| 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** | це техніка котра тестує ПЗ без його запуску (без запуску коду), спрямована на виявлення дефектів ще на етапі створення вимог.  можна поділити на дві категорії такі як **ревью** (виправлення несправностей в документах) та **статистичний аналіз** (аналіз коду, на наявність помилок)**.** | ця техніка тестування перевіряє ПЗ тоді, коли воно працює (коли код виконується). виконується шляхом фактичного використання вже готової програми, перевіряється чи працює програма так, як очікується. | | **Перевага №1** | виконується одна із “заповідей” тестування - тестування на самих ранніх стадія розробки ПЗ. із-за чого, виправлення помилок коштує дешевше, аніж у вже запущенному ПЗ. | ретельно досліджуючи тестувальник перевіряє усю функціональність ПЗ, щоб якість відповідала найвищим стандартам. | | **Перевага №2** | сприяє продуктивності розробки, тому що значно скорочує час та зусилля на виправлення помилок | виявляє складні помилки, котрі могло б не помітити статистичне тестування (етап ревью) | | **Перевага №3** | підвищує якість ПЗ. | ПЗ тестується з точки зору користувача, допомагає уникнути помилок, на які могли б потрапити користувачі. це ще раз підвищує якість ПЗ. | | **Перевага №4** | покращує взаємозв’язок та обмін інформацією між членами команди | може бути автоматизовано, на відміну від статистичної техніки. | | **Обмеження №1** | цей процес займає багато часу, бо він виконується вручну + треба передивитися весь код та перечитати усю документацію | також займає багато часу | | **Обмеження №2** | потрібні кваліфіковані спеціалісти, котрі розуміються у коді та його виконанні. | висока вартість проведення тестування та виправлення помилок, якщо вони були знайдені. | | **Обмеження №3** | цей метод перешкоджає виявленню вразливостей у вже працюючому ПЗ. | баги виявляються пізніше, вже після завершення розробки і їх виявляється менше аніж при статичній техніці. |   **Висновок:** хоча обидва ці види тестування займають багато часу, мають недоліки та переваги один перед одним, мені здається що в сумістності вони дуже добре “співпрацюють” та можуть запобігти зниженню якості ПЗ. я б використовувала їх разом, тоді, як я вважаю, в ПЗ буде виявлено набагато менше багів, аніж при використанні тільки однієї техніки. але якщо в нас мало часу на розробку ПЗ, тоді б, напевно, я надала перевагу динамічній техніці, бо бачення ПЗ з точки зору користувача може усунути те, на що може натрапити якраз таки кінцевий користувач, тоді є шанс загубити клієнта, а це не дуже добре. |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 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.   моя думка:  (якщо IF має тільки одне твердження то результатом тесту може бути або негативний сценарій, або позитивний, більше ніякого результату не предусмотрено. я вважаю що і варіант “а.” буде правильною відповіддю, бо 50 відсотків рішень так і 50 відсотків рішень ні справді покриті) це щось типу прикладу 2+2, де правильна відповідь тіки 4, а усі інші не є правильними.   * 1. Некоректно. Один тест може гарантувати 25% перевірки рішень в цьому випадку.   2. 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   моя думка:  я вважаю, що повинно бути 2 покриття рішень -*якщо* MS Word відкриється і якщо ні. є два рішення так і ні. і один для покриття операторів, перевірки самого MS Word, якщо вірш не напишиться, то цей тест кейс можна буде перенести в баг, бо функція повинна надавати змогу написати вірш в програмі.   * 1. 1 – для покриття операторів, 1 – для покриття рішень   2. 2 – для покриття операторів, 2 – для покриття рішень   3. 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?   [таблиця](https://miro.com/welcomeonboard/QlgxNDlQSjFyR3piSzJkRlpKd1dHaTlNUjRGUjVYVTNjcDA4WlVrVnBMUkVGQlVWV08zMzlyNThROEJISFBtU3wzNDU4NzY0NTMyOTUxMDA2NTQz?share_link_id=884579140227)  кейсів для перевірки потрібно 5   1. 1a - 2c - 3d - 4f - 5 2. 1a - 2b - 6g - 7h - 9 3. 1a - 2b - 6g - 7i - 8j - 10m - 13o - 15 4. 1a - 2b - 6g - 7i - 8j - 10L - 12n - 14 5. 1a - 2b - 6g - 7i - 8j - 10k - 11 |