувачів вищої освіти першого (бакалаврського) рівня за спеціальністю 121 Інженерія програмного забезпечення. Робоча програма навчальної дисципліни складена у відповідності до освітньо-професійної програми «Інженерія програмного забезпечення» підготовки бакалаврів за названою спеціальністю.

Конструювання програмного забезпечення (software construction) визначає детальне розроблення робочої програмної системи за допомогою комбінації кодування, верифікації (перевірки), модульного тестування (unit testing), інтеграційного тестування та відлагодження.

Іноді конструювання називають «кодуванням» або «програмуванням». Проте кодування часто передбачає механічну трансляцію розробленого плану в команди мови програмування, тоді як конструювання є зовсім не механічним процесом і часто пов’язане з творчістю та аналізом. У свою чергу зміст конструювання та програмування досить близький.

Конструювання пов’язане із проектуванням (Software Design) і тестуванням (Software Testing). Причиною цього є те, що сам по собі процес конструювання програмного забезпечення зачіпає важливі аспекти діяльності з проектування й тестування. Крім того, конструювання відштовхується від результатів проектування, а тестування передбачає роботу з результатами конструювання.

Метою викладання дисципліни є формування у здобувачів вищої освіти цілісної системи знань концепцій, принципів, методів та технологій конструювання програмного забезпечення.

Основним **завданнями** дисципліни є оволодіння визначальними принципами і технологіями конструювання програмного забезпечення, що дозволяють:

- аналізувати та оцінювати методи та інструментальні засоби конструювання ПЗ;
- вдосконалити володіння технологією об'єктно-орієнтованого конструювання;
- оволодіти методами та практичними підходами побудови моделей програм у процесі конструювання;
- оволодіти технологіями використання шаблонів конструювання програмного забезпечення;
- оволодіти методами та практикою тестування моделей та модулів;
- оволодіти технологіями підвищення якості програмного коду.

У результаті освоєння повного курсу навчальної дисципліни «Конструювання програмного забезпечення» у здобувачів вищої освіти мають сформуватися визначені освітньою програмою компетентності.

Загальні компетентності

К02. Здатність застосовувати знання у практичних ситуаціях.

Фахові компетентності

К15. Здатність розробляти архітектури, модулі та компоненти програмних систем.

К16. Здатність формулювати та забезпечувати вимоги щодо якості програмного забезпечення у відповідності з вимогами замовника, технічним завданням та стандартами.

К23. Здатність реалізовувати фази та ітерації життєвого циклу програмних систем та інформаційних технологій на основі відповідних моделей і підходів розробки програмного забезпечення.

К24. Здатність здійснювати процес інтеграції системи, застосовувати стандарти і процедури управління змінами для підтримки цілісності, загальної функціональності і надійності програмного забезпечення.

К25. Здатність обґрунтовано обирати та освоювати інструментарій з розробки та супроводження програмного забезпечення.

3. ОЧІКУВАНІ РЕЗУЛЬТАТИ НАВЧАННЯ

У результаті освоєння повного курсу навчальної дисципліни «Конструювання програмного забезпечення» у здобувачів вищої освіти формуються глибокі, міцні і системні знання, які передбачають вільне володіння понятійним апаратом, розуміння основних задач предмету, його мети та завдання, а також здатність до практичного застосування цих знань при реалізації прикладних застосувань. Згідно з освітньо-професійною програмою мають бути досягнуті наступні **програмні результати навчання**:

ПР03. Знати основні процеси, фази та ітерації життєвого циклу програмного забезпечення.

ПР06. Уміння вибирати та використовувати відповідну задачі методологію створення програмного забезпечення.

ПР13. Знати і застосовувати методи розробки алгоритмів, конструювання програмного забезпечення та структур даних і знань.

ПР15. Мотивовано обирати мови програмування та технології розробки для розв'язання завдань створення і супроводження програмного забезпечення.

ПР19. Знати та вміти застосовувати методи верифікації та валідації програмного забезпечення.

ПР20. Знати підходи щодо оцінки та забезпечення якості програмного забезпечення.

За результатами вивчення дисципліни здобувачі вищої освіти мають

знати:

- основні складові процесу конструювання ПЗ та взаємов'язки між ними;

- основні етапи конструювання;
- основні підходи і методи кодування (програмування), верифікації, модульного тестування, інтеграційного тестування та відлагодження.

