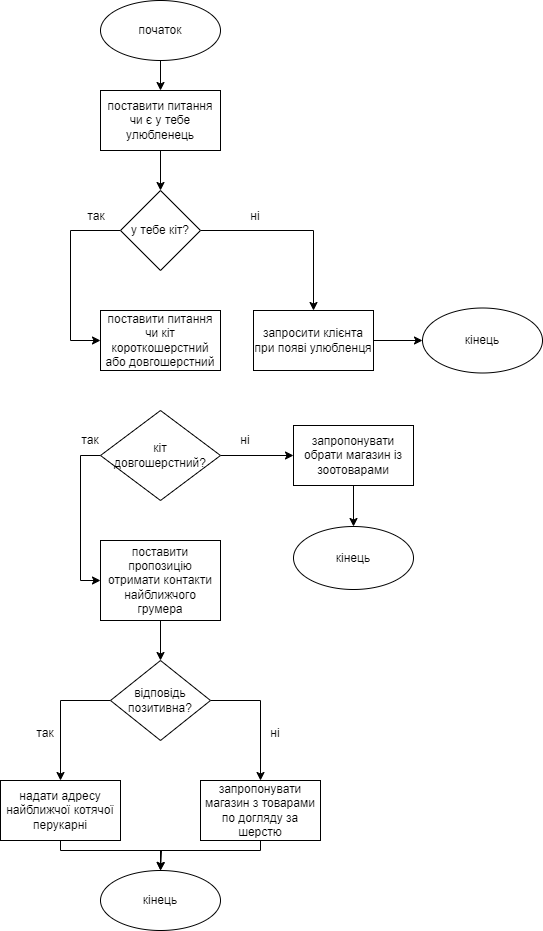
| Must have рівень:   1. Зроби порівняння статичних та динамічних технік тестування. Наведи переваги та можливі обмеження при використанні кожної з них.  |  | Статистична техніка тестування | Динамічна техніка тестування | | --- | --- | --- | | Основна інформація | Статичне тестування, на відміну від динамічного, не потребує запуску програми чи додатка. Це дозволяє виявляти очевидні помилки на початкових етапах розробки продукту. Статичне тестування включає рев'ю, що означає перевірку робочих продуктів, таких як документація, користувацькі історії, діаграми та інструкції, а також статичний аналіз, який включає автоматизоване тестування коду та документації для виявлення очевидних помилок та недоліків. | Динамічне тестування, навпаки, передбачає запуск програмного коду для аналізу його поведінки під час виконання. Для проведення динамічного тестування необхідно написати, скомпілювати та запустити програмний код. В процесі може бути оцінена продуктивність програми, включаючи параметри, такі як завантаження процесора, використання пам'яті та час відповіді. | | Перевага №1 | Ефективність при визначенні дефектів ранньою стадією | Виявлення дефектів під час виконання програми | | Перевага №2 | Економія ресурсів | Можливість оцінки продуктивності | | Перевага №3 | Зручність в області аналізу документації | Більша автоматизація | | Обмеження №1 | Не виявляє помилок виконання програми | Вимагає часу та ресурсів | | Обмеження №2 | Обмежена можливість виявлення дефектів у складних системах | Не може покрити всі можливі сценарії | | Обмеження №3 | Потребує досвідчених аналітиків | Може бути складно відтворити деякі сценарії | | Висновок | Статичне тестування дозволяє виявляти помилки на ранніх етапах розробки без запуску програми. | Динамічне тестування визначається виконанням програмного коду під час роботи, надаючи інформацію про його функціональність та продуктивність в реальному середовищі. | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 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.   Пояснення: Якщо у коді є лише одна умова IF, то є два можливих варіанти виконання цієї умови - коли вона істинна і коли вона хибна. Отже, одне тестове випробування може покрити один з цих варіантів, що відповідає 50% покриття рішень. Але це не означає, що цей тест гарантує 100% покриття тверджень, оскільки інша гілка коду (яка виконується, коли умова хибна) залишається непокритою. Таким чином, кожен тест може гарантувати лише 25% покриття рішень у цьому випадку.   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? |



№1

№2

1. Тест-кейс 1: Користувач має кота короткошерстного.

- Введена відповідь: "короткошерста".

- Очікувана відповідь: "Запропонуй магазин з товарами по догляду за шерстю".

2. Тест-кейс 2: Користувач має кота довгошерстого і хоче отримати контакти грумера.

- Введена відповідь: "довгошерста" -> "так".

- Очікувана відповідь: "Надайте адресу найближчої котячої перукарні".

3. Тест-кейс 3: Користувач має кота довгошерстого, але не хоче отримувати контакти грумера.

- Введена відповідь: "довгошерста" -> "ні".

- Очікувана відповідь: "Запропонуй магазин з товарами по догляду за шерстю".

4. Тест-кейс 4: Користувач не має кота.

- Очікувана відповідь: "Коли вирішите завести улюбленця – приходьте".

5. Тест-кейс 5: Користувач має кота короткошерстного.

- Введена відповідь: "короткошерста".

- Очікувана відповідь: "Запропонуй магазин з товарами по догляду за шерстю".

6. Тест-кейс 6: Користувач має кота довгошерстого, але не хоче отримувати контакти грумера.

- Введена відповідь: "довгошерста" -> "ні".

- Очікувана відповідь: "Запропонуй магазин з товарами по догляду за шерстю".