**1lvl**

|  |  |  |
| --- | --- | --- |
|  | Статистична техніка тестування | Динамічна техніка тестування |
| Основна інформація | Використовується для аналізу програмного коду та документації без його активного виконання. | Передбачає запуск програмного коду. Тобто поведінка програми аналізується під час її роботи. |
| Перевага №1 | Дозволяє виявляти проблеми в програмі на ранніх етапах розробки | Дозволяє виявляти помилки, які виникають тільки під час виконання програми з реальними даними. |
| Перевага №2 | Може охоплювати всі можливі шляхи в коді, які можуть бути пропущені під час динамічного тестування | Дозволяє перевіряти, чи працює програма правильно з погляду коректності функцій та алгоритмів. |
| Перевага №3 | Ефективно для аналізу складних систем. Статичне тестування дозволяє аналізувати складний код та залежності між компонентами без фактичного запуску програми. | Дозволяє перевірити взаємодію програми з користувачем та інтерактивні функції. |
| Перевага №4 | Зниження загальної вартості на підтримку ПЗ впродовж усього терміну служби  через раннє виявлення дефектів; | Дозволяє оцінити продуктивність програми під час реального виконання |
| Перевага №5 | Скорочення витрат і часу на розробку та тестування; | Перевірка взаємодії різних компонентів програми під час їх взаємодії. |
| Обмеження №1 | Не може виявити помилки, пов'язані з динамічними аспектами програми | Не всі можливі шляхи в програмі можуть бути вичерпно протестовані. |
| Обмеження №2 | Потрібні спеціальні інструменти та знання для тестування. | Потрібне середовище, де можна виконати програму з тестовими даними, що може бути складним або дорогим. |
| Обмеження №3 | Можливі труднощі в аналізі складних програм з великою кількістю взаємозв'язків. | Може займати багато часу для великих або складних програм. |
| Обмеження №4 | Вимагає знання мови програмування | Іноді результати динамічного тестування можуть бути залежними від умов виконання тестів(конфігурація тестового середовища) |
| Обмеження №5 | Потребує періодичної актуалізації при внесенні змін у код програми або використання нових версій | Повинні бути регулярно оновлювані під час розвитку проекту, оскільки зміни в коді можуть призвести до дефектів у попередньо успішних тестах. |

**2lvl**

1. 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. 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

В даному випадку, щоб покрити всі оператори потрібно виконати код 1 раз, тому тільки 1 тест кейс

Щоб покрити рішення “IF MS Word starts THEN” потрібен 1 тест кейс щоб перевірити умову “THEN”, тому що для покриття другого рішення не потрібно запускати MS Word.

1. Скільки потрібно тестів для перевірки тверджень коду:

Read P

Read Q

IF P+Q > 100 THEN

Print “Large”

ENDIF

If P > 50 THEN

Print “P Large”

ENDIF

1. 2
2. 1
3. 3
4. 4

P = 40, Q = 70 (40 + 70 > 100) - виконується перше твердження, “Large” буде виведено.

P = 50, Q = 50 (50 + 50 <= 100) - не виконується перше твердження.

P = 60 - виконується друге твердження, “P Large” буде виведено.

P = 40 - не виконується друге твердження.