вміти:

- здійснювати перевірку умов для успішного конструювання ПЗ;
- проектувати та описувати класи і методи;
- будувати управляючі структури та організовувати блоки команд;
- визначати способи подальшого тестування коду;
- здійснювати блочне тестування, інтеграційне тестування і відлагодження коду;
- проводити «шліфування» коду шляхом його ретельного форматування та коментування;
- здійснювати інтеграцію програмних компонентів, розроблених окремо;
- проводити оптимізацію коду, яка направлена на підвищення його швидкодії і зниження міри використання ресурсів.

4. ПРОГРАМА НАВЧАЛЬНОЇ ДИСЦИПЛІНИ

Змістовий модуль 1. «Інструменти, оптимізація та управління ресурсами»

Тема 1. Інструментарій для реалізації фаз і ітерацій життєвого циклу програмних систем.

Тема 2. Вимоги до програмного забезпечення.

Тема 3. Управління пам’яттю та ефективне використання ресурсів.

Тема 4. Пакетна обробка великих об’ємів даних.

Тема 5. Принципи рефакторингу та "запахи" в коді.

Змістовий модуль 2. «Рефакторинг, оптимізація та безпека коду»

Тема 6. Оптимізація коду: перенос функціоналу, організація даних, спрощення логіки.

Тема 7. Рефакторинг API та робота з наслідуванням.

Тема 8. Принципи конструювання безпечного коду.

5. СТРУКТУРА НАВЧАЛЬНОЇ ДИСЦИПЛІНИ

Назви змістових модулів і тем	Кількість годин											
	денна форма						заочна форма					
	усього	у тому числі					усього	у тому числі				
		лек	пр	лаб	інд	с.р.		лек	пр	лаб	інд	с.р.
1	2	3	4	5	6	7	8	9	10	11	12	13
Змістовий модуль 1. «Інструменти, оптимізація та управління ресурсами»												
Тема 1. Інструментарій для реалізації фаз і ітерацій життєвого циклу програмних систем.	6	2				4	6					
Тема 2. Алгоритми та оптимізація.	12	2		2		8	12	2		2		56
Тема 3. Управління пам’яттю та ефективне використання ресурсів.	12	2		2		8	12					
Тема 4. Пакетна обробка	12	2		2		8	12					

великих об’ємів даних.												
Тема 5. Принципи рефакторингу та "запахи" в коді.	18	4		2		12	18					
Разом за змістовим модулем 1	**60**	**12**	**-**	**8**	**-**	**40**	**60**	**2**	**-**	**2**	**-**	**56**
Змістовий модуль 2. «Рефакторинг, оптимізація та безпека коду»												
Тема 6. Оптимізація коду: перенос функціоналу, організація даних, спрощення логіки.	36	4		8		24	36	2		2		32
Тема 7. Рефакторинг API та робота з наслідуванням.	12	2		2		8	12	2		2		20
Тема 8. Принципи конструювання безпечного коду.	12	2		2		8	12					
Разом за змістовим модулем 2	**60**	**8**	**-**	**12**	**-**	**40**	**60**	**4**	**-**	**4**	**-**	**52**
Усього годин	**120**	**20**	**-**	**20**	**-**	**80**	**120**	**6**	**-**	**6**	**-**	**108**

6. ТЕМИ СЕМІНАРСЬКИХ ЗАНЯТЬ

Не передбачено навчальним планом

7. ТЕМИ ПРАКТИЧНИХ ЗАНЯТЬ

Не передбачено навчальним планом

8. ТЕМИ ЛАБОРАТОРНИХ ЗАНЯТЬ

(денна форма навчання)

№ з/п	Назва теми	Кількість годин
1.	Інструментарій для реалізації фаз і ітерацій життєвого циклу програмних систем.	2
2.	Алгоритми та оптимізація.	4
3.	Управління пам’яттю та ефективне використання ресурсів.	2
4.	Пакетна обробка великих об’ємів даних.	2
5.	Принципи рефакторингу та "запахи" в коді.	2
6.	Оптимізація коду: перенос функціоналу, організація даних, спрощення логіки.	2
7.	Рефакторинг API та робота з наслідуванням.	2
8.	Принципи конструювання безпечного коду.	4
	Разом	**20**

9. САМОСТІЙНА РОБОТА

(денна форма навчання)

