1.1

Kiểm thử phần mềm là gì?

Kiểm thử phần mềm (Software Testing) là quá trình kiểm tra và đánh giá hệ thống hoặc ứng dụng để đảm bảo rằng nó hoạt động đúng theo yêu cầu, đáp ứng các tiêu chuẩn chất lượng và không có lỗi nghiêm trọng. Mục tiêu của kiểm thử là:

- **Phát hiện lỗi** và khắc phục trước khi phát hành sản phẩm đến người dùng.
- **Đảm bảo chất lượng**: Kiểm thử giúp đảm bảo rằng hệ thống hoạt động ổn định, an toàn và đáp ứng các yêu cầu chức năng lẫn phi chức năng.
- **Tăng cường độ tin cậy**: Một hệ thống được kiểm thử kỹ lưỡng sẽ giúp tăng cường niềm tin của người dùng và khách hàng đối với sản phẩm.

Các loại kiểm thử phần mềm chính

1. **Kiểm thử hộp đen (Black Box Testing)**: Tập trung vào kiểm thử các chức năng của hệ thống mà không quan tâm đến mã nguồn bên trong. Ví dụ: kiểm thử tính năng, kiểm thử giao diện người dùng.
2. **Kiểm thử hộp trắng (White Box Testing)**: Đánh giá các phần bên trong của mã nguồn, kiểm tra dòng chảy dữ liệu, luồng điều khiển và logic của chương trình.
3. **Kiểm thử đơn vị (Unit Testing)**: Kiểm thử từng thành phần nhỏ nhất của mã nguồn (như hàm, phương thức) để đảm bảo chúng hoạt động đúng.
4. **Kiểm thử tích hợp (Integration Testing)**: Đảm bảo các thành phần riêng lẻ hoạt động tốt khi kết hợp với nhau.
5. **Kiểm thử hệ thống (System Testing)**: Kiểm tra toàn bộ hệ thống sau khi tích hợp để đảm bảo hệ thống đáp ứng yêu cầu ban đầu.
6. **Kiểm thử chấp nhận (Acceptance Testing)**: Xác minh xem hệ thống có phù hợp với yêu cầu của khách hàng và người dùng cuối hay không.

Rủi ro nếu hệ thống hoặc dịch vụ phần mềm không được kiểm thử đầy đủ

Nếu phần mềm không được kiểm thử đầy đủ, có thể xảy ra các rủi ro sau:

1. **Lỗi và sự cố hoạt động**:
    - Phần mềm chưa kiểm thử đầy đủ thường có khả năng chứa lỗi, gây ra các lỗi bất ngờ trong quá trình sử dụng.
    - Các lỗi nghiêm trọng có thể dẫn đến sự cố hệ thống, làm gián đoạn dịch vụ hoặc gây thiệt hại về dữ liệu.
2. **Rủi ro bảo mật**:

- Nếu không kiểm thử kỹ lưỡng, hệ thống có thể dễ dàng bị khai thác các lỗ hổng bảo mật, dẫn đến các cuộc tấn công mạng, mất mát dữ liệu nhạy cảm hoặc quyền truy cập trái phép.
- Điều này đặc biệt nghiêm trọng với các phần mềm tài chính, y tế, hoặc các hệ thống chứa dữ liệu nhạy cảm.

