| 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 | не вимагає запускати програму чи додаток, дає змогу знайти найбільш очевидні помилки ще на ранніх етапах створення продукту та включає рев’ю (перевірку робочих продуктів, як-от документація, користувацькі історії, діаграми, інструкції та інші документи, що використовуються під час розробки ПЗ) і статичний аналіз (автоматизоване тестування коду та документації для пошуку очевидних помилок та хиб). | тип тестування, який передбачає запуск програмного коду. Тобто поведінка програми аналізується під час її роботи. Для виконання динамічного тестування необхідно, щоб програмний код, який тестується, був написаний, скомпільований та запущений. | | Перевага №1 | -можливо почати тестування на самих ранніх етапах, до динамічного, що вважається дуже ефективним | -на відміну від статичного тестування, в рамках динамічного тестувальник в реальних умовах досліджує функціональність програми | | Перевага №2 | -можилвість попередження проблем у коді або дизайні через виявлення невідповідностей, неточностей, суперечностей, пропусків і тд | -можливо виявити проблеми, які були пропущені або не можливо було виявити в рамках статичного тестування: | | Перевага №3 | підвищення якості розробки та можливості підтримки коду  -скорочення витрат на розробку та тестування  -зниження загальних витрат за рахунок виявлення дефектів на ранніх стадіях | -динамічне тестування можна оптимізувати та автоматизувати | | Обмеження №1 | -статичне тестування є досить поверхневим і загальним і включає в себе обговорення та рев’ю | -динамічне тестування є дорожчим та більш затратним з точки зору часу процесом | | Обмеження №2 | -через неуважність або недосвідченість виконавця багато кейсів, які на етапах рев’ю могли бути б розкритими, такими не стануть і все одно доведеться викривати баги в рамках динамічного тестування | -оскільки динамічне тестування можливо використовувати вже зазвичай після завершення кодування, то і баги будуть виявлені пізніше | | Обмеження №3 (і т.д.) | -є доречним на ранніх етапах розробки та тестування, але на більш пізніших вже більш доречним буде саме динамічне |  | | Висновок | Статичне тестування є дуже ефективним на ранніх стадіях розробки та тестування, допомогає налагодити процеси та коммунікацію між учасниками команди, скорочує витрати та підвищує якість розробки та тестування, передбачає різні проблеми у коді або дизайні заздалегідь. Має певні недоліки пов’язани із тим, хто саме виконує статичне тестування та на яких етапах розробки та тестування.  Обидва види є взаємодоповнюючими та в рамках розробки та тестування краще використовувати як статичне, так і динамічне. | Динамічне тестування перевіряє та підтверджує якість програмного забезпечення. В рамках динамічного тестування виконуються тести ПЗ які перевіряють функціональність, продуктивність, ефективність, надійність і тд, що неможливо в рамках статичного. Але також має свої недоліки щодо вартості та своєчасності використання. Обидва види є взаємодоповнюючими та в рамках розробки та тестування краще використовувати як статичне, так і динамічне. | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 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   *Якщо я розумію вірно, оператори це саме кроки, а рішення, це наше IF, де ми перевіримо на True / False.*     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? - 5 тест кейсів (схема знизу) |

