| 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 | Техніка тестування, яка виконується для пошуку потенційних дефектів і застосовується без запуску коду. | Техніка тестування, яка виконується для перевірки функціональності програми та при якій код має бути запущений. | | Перевага №1 | Відбувається на ранніх етапах, тобто являється частиною раннього тестування, що дозволяє знайти та завчасно виправити дефекти до того, як код буде запушений (під час динамічного тесту) | Виявляє більш складні дефекти, котрі могли бути не помічені або котрі не проявляли себе під час статичного тестування | | Перевага №2 | Значно економить час та витрати, якщо подібне тестування було проведене перед динамічним, адже виправивши віднайдені помилки в коді, ви економите час на динамічному тестуванні і відповідно - гроші на виправлення дефектів вже після запуску коду | При використанні певних технологій та інструментів, процес динамічного тестування можна автоматизувати, що звичайно також зекономить час. | | Перевага №3 | Під час подібної перевірки підвищується хард скіл самих розробників та їх комунікація з іншими колегами, адже таким чином виявляються неточності, пропуски, неоднозначності в коді, відповідно це веде розробника до певних висновків і уникнення таких помилок в майбутньому | Таке тестування фактично показує роботу ПЗ з точки зору кінцевого користувача, тож динамічне тестування допомагає перевірити функціонал створеного продукту “очима” майбутнього юзера | | Обмеження №1 | Такий процес важко автоматизувати, тож в основному цей тест виконується вручну, що займає доволі багато часу | Якщо не провести статичне тестування, то динамічне може виявитись доволі витратним для бізнесу і виправлення дефектів виявиться значно затратнішим | | Обмеження №2 | Для виконання такого тестування необхідна доволі обширна кількість знань про сам код, тобто спеціаліст має бути інформаційно підкований в тому, що він перевіряє і як саме має бути | Виконувати динамічне тестування неможливо, не закінчивши код, тож перш ніж до нього приступити, треба почекати доки процес розробки коду та його перевірки буде завершений, тому такий вид тестування може відбуватись лише на пізніх етапах розробки | | Обмеження №3 | Інколи, наприклад якщо ваша компанія це аутсорс чи аутстаф компанія, для проведення статичного аналізу коду необхідний доступ до вихідних кодів програми, що може бути обмеженням у випадку комерційних або закритих проектів. Може бути таке, що вихідний код обмежений з точки зору конфіденційності та інтелектуальної власності. | Після внесення змін до програми необхідно оновлювати динамічні тест-кейси, щоб вони відповідали вже оновленому функціоналу. Це звичайно означає додаткові витрати часу та зусилля на тестування. | | Висновок | Отже, кожна техніка тестування може існувати сама по собі і виконуватись без взаємодії з іншою, тим не менш, якщо під час тестування розробленого ПЗ використовувати лиш одну техніку, це може мати наслідки, як-от витрати часу та грошей для виправлення великої кількості дефектів, якщо не користуватися статичним тестуванням, або отримувати скарги користувачів та втрачати їх прихильність, якщо не скористатись динамічним тестуванням, що теж призведе до втрати грошей. Для отримання справді якісного і працюючого продукту найкраще використовувати обидві техніки, але кожен бізнес самостійно зважає на свої можливості та необхідності. | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 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? |