Must have рівень:

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 тестування 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). | Динамічне тестування — 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. При цьому може виконуватися перевірка зовнішніх параметрів роботи програми: завантаження процесора, використання пам’яті, час відповіді тощо — тобто, її продуктивність. |
| Перевага №1 | Пошук та усунення дефектів вважається більш ефективним до проведення динамічного тестування; | В процессе тестирования проводится тщательное изучение всего функционала программы, в результате чего получаем высокое качество проверк |
| Перевага №2 | ідентифікація дефектів, які складно виявити при динамічному тестуванні; | Динамическое тестирование – хорошо структурированный процесс, осуществляющий проверку программы со стороны пользователя, что, в свою очередь, значительно повышает качество программного обеспечения; |
| Перевага №3 | попередження дефектів дизайну або кодування шляхом виявлення невідповідностей, неоднозначностей, суперечностей, пропусків та неточностей; | Фиксация сложных дефектов, которые могли остаться незамеченными на этапе проверки кода; |
| Перевага №4 | підвищення якості розробки, включно з покращенням дизайну та підтримкою коду; | Динамический тип тестирования, при помощи специальных инструментов, можно сделать автоматизированным. |
| Перевага №5 | скорочення витрат і часу на розробку та тестування |  |
| Перевага №6 | зниження загальної вартості на підтримку ПЗ впродовж усього терміну служби через раннє виявлення дефектів |  |
| Перевага №7 | покращення комунікацій між членами команди у процесі тестування та розробки |  |
| Обмеження №1 | Поскольку статическое тестирование, чаще всего, выполняется вручную, процесс получается длительным; | Динамическое тестирование – достаточно сложный механизм, выполнение которого требует много времени; |
| Обмеження №2 | Мешает находить уязвимости, находящиеся в среде выполнения. | Динамический метод тестирования это дорогостоящий процесс; |
| Обмеження №3 (і т.д.) |  | В основном данный метод тестирования выполняется по завершению кодирования, и баги находятся уже в процессе реального жизненного цикла разработки. |
| Висновок | Хоть статическое тестирование и долговременный процесс, требующий выделить время на бурные обсуждения и собрания, все же стоит его найти, поскольку очень важно предотвратить возникновение багов на последних этапах создания продукта. Именно поэтому статическое тестирование считают важным этапом на дороге к разработке программного обеспечения без дефектов.  Но не стоит забывать и о важности динамического тестирования. (производительность, функционал, надежность). Тестируя ПО, команда сможет проверить создаваемый продукт на качество и эффективность. | |

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.
5. Є псевдокод: Switch PC on -> Start MS Word -> IF MS Word starts THEN -> Write a poem -> Close MS Word.

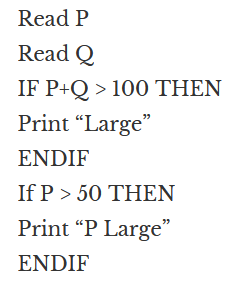
Скільки тест кейсів знадобиться, щоб перевірити його функціонал?

1 – для покриття операторів, 2 – для покриття рішень

1 – для покриття операторів, 1 – для покриття рішень

2 – для покриття операторів, 2 – для покриття рішень

2 – для покриття операторів, 1 – для покриття рішень

1. Скільки потрібно тестів для перевірки тверджень коду: 

2

1

3

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*

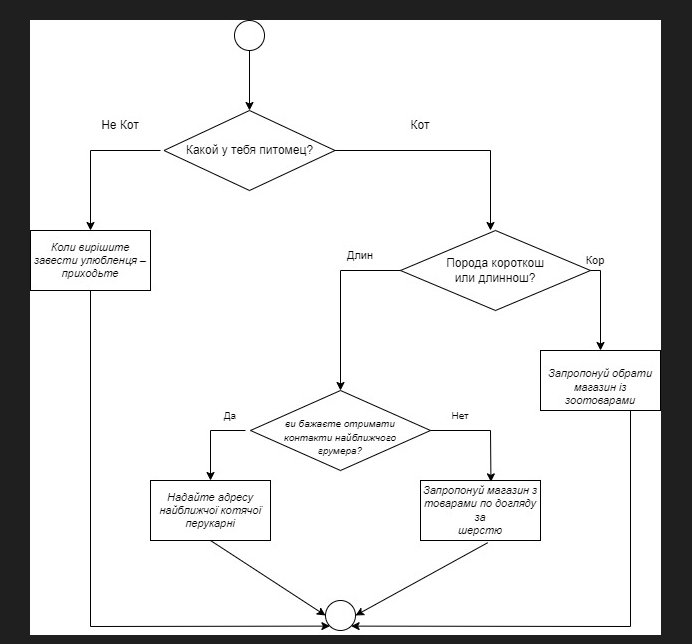
*Скажи “Коли вирішите завести улюбленця – приходьте”*

*Закінчити*

Завдання:

Намалюй схему алгоритму (в інструменті на вибір, наприклад, у вбудованому Google Docs редакторі, [figjam](https://www.figma.com/figjam/) чи [miro](https://miro.com/))

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, щоб переконатися, що

а) всі запитання були поставлені – 3

б) всі комбінації були пройдені – 4

в) всі відповіді були отримані - 4