№ з/п	Назва теми	Кількість годин

1	Інструментарій для реалізації фаз і ітерацій життєвого циклу програмних систем.	4
2	Алгоритми та оптимізація.	8
3	Управління пам’яттю та ефективне використання ресурсів.	8
4	Пакетна обробка великих об’ємів даних.	8
5	Принципи рефакторингу та "запахи" в коді.	12
6	Оптимізація коду: перенос функціоналу, організація даних, спрощення логіки.	24
7	Рефакторинг API та робота з наслідуванням.	8
8	Принципи конструювання безпечного коду.	8
	Разом	**80**

Самостійна робота студентів над теоретичним та практичним матеріалом навчальної дисципліни здійснюється в таких формах:

- вивчення лекційного матеріалу за темою дисципліни;
- вивчення теоретичного матеріалу за темою для самостійного опрацювання;
- опрацювання літератури за темою;
- підготовка до лабораторних робіт;
- робота з персональним комп’ютером для виконання завдань або обробки даних;
- робота в глобальній комп’ютерній мережі Інтернет для пошуку інформації по темі;
- виконання індивідуальних навчальних завдань.

10. ІНДИВІДУАЛЬНІ ЗАВДАННЯ

Не передбачено

11. МЕТОДИ НАВЧАННЯ

МН1 – словесний метод (лекція, дискусія, обговорення досліджуваного явища чи процесу, аналіз проблемних ситуацій);
МН2 – практичний метод (лабораторні заняття);
МН3 – наочний метод (ілюстрації, демонстрації);
МН4 – робота з навчально-методичною літературою (конспектування, складання рефератів);
МН5 – інтерактивний метод (із застосуванням аудіо, відео, новітніх інформаційних технологій та комп’ютерних засобів навчання);
МН6 – самостійна робота (самостійний аналіз, проектування та програмна реалізація завдань).

12. МЕТОДИ ОЦІНЮВАННЯ

МО1 – екзамени;
МО2 – усне або письмове опитування;
МО4 – тестування;
МО7 – презентації результатів виконаних завдань та досліджень;
МО9 – захист лабораторних робіт;

13. ЗАСОБИ ДІАГНОСТИКИ РЕЗУЛЬТАТІВ НАВЧАННЯ

- лабораторні роботи;
- усне або письмове опитування під час лабораторних занять;
- тестування;
- контрольні роботи;

- презентації результатів виконаних завдань та досліджень;
- підсумковий контроль у формі екзамену.

Види та методи навчання і оцінювання

Код компе-тентності (за ОПП)	Назва компетентності	Код ПРН	Назва програмного результату навчання	Методи навчання	Методи оцінювання результатів навчання
К02	Здатність застосовувати знання у практичних ситуаціях.	ПР06	Уміння вибирати та використовувати відповідну задачі методологію створення програмного забезпечення.	МН1, МН2, МН3, МН4, МН5, МН6	МО1, МО2, МО4, МО7, МО9
		ПР13	Знати і застосовувати методи розробки алгоритмів, конструювання програмного забезпечення та структур даних і знань.	МН1, МН2, МН3, МН4, МН5, МН6	МО1, МО2, МО4, МО7, МО9
		ПР19	Знати та вміти застосовувати методи верифікації та валідації програмного забезпечення.	МН1, МН2, МН3, МН4, МН5, МН6	МО1, МО2, МО4, МО7, МО9
		ПР20	Знати підходи щодо оцінки та забезпечення якості програмного забезпечення.	МН1, МН2, МН3, МН4, МН5, МН6	МО1, МО2, МО4, МО7, МО9
К15	Здатність розробляти архітектури, модулі та компоненти програмних систем.	ПР13	Знати і застосовувати методи розробки алгоритмів, конструювання програмного забезпечення та структур даних і знань.	МН1, МН2, МН3, МН4, МН5, МН6	МО1, МО2, МО4, МО7, МО9
К16	Здатність формулювати та забезпечувати вимоги щодо якості програмного забезпечення у відповідності з вимогами замовника, технічним завданням та стандартами..	ПР19	Знати та вміти застосовувати методи верифікації та валідації програмного забезпечення.	МН1, МН2, МН3, МН4, МН5, МН6	МО1, МО2, МО4, МО7, МО9
		ПР20	Знати підходи щодо оцінки та забезпечення якості програмного забезпечення.	МН1, МН2, МН3, МН4, МН5, МН6	МО1, МО2, МО4, МО7, МО9
К23	Здатність реалізовувати фази та ітерації життєвого циклу програмних систем та інформаційних технологій на основі відповідних моделей і підходів розробки програмного забезпечення.	ПР03	Знати основні процеси, фази та ітерації життєвого циклу програмного забезпечення.	МН1, МН2, МН3, МН4, МН5, МН6	МО1, МО2, МО4, МО7, МО9
		ПР06	Уміння вибирати та використовувати відповідну задачі методологію створення програмного забезпечення.	МН1, МН2, МН3, МН4, МН5, МН6	МО1, МО2, МО4, МО7, МО9
		ПР15	Мотивовано обирати мови програмування та технології розробки для розв’язання завдань створення і супроводження програмного забезпечення.	МН1, МН2, МН3, МН4, МН5, МН6	МО1, МО2, МО4, МО7, МО9
К24	Здатність здійснювати процес інтеграції системи, застосовувати	ПР03	Знати основні процеси, фази та ітерації життєвого циклу програмного забезпечення.	МН1, МН2, МН3, МН4, МН5, МН6	МО1, МО2, МО4, МО7, МО9

	стандарти і процедури управління змінами для підтримки цілісності, загальної функціональності і надійності програмного забезпечення.				
К25	Здатність обґрунтовано обирати та освоювати інструментарій з розробки та супроводження програмного забезпечення.	ПР15	Мотивовано обирати мови програмування та технології розробки для розв’язання завдань створення і супроводження програмного забезпечення.	МН1, МН2, МН3, МН4, МН5, МН6	МО1, МО2, МО4, МО7, МО9

14. КРИТЕРІЇ ОЦІНЮВАННЯ РЕЗУЛЬТАТІВ НАВЧАННЯ

Результат освітньої діяльності здобувача вищої освіти оцінюється згідно Положення про оцінювання знань і умінь здобувачів вищої освіти РДГУ за такими критеріями оцінювання та рівнями компетентності:

Суми балів за 100-бальною шкалою	**Оцінка в ЄКТС**	**Значення оцінки ЄКТС**	**Критерії оцінювання**	**Рівень компетентності**	**Оцінка за національною шкалою**
90-100	A	відмінно	здобувач вищої освіти виявляє особливі творчі здібності, вміє самостійно здобувати знання, без допомоги викладача знаходить і опрацьовує необхідну інформацію, вміє використовувати набуті знання і вміння для прийняття рішень у нестандартних ситуаціях, переконливо аргументує відповіді, самостійно розкриває власні здібності	високий (творчий)	відмінно
82-89	B	добре	здобувач вищої освіти вільно володіє теоретичним матеріалу, застосовує його на практиці, вільно розв'язує вправи і задачі у стандартних ситуаціях, самостійно виправляє допущені помилки, кількість яких незначна	достатній (конструктивно-варіативний)	добре
74-81	C	добре	здобувач вищої освіти вміє зіставляти, узагальнювати, систематизувати інформацію під керівництвом викладача, загалом самостійно застосовувати її на практиці;		

			контролювати власну діяльність; виправляти помилки, з-поміж яких є суттєві, добирати аргументи для підтвердження думок		
64-73	D	задовільно	здобувач вищої освіти відтворює значну частину теоретичного матеріалу, виявляє знання і розуміння основних положень, за допомогою викладача може аналізувати навчальний матеріал, виправляти помилки, з-поміж яких є значна кількість суттєвих	середній (репродуктивний)	задовільно
60-63	E	задовільно	здобувач вищої освіти володіє навчальним матеріалом на рівні, вищому за початковий, значну частину його відтворює на репродуктивному рівні		
35-59	FX	незадовільно з можливістю повторного складання семестрового контролю	здобувач вищої освіти володіє матеріалом на рівні окремих фрагментів, що становлять незначну частину навчального матеріалу	низький (рецептивно-продуктивний)	незадовільно
1-34	F	незадовільно з обов'язковим повторним вивченням дисципліни	здобувач вищої освіти володіє матеріалом на рівні елементарного розпізнавання і відтворення окремих фактів, елементів, об'єктів	низький (рецептивно-продуктивний)	незадовільно

Підсумкова (загальна) оцінка з навчальної дисципліни є сумою рейтингових оцінок (балів), одержаних за окремі оцінювальні форми навчальної діяльності: поточне і підсумкове оцінювання рівня засвоєння теоретичного та практичного матеріалу під час аудиторних занять і самостійної роботи; оцінка (бали) за виконання лабораторних завдань; оцінка (бали) за індивідуальну науково-дослідну роботу; оцінка (бали) за участь у наукових конференціях, олімпіадах, підготовку наукових публікацій, рефератів тощо.

15. РОЗПОДІЛ БАЛІВ, ЯКІ ОТРИМУЮТЬ ЗДОБУВАЧІ ВИЩОЇ ОСВІТИ

В університеті діє накопичувальна кредитно-трансферна система оцінювання програмних результатів навчання студентів, що реалізується в ході виконання і захисту лабораторних робіт, виконання ІНДЗ та модульного контролю, для яких визначено мінімальну кількість балів, яку слід набрати для формування рейтингового балу студента та виставлення його у залікову книжку і відомість успішності студентів з відповідними оцінками за національною та європейською кредитно-трансферною системами (ЄКТС).

Поточне тестування та самостійна робота								Екзамен	Сума
Змістовий модуль № 1					Змістовий модуль № 2				
Т1	Т2	Т3	Т4	Т5	Т6	Т7	Т8		
2	4	4	4	4	16	4	4		
Модульний контроль – 9					Модульний контроль – 9				
27					**33**			**40**	**100**

Т1, Т2, …, Т8 – теми змістових модулів.

Розподіл по видах освітньої діяльності

№ теми	Вид навчальної діяльності	Оціночні бали	Кількість балів
Т1	Самостійна робота	2	2
Т2	Виконання завдань лабораторної роботи № 1 Презентації результатів виконаних завдань Самостійна робота	2 1 1	4
Т3	Виконання завдань лабораторної роботи № 2 Презентації результатів виконаних завдань Самостійна робота	2 1 1	4
Т4	Виконання завдань лабораторної роботи № 3 Презентації результатів виконаних завдань Самостійна робота	2 1 1	4
Т5	Виконання завдань лабораторної роботи № 4 Презентації результатів виконаних завдань Самостійна робота	2 1 1	4
Модульний контроль № 1 (тест)		9	9
Т6	Виконання завдань лабораторних робіт №№ 5–8 Презентації результатів виконаних завдань Самостійна робота	8 4 4	16
Т7	Виконання завдань лабораторної роботи № 9 Презентації результатів виконаних завдань Самостійна робота	2 1 1	4
Т8	Виконання завдань лабораторної роботи № 10 Презентації результатів виконаних завдань Самостійна робота	2 1 1	4
Модульний контроль № 2 (контрольна робота)		9	9
Екзамен		40	40
Разом		**100**	

16. МЕТОДИЧНЕ ЗАБЕЗПЕЧЕННЯ

В якості навчально-методичного забезпечення самостійної роботи студентів використовуються:

- підручники, навчальні посібники, словники, довідники, енциклопедії, періодичні видання;
- лекційний матеріал у електронному вигляді, інтерактивні презентації;
- завдання та методичні вказівки для виконання лабораторних робіт;
- завдання модульної контрольної роботи;
- тести поточного контролю знань.

17. ПИТАННЯ ДЛЯ ПІДГОТОВКИ ДО ПІДСУМКОВОГО КОНТРОЛЮ

1. Основні фази життєвого циклу програмного забезпечення.
2. Інструменти для управління життєвим циклом ПЗ.
3. Використання CI/CD для автоматизації життєвого циклу розробки.
4. Відмінності між Scrum та Kanban у розробці ПЗ.

