| 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 не потребує запуску програми, тобто дає змогу виявити помилки на ранніх етапах розробки та проблеми в документації | Техніка передбачає запуск програми, аналізується програма під час роботи. Тестується код, а отже він повинен бути написаний до початку тестування | | Перевага №1 | Раннє виявлення та виправлення будь-яких помилок кодування | Ретельне вивчення всього функціоналу програми і, як наслідок, якісні результати | | Перевага №2 | Зниження вартості на ранніх стадіях розробки, виходячи з обсягу доопрацювань, необхідних для виправлення помилок | Це добре структурований процес, який перевіряє програму з точки зору користувача, що підвищує якість програми | | Перевага №3 | Скорочення термінів розробки | Дозволяє виправити складні помилки, які могли залишитися непоміченими на етапі перегляду коду (який є частиною статичного тестування) | | Перевага №4 | Статичне тестування може підвищити рівень комунікації між командами | Можна автоматизувати динамічне тестування | | Обмеження №1 | Процес статичного тестування може забирати багато часу, оскільки він здебільшого виконується вручну. | Займає багато часу, оскільки виконується програма/програмне забезпечення або код, що потребує багато ресурсів. | | Обмеження №2 | Це перешкоджає пошуку вразливостей, внесених у середовище виконання. | Збільшує вартість проекту, оскільки програма не починається на ранній стадії життєвого циклу програмного забезпечення, і будь-які проблеми, які вирішуються пізніше, а отже, виправлення коштує дорожче | | Висновок | Незважаючи на те, що статичне тестування вимагає тривалого часу, проведеного в п'янких дискусіях і зустрічах, воно варте того, щоб запобігти появі дефектів на останніх етапах розробки продукту. Тому статичне тестування, як етап верифікації, по праву вважається основним кроком на шляху до розробки програми без помилок. | Незважаючи на те, що динамічне тестування потребує багато часу, воно є важливим, так як (як етап валідації) додає цінність до фінальної якості продукту. Від безпосереднього тестування програмного забезпечення до перевірки функціональності, ефективності, надійності та інших важливих елементів програмного забезпечення. | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 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 »*  *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/))   <https://miro.com/app/board/uXjVPiA4oBI=/?share_link_id=484334333947>   1. Який потрібен мінімальний набір тест-кейсів, щоб переконатися, що всі запитання були поставлені, всі комбінації були пройдені та всі відповіді були отримані? **3 - для покриття операторів, 5 - для покриття рішень** |