| 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** | Тестування виконуються без запуску програми і зазвичай включають аналіз коду, документів та інших артефактів проекту. | Тестування виконуються під час роботи програми і включають запуск програми та перевірку її роботи. | | Перевага №1 | Підвищення розуміння іншими членами команди: Статичне тестування може служити як засіб для передачі знань іншим членам команди, допомагаючи їм краще розуміти код і виявляти потенційні проблеми. | Виявлення дефектів, що виникають під час виконання програми: Динамічне тестування дозволяє виявити дефекти, які виникають під час виконання програми, такі як помилки вводу-виводу, проблеми з пам'яттю або некоректна реакція на реальний вхід. | | Перевага №2 | Покращення якості коду: Статичне тестування може допомогти виявити проблеми з якістю коду, такі як недотримання стандартів кодування, погано зрозумілу логіку або неправильну взаємодію між компонентами. | Перевірка функціональності: Динамічне тестування дозволяє перевірити, чи виконує програма очікувану функціональність. Це допомагає впевнитися, що програма працює так, як очікується. | | Перевага №3 | Оптимізація витрат на тестування: Статичне тестування може бути ефективним і економічним способом виявлення дефектів, оскільки воно не вимагає виконання коду або витрат на інфраструктуру тестового середовища. | Виявлення проблем з продуктивністю: Динамічне тестування дозволяє оцінити продуктивність програми, виявити проблеми з швидкодією, нересурсом або поганою оптимізацією. | | Перевага №4 | Виявлення проблем з безпекою: Статичне тестування може допомогти виявити потенційні проблеми з безпекою, такі як уразливості, недостатні права доступу або недостатні заходи безпеки. | Валідація інтеграції: Динамічне тестування дозволяє перевірити, чи працюють різні компоненти програми разом правильно, виявити проблеми з інтеграцією та взаємодією. | | Перевага №5 | Виявлення дефектів, які важко відтворити: Статичне тестування може виявити дефекти, які можуть бути складні або неможливі відтворити в ході динамічного тестування. | Автоматизація: Динамічне тестування може бути автоматизоване за допомогою спеціальних інструментів, що дозволяє автоматично виконувати тести, знижуючи зусилля і час, необхідні для тестування. | | Перевага №6 | Виявлення дефектів раніше: Статичне тестування може бути проведене на ранніх етапах життєвого циклу розробки, що дозволяє виявити дефекти на початкових стадіях, коли виправлення проблем менш витратне. | Виявлення дефектів, що залежать від конкретного середовища: Динамічне тестування дозволяє виявити дефекти, які залежать від конкретного середовища, такі як проблеми зі сумісністю, залежностями від операційної системи або конфігурації. | | Обмеження №1 | Неможливість виявлення дефектів, пов'язаних з виконанням програми: Статичне тестування не може виявити дефекти, які виникають під час виконання програми, такі як помилки вводу-виводу, проблеми з пам'яттю або некоректна реакція на реальний вхід. | Вимога наявності виконуваного коду: Для проведення динамічного тестування необхідно мати доступ до виконуваного коду програми. Це може бути проблемою, якщо вихідний код програми недоступний або захищений авторськими правами. | | Обмеження №2 | Обмежена ефективність для складних систем: Статичне тестування може бути обмежене в ефективності для дуже великих або складних систем, оскільки воно може вимагати значно більше зусиль і часу для перегляду всього коду. | Обмежена покриття коду: Динамічне тестування може не виявити всі можливі дефекти або проблеми в програмі. Це залежить від якості тестових сценаріїв та того, наскільки добре вони покривають різні шляхи виконання програми. | | Обмеження №3 | Залежність від людського фактору: Статичне тестування часто залежить від експертизи тестувальників і їх здатності виявляти потенційні проблеми. Це може призвести до пропуску дефектів, якщо тестувальник пропускає щось або не розуміє певні аспекти коду. | Залежність від тестових данних: Для динамічного тестування необхідно мати вхідні дані, які використовуються для виконання програми. Це може бути складно, особливо якщо тестові дані повинні бути реалістичними або покривати різні варіанти виконання. | | Обмеження №4 | Вимога наявності вихідних кодів: Для статичного тестування необхідно мати доступ до вихідних кодів програми. Це може бути проблемою, якщо вихідний код програми недоступний або захищений авторськими правами. | Витратність виконання: Динамічне тестування може бути часо- та ресурсоємним процесом, особливо для великих або складних програм. Виконання тестових сценаріїв може займати багато часу і вимагати великої кількості обчислювальних ресурсів. | | Обмеження №5 | Витратність організації: Статичне тестування може вимагати значних зусиль і ресурсів для організації і проведення. Необхідно мати відповідну інфраструктуру, інструменти та навички для ефективного виконання статичного тестування. | Відсутність абсолютної гарантії: Навіть якщо динамічне тестування пройшло успішно, це не гарантує, що програма не містить дефектів. Відсутність помилок під час тестування не означає, що програма є безпомилковою. | | Обмеження №6 | Необхідність актуалізації: Статичні техніки тестування потребують постійної актуалізації, оскільки код може змінюватися протягом розробки. Це може займати час і зусилля, особливо для великих проектів. | Обмежена можливість репродукції: Деякі дефекти можуть бути важко або неможливо відтворити під час динамічного тестування. Це може ускладнити процес виявлення та виправлення проблем. | | **Висновок:** | Статичні техніки тестування допомагають покращити якість програмного забезпечення, знизити ризики і витрати на виправлення дефектів. Хоча статичні техніки тестування мають свої обмеження, їх використання разом з динамічним тестуванням може допомогти забезпечити комплексне покриття тестування програмного забезпечення. | Динамічна техніка тестування допомагає виявити дефекти, пов'язані з реальним виконанням програми, перевірити функціональність, інтеграцію та продуктивність, а також забезпечити автоматизацію тестування. Використання динамічних технік разом зі статичними техніками допомагає забезпечити комплексне тестування програмного забезпечення. вона все ще є важливою складовою процесу тестування програмного забезпечення і допомагає виявляти дефекти, перевіряти функціональність та забезпечувати якість програм. | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 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  1. Скільки потрібно тестів для перевірки тверджень коду:     1. 2    2. 1    3. 3    4. 4 |