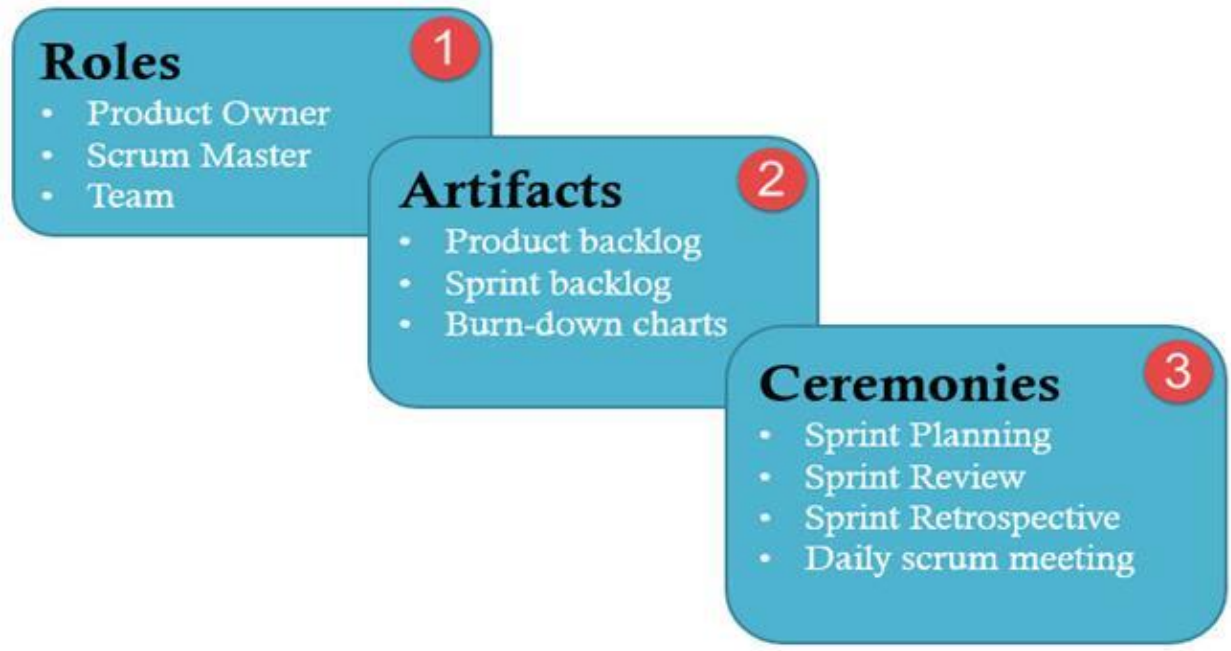
3. **Mất uy tín và lòng tin của khách hàng**:
   - Phần mềm không hoạt động đúng hoặc gây phiền toái cho người dùng sẽ ảnh hưởng đến uy tín của công ty, làm giảm lòng tin của khách hàng.
   - Trong một số trường hợp, điều này có thể dẫn đến mất khách hàng và giảm doanh thu.
4. **Chi phí bảo trì và sửa lỗi cao hơn**:
   - Khi phần mềm không được kiểm thử đầy đủ trước khi phát hành, lỗi sẽ xuất hiện trong quá trình sử dụng. Việc phát hiện và sửa lỗi trong giai đoạn này thường tốn kém nhiều thời gian và chi phí hơn so với giai đoạn phát triển.
   - Các lỗi nghiêm trọng có thể đòi hỏi phải cập nhật hoặc tái cấu trúc phần mềm, tăng chi phí bảo trì.
5. **Không đáp ứng được yêu cầu người dùng**:
   - Phần mềm có thể không đáp ứng đúng các yêu cầu chức năng hoặc phi chức năng (như hiệu suất, độ ổn định, khả năng mở rộng), dẫn đến sự không hài lòng từ phía người dùng.
6. **Rủi ro pháp lý và tuân thủ**:
   - Đối với một số ngành (như y tế, tài chính), phần mềm cần phải tuân thủ các tiêu chuẩn pháp lý. Nếu phần mềm không được kiểm thử đầy đủ và không đáp ứng các tiêu chuẩn này, công ty có thể gặp phải các vấn đề pháp lý và bị phạt.

1.2

- Scrum là một mô hình quản lý và kiểm soát chia cắt độ phức tạp để tập trung vào việc xây dựng phần mềm đáp ứng nhu cầu kinh doanh. Scrum làm tăng đáng kể năng suất và giảm thời gian hơn so với các mô hình cổ điển như "waterfall". Mô hình Scrum cho phép các tổ chức điều chỉnh trơn tru, nhanh chóng thay đổi các yêu cầu, và sản xuất để đáp ứng phát triển mục tiêu kinh doanh.
- Scrum đã rút ngắn lịch trình của chu kỳ phát hành bằng việc điều chỉnh scope gọi là các sprints. Mỗi phiên bản có thể có nhiều sprints Mỗi dự án Scrum có thể có nhiều chu kỳ phát hành (Release Cycles):

- Là 1 trình tự lặp đi lặp lại các cuộc họp, sự kiện và milestones.
- Là 1 quá trình testing và cài đặt các yêu cầu mới, như là stories, để đảm bảo răng có 1 số công việc được release sau mỗi each sprint.

