| 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 | вид тестування ПЗ, який виконується без активного запуску коду | вид тестування ПЗ, який передбачає запуск коду | | Перевага №1 | дозволяє отримувати об'єктивні результати, оскільки вона базується на числових даних та аналізі | дозволяє виявити реальні помилки та проблеми програми під час її виконання, що робить його корисним для виявлення дефектів, які можуть виникнути в реальних умовах використання | | Перевага №2 | дозволяє виявити дефекти та недоліки, які можуть бути важко виявити за допомогою інших методів тестування | Виконання програми в реальному або симульованому середовищі дозволяє перевірити її роботу в умовах, близьких до реальних. | | Перевага №3 (і т.д.) | дозволяє визначити якість програми, включаючи стабільність, надійність та продуктивність | Динамічне тестування може допомогти виявити проблеми з витратою пам'яті, процесорного часу та інших ресурсів. | | Обмеження №1 | Для використання статистичної техніки тестування потрібні дані. Це може бути важким завданням, якщо ви не маєте доступу до достатньої кількості реальних даних для аналізу. | може не виявити всі можливі шляхи виконання програми, оскільки кількість можливих варіантів велика. | | Обмеження №2 | Використання статистичних методів може бути складним завданням і вимагати спеціальних знань та навичок в області статистики. | Потрібен доступ до вихідного коду програми, що може бути проблематичним у випадку, коли програма підлягає комерційним обмеженням | | Обмеження №3 (і т.д.) | Якщо аналіз статистичних даних виконується некоректно або неправильно, це може призвести до невірних висновків та помилок у визначенні проблем. | може бути менш чутливим до деяких типів помилок, які інші методи тестування можуть виявити краще | | Висновок | Статистична техніка тестування може бути корисним інструментом для забезпечення якості програмного забезпечення, але вона повинна використовуватися з розумінням і належними знаннями статистики для досягнення найкращих результатів. | Динамічне тестування є важливим інструментом для виявлення та виправлення помилок у програмному забезпеченні, але воно повинно використовуватися спільно з іншими методами тестування для забезпечення високої якості програми. | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 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    3. 3    4. 4 |
| Програма максимум:   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?     Думаю тест кейсів треба 4.  Тест кейс 1. Початкове питання “Хто улюбленець?”  - не кіт - “заведеш приходь”  - кіт - “Яка порода?”  Тест кейс 2. “Яка порода?”  -короткошерста  -Довгошерста  Тест кейс 3. Якщо обрали “Довгошерста” - “Бажаєте грумера?”   * так - надати адресу * ні   Тест кейс 4. Якщо обрали “ні”   * запропонувати “*магазин з товарами по догляду за шерстю”*   - запропонувати “*магазин із зоотоварами”* |