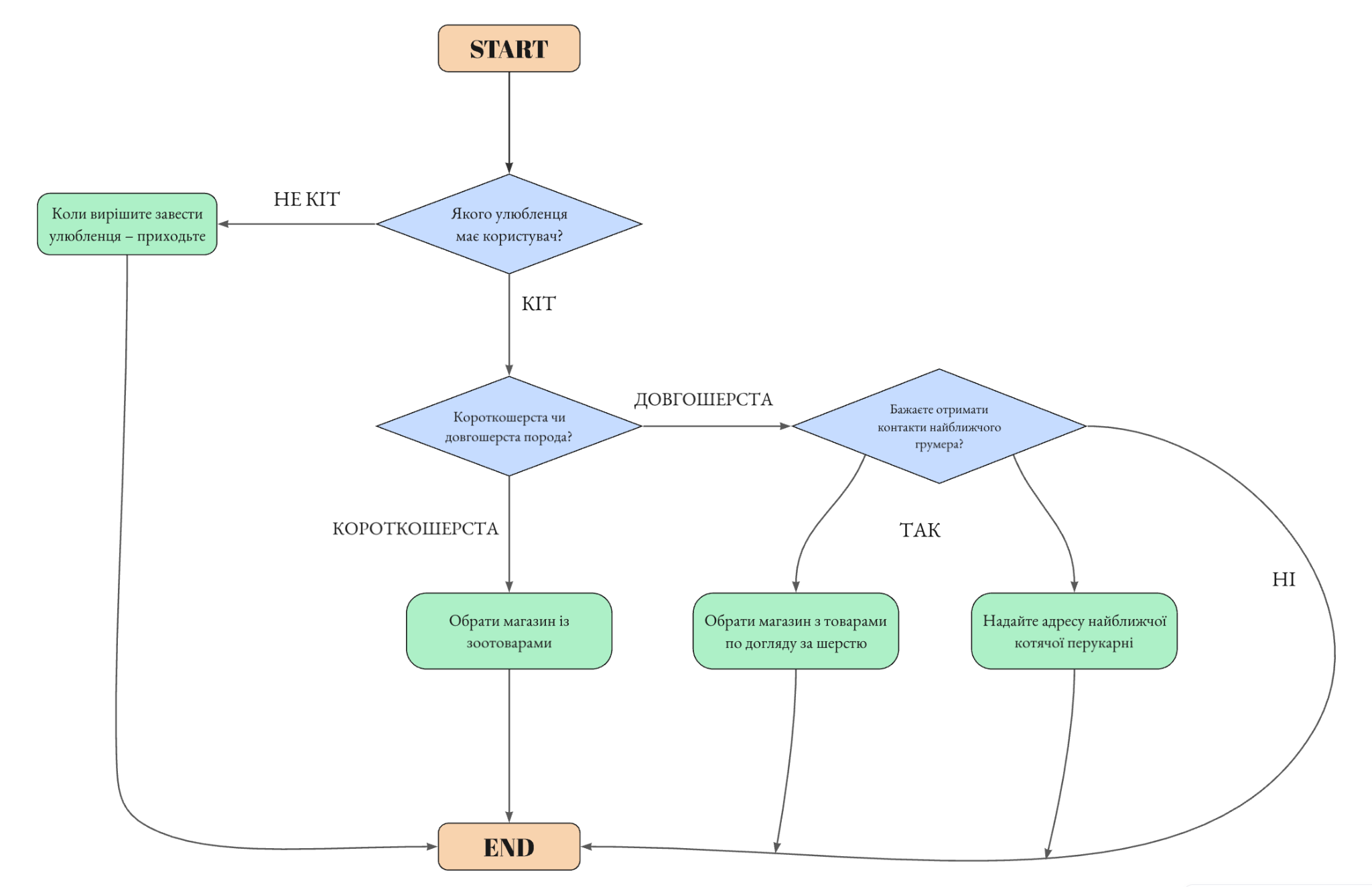
| Must have рівень:   1. Зроби порівняння статичних та динамічних технік тестування. Наведи переваги та можливі обмеження при використанні кожної з них.  |  | **Статистична техніка тестування** | **Динамічна техніка тестування** | | --- | --- | --- | | Основна інформація | це тестування, де ми не працюємо з запущеним кодом.  види:  -ревʼю (аналіз вимог, дизайну, юзерсторі, бізнес-процесів і тп)  -статичний аналіз коду | уе тестування, де ми запускаємо код/ програму та дивимось чи правильно вони працюють.  методи:  -чорна скринька  - біла скринька  -сіра скринька  види:  -юніт тести  - інтеграційне  - приймальне  - регресійне  - системне | |  | | | | Перевага №1 | можливість виявлення багів на ранніх етапах | виявляє баги, які непомітні під час простого аналізу коду, а виявляють себе тільки під час запуску коду на практиці | | Перевага №2 | cost-effective - вартість виправлення багу на ранньому етапі значно нижча | перевіряє функціональність та інтеграцію модулів між собою | | Перевага №3 | підвищує якість коду, документації, вимог, та в цілому ПЗ | може бути автоматизовано | | Перевага №4 |  | покращує та робить user experience надійнішим | |  | | | | Обмеження №1 | не виконує код, а лише сканує, тому може знайти лише поверхневі баги, повʼязані зі структурою , помилками у вимогах/дизайні | займає багато часу на тестування, а відповідно виправлення багу також займає більше часу | | Обмеження №2 | залежить від навичок та досвіду тестувальника | вартість такого виду тестування вище, бо потребує, крім часу, ще багато ресурсів | | Обмеження №3 | якщо все виконується вручну, то це займає більше часу | тестування виняткових випадків може бути складним у проведенні | | Обмеження №4 | не заміняє динамічне тестування, а лише доповнює його | автоматизовані тести можуть викликати відчуття того, що все перевірено, однак вони перевіряють лише те, що прописано у сценарії | |  | | | | Висновок | дає змогу виявити баги ще до запуску коду та може економити ресурси, але знаходить доволі поверхневі баги | потребує набагато більше часу та ресурсів, але виявляє більш критичні та суттєві баги, які непомітні у статичному тестуванні | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 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. - НЕВІРНЕ   2. Коректно. Результат будь-якого тесту умови IF буде або правдими, або ні. - КОРЕКТНЕ   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? |

1.Посилання на схему міро:

<https://miro.com/welcomeonboard/VVdjcFZsZzdEN3dPb2gwbEtkREVxaW42bkhqZjY0VDI4R2ZXc2JlbVRxMWlnWTZvbDc4ckRjQ1hSS3ZVK2J3ZnlLb09KeG9icENrai8yd0cwRDFzRURJV3Z2elkxT3hIZUJPcjBUb0ZFWEVQakExbGJIdUR6ZFQzRXpPTHdvVVZnbHpza3F6REdEcmNpNEFOMmJXWXBBPT0hdjE=?share_link_id=688761308662>

Також прикріплюю скрин нижче:



2.Який потрібен мінімальний набір тест-кейсів, щоб переконатися, що всі запитання були поставлені, всі комбінації були пройдені та всі відповіді були отримані?

мінімальний набір тест-кейсів = мінімальний набір покриття буде дорівнювати 5