Scrum được dựa trên 3 yếu tố sau:

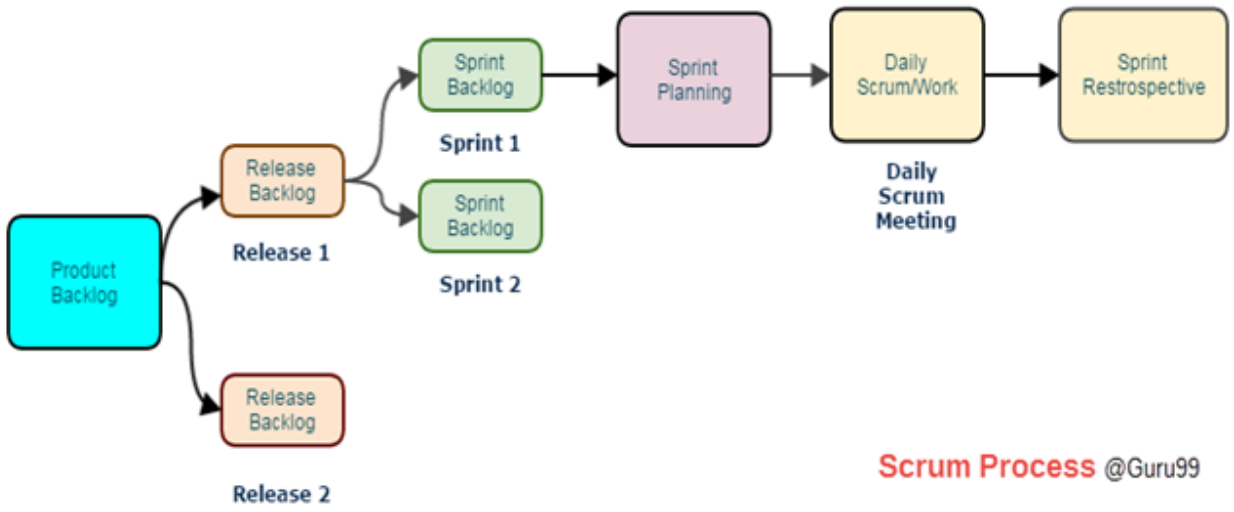


1.Vai trò trong Scrum:

Có ba vai trò chính trong Scrum - Product Owner, Scrum Master và Development Team.

Product Owner	Scrum Master	Nhóm phát triển
Định nghĩa các tính năng của sản phẩm	Quản lý team và kiểm soát năng suất của team	Thường có khoảng 5-9 thành viên
Quyết định ngày phát hành và các tính năng tương ứng	Duy trì danh sách chặn và loại bỏ các rào cản trong quá trình phát triển	Bao gồm các developers, designer và đôi khi là testers, v.v.
Ưu tiên các tính năng theo giá trị thị trường và lợi nhuận của sản phẩm	Phối hợp với team dưới tất cả các vai trò và chức năng	Tự tổ chức và lên lịch cho công việc của họ
Chịu trách nhiệm về lợi nhuận của sản phẩm	Bảo vệ đội khỏi các can thiệp bên ngoài	Có quyền làm mọi thứ trong ranh giới của dự án để đáp ứng mục tiêu của sprint
Có thể chấp nhận hoặc từ chối kết quả đã yêu cầu	Tổ chức daily scrum, xem xét sprint và lên kế hoạch cho các cuộc họp	Tích cực tham gia các tiến trình *(ceremonies)* hà

2.Scrum Artifacts



Quy trình scrum bao gồm:

- User stories: là lời giải thích ngắn gọn của chức năng cần được test.

- Product Backlog: là tập hợp các User stories, được quản lý và đánh giá độ ưu tiên bởi product owner. Tất cả các thành viên khác đều có thể add thêm vào nó dưới sự kiểm soát của product owner.
- Release Backlog: là thời gian được đưa ra để quyết định ngày release sản phẩm.
- Sprints: là khoảng thời gian hoàn thành user stories. Nó được quyết định bởi product owner và developer team, thông thường từ 2-4 tuần.
- Sprint Backlog: Đó là một tập hợp các user stories được hoàn thành trong 1 sprint.
- Block List: Đây là một danh sách các block và các quyết định thuộc sở hữu của scrum master và được cập nhật hàng ngày.
- Burndown chart: Burn-down chart đại diện cho tiến độ chung của các tiến trình và công việc được hoàn thành trong suốt quá trình.

3.Ceremonies (Processes) in Scrum

- Sprint Planning: 1 sprint bắt đầu với từ release backlog trong sprint backlog. Nó được tổ chức bởi scrum master. Testers ước lượng nỗ lực để test vài stories trong the Sprint Backlog.
- Daily Scrum: Nó được tổ chức bởi scrum master, kéo dài khoảng 15 phút. Trong lúc Daily scrum, các thành viên sẽ thảo luận về các công việc hoàn thành trong ngày hôm trước, công việc và kế hoạch của ngày hôm nay và hôm sau, các vấn đề gặp phải trong các sprints. Tiến độ công việc cũng được theo dõi.
- Sprint Review/ Retrospective: Nó được tổ chức bởi scrum master, nó kéo dài khoảng 2-4 giờ và thảo luận về những gì team đã hoàn thành trong các sprint và những bài học rút ra.

