| Must have рівень:   1. Зроби порівняння статичних та динамічних технік тестування. Наведи переваги та можливі обмеження при використанні кожної з них.  |  | Статистична техніка тестування | Динамічна техніка тестування | | --- | --- | --- | | Основна інформація | Статичне тестування - це процес перевірки якості програмного забезпечення без його фактичного виконання. Воно здійснюється шляхом перегляду або інспекції коду, документації, діаграм, специфікацій тощо. | Динамічне тестування - це процес перевірки якості програмного забезпечення під час його фактичного виконання. Воно здійснюється шляхом запуску програми з різними вхідними даними для перевірки її функціональності та виявлення помилок. | | Перевага №1 | Виявлення помилок на ранніх етапах: Дозволяє виявити проблеми в програмі до того, як вона буде виконуватися, що зменшує вартість виправлень. | Виявлення реальних проблем: Дозволяє виявити проблеми, які можуть виникнути під час фактичного виконання програми. | | Перевага №2 | Підвищення якості продукту: Допомагає знайти та виправити помилки та неточності, що покращує якість програмного забезпечення. | Перевірка функціональності: Динамічне тестування дозволяє перевірити, чи працює програма, як очікувалося згідно з вимогами. | | Перевага №3 (і т.д.) | Ефективність ресурсів: Може зекономити час і ресурси, оскільки проблеми виявляються без фактичного виконання програми. |  | | Обмеження №1 | Не замінює динамічне тестування: Статичне тестування не може виявити всі типи помилок, які можуть виникнути під час реального виконання програми. | Вимагає більше ресурсів: Динамічне тестування вимагає часу та ресурсів для запуску тестів та обробки результатів. | | Обмеження №2 | Залежить від експертизи: Результати статичного тестування залежать від кваліфікації та досвіду переглядачів. | Пізнє виявлення проблем: Проблеми, виявлені під час динамічного тестування, можуть бути дорогими у виправленні, оскільки вони виявляються після написання коду. | | Висновок | Обидва види тестування мають свої переваги та обмеження. Статичне тестування допомагає знайти проблеми на ранніх етапах розробки та знижує вартість виправлень, але воно не може замінити динамічне тестування, яке перевіряє функціональність програми під час її виконання. Обидва підходи повинні використовуватися узгоджено, щоб забезпечити високу якість програмного забезпечення та забезпечити успішний реліз продукту. | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Середній рівень:   1. Виконай завдання попереднього рівня. 2. Наступне твердження стосується покриття рішень:   *Коли код має одну ‘IF” умову, не має циклів (LOOP) або перемикачів (CASE), будь-який тест, який ми виконаємо, дасть результат 50% покриття рішень (decision coverage).*  Яке твердження є коректним?   * 1. Коректно. Будь-який тест кейс надає 100% покриття тверджень, таким чином покриває 50% рішень.   2. Коректно. Результат будь-якого тесту умови IF буде або правдими, або ні.   3. Некоректно. Один тест може гарантувати 25% перевірки рішень в цьому випадку.   4. Некоректно, бо занадто загальне твердження. Ми не можемо знати, чи є воно коректним, бо це залежить від тестованого ПЗ.  1. Є псевдокод: Switch PC on -> Start MS Word -> IF MS Word starts THEN -> Write a poem -> Close MS Word.   Скільки тест кейсів знадобиться, щоб перевірити його функціонал?   * 1. 1 – для покриття операторів, 2 – для покриття рішень   2. 1 – для покриття операторів, 1 – для покриття рішень   3. 2 – для покриття операторів, 2 – для покриття рішень   4. 2 – для покриття операторів, 1 – для покриття рішень  1. Скільки потрібно тестів для перевірки тверджень коду:     1. 2    2. 1 (TC1: P=60, Q=60)    3. 3    4. 4 |
| Програма максимум: **Виконана нижче)**   1. Виконай завдання двох попередніх рівнів. 2. Продовжуємо розвивати стартап для застосунку, який дозволяє обмінюватися фотографіями котиків.   Є алгоритм:  *Запитай, якого улюбленця має користувач.*  *Якщо користувач відповість, що має кота, то запитай, яка порода його улюбленця: «короткошерста чи довгошерста?»*  *Якщо клієнт відповість «довгошерста», то запитай: «ви бажаєте отримати контакти найближчого грумера?»*  *Якщо клієнт відповість «так», то скажи: «Надайте адресу найближчої котячої перукарні»*  *Інакше*  *Скажи: «Запропонуй магазин з товарами по догляду за шерстю»*  *Закінчити*  *Інакше*  *Скажи «Запропонуй обрати магазин із зоотоварами»*  *Закінчити*  *Якщо клієнт не має кота*  *Скажи “Коли вирішите завести улюбленця – приходьте”*  *Закінчити*  Завдання:   1. Намалюй схему алгоритму (в інструменті на вибір, наприклад, у вбудованому Google Docs редакторі, [figjam](https://www.figma.com/figjam/) чи [miro](https://miro.com/)) 2. Який потрібен мінімальний набір тест-кейсів, щоб переконатися, що всі запитання були поставлені, всі комбінації були пройдені та всі відповіді були отримані? |



**Statement: 4**

* **TC1: Так, Довгошерстий, Так**
* **TC2: Так, Довгошерстий, Ні**
* **TC3: Так, Короткошерстий**
* **TC4: Ні**

**Decision: 4**

* **TC1: Так, Довгошерстий, Так**
* **TC2: Так, Довгошерстий, Ні**
* **TC3: Так, Короткошерстий**
* **TC4: Ні**

**Condition: 4**

* **TC1: Так, Довгошерстий, Так**
* **TC2: Так, Довгошерстий, Ні**
* **TC3: Так, Короткошерстий**
* **TC4: Ні**