5. Вплив DevOps на якість та швидкість розробки програмного забезпечення.
6. Метрики оцінки ефективності життєвого циклу ПЗ.
7. Інструменти моніторингу продуктивності ПЗ на різних етапах розробки.
8. Методи забезпечення ефективної взаємодії між розробниками, тестувальниками та DevOps.
9. Оцінка складності алгоритмів (Big O notation).
10. Методи оптимізації алгоритмів.
11. Використання кешування для оптимізації алгоритмів.
12. Жадібні алгоритми та динамічне програмування.
13. Алгоритм Дейкстри та його застосування.
14. Хеш-таблиці та їх ефективність.
15. Оптимізація алгоритмів для роботи з великими наборами даних.
16. Методи профілювання продуктивності алгоритмів.
17. Використання стеку та купи (heap) в управлінні пам’яттю.
18. Механізм роботи збірки сміття (garbage collection).
19. Стратегії управління пам’яттю у мовах програмування з ручним керуванням пам’яттю.
20. Поняття memory leak та способи його уникнення.
21. Ефективне використання кеш-пам’яті у додатках.
22. Інструменти аналізу витрат пам’яті в програмному забезпеченні.
23. Робота механізму reference counting в управлінні пам’яттю.
24. Причини уникнення надмірного використання глобальних змінних для ефективного управління пам’яттю.
25. Поняття Batch processing та його відмінності від Stream processing.
26. Переваги використання пакетної обробки даних.
27. Організація ефективної черги обробки пакетних завдань.
28. Інструменти для реалізації пакетної обробки.
29. Методи балансування навантаження в системах пакетної обробки.
30. Підходи до оптимізації продуктивності пакетної обробки.
31. Гарантування цілісності даних у процесі пакетної обробки.
32. Типові проблеми у пакетній обробці та шляхи їх вирішення.
33. "Запахи" в коді і методи їх розпізнавання.
34. Основні техніки рефакторингу.
35. Вплив дублювання коду на підтримку проєкту.
36. Переваги та ризики внесення рефакторингу в активні проєкти.
37. Методи оцінки необхідності рефакторингу у кодовій базі.
38. Інструменти для автоматичного виявлення "запахів" у коді.
39. Розділення великих класів та методів під час рефакторингу.
40. Використання тестування під час рефакторингу.
41. Методи уникнення надмірного використання умовних операторів у коді.
42. Використання принципу DRY (Don't Repeat Yourself) для спрощення коду.
43. Причини уникнення глибокої вкладеності циклів і умов.
44. Оптимальні методи організації структур даних.
45. Вплив іммутабельних структур даних на продуктивність.
46. Методи скорочення надмірного логічного навантаження в коді.
47. Застосування шаблонних методів для оптимізації логіки коду.
48. Використання компілятора для оптимізації коду.
49. Причини уникнення глибокої ієрархії наслідування.
50. Методи проєктування API для гнучкого використання.
51. Відмінності між наслідуванням та композицією.
52. Використання SOLID-принципів у рефакторингу API.
53. Застосування патернів проєктування для рефакторингу API.
54. Вплив документування API на його підтримку.
55. Дотримання backward compatibility при зміні API.
56. Методи розширюваності API для спрощення майбутніх змін.

57. Шляхи запобігання SQL-ін'єкціям у веб-додатках.
58. Методи захисту від атак XSS (Cross-Site Scripting).
59. Безпечна робота з конфіденційними даними у коді.
60. Використання хешування замість зберігання паролів у відкритому вигляді.
61. Методи безпечного зберігання API-ключів та токенів.
62. Використання шифрування для захисту даних у програмному забезпеченні.
63. Методи перевірки вхідних даних для запобігання атакам.

18. РЕКОМЕНДОВАНА ЛІТЕРАТУРА

1. Мартін Р. Чистий код. К. : Фабула, 2019. 416 с.
2. Мартін Р. Чистий кодер. К. : Фабула, 2023. 256 с.
3. Томас Г. Кормен. Вступ до алгоритмів. К. : К.І.С., 2023. 1288 с.
4. ISO/IEC 25010:2011. Systems and software engineering. Systems and software Quality Requirements and Evaluation (SQuaRE). System and software quality models – Revises ISO/IEC 9126-1:2001: introduced 01.03.2011. – ISO/IEC, 2011. 34 p.
5. James Gough Mastering API Architecture: Defining, Connecting, and Securing Distributed Systems and Microservices O'Reilly Media, 2022. 400 p.
6. Jez Humble, David Farley Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation Longman (Pearson Education), 2010. 464 p.
7. Martin Fowler «UML Distilled: A Brief Guide to the Standard Object Modeling Language, 2 Edition Addison-Wesley Professional, 2013. 192 p.
8. Martin Fowler «UML Distilled: A Brief Guide to the Standard Object Modeling Language, 3rd Edition Addison-Wesley Professional, 2016. 208 p.
9. Мартін Р. Чиста архітектура. Мистецтво розробки програмного забезпечення. Фабула, 2019. 368 с.

19. ІНФОРМАЦІЙНІ (ІНТЕРНЕТ) РЕСУРСИ

- https://dou.ua/forums/tags/Junior/?from=fpcommunity
- https://www.ibm.com/docs/en/zos-basic-skills?topic=jobs-what-is-batch-processing
- https://www.w3schools.com/dsa/dsa_intro.php
- https://techdevguide.withgoogle.com/paths/data-structures-and-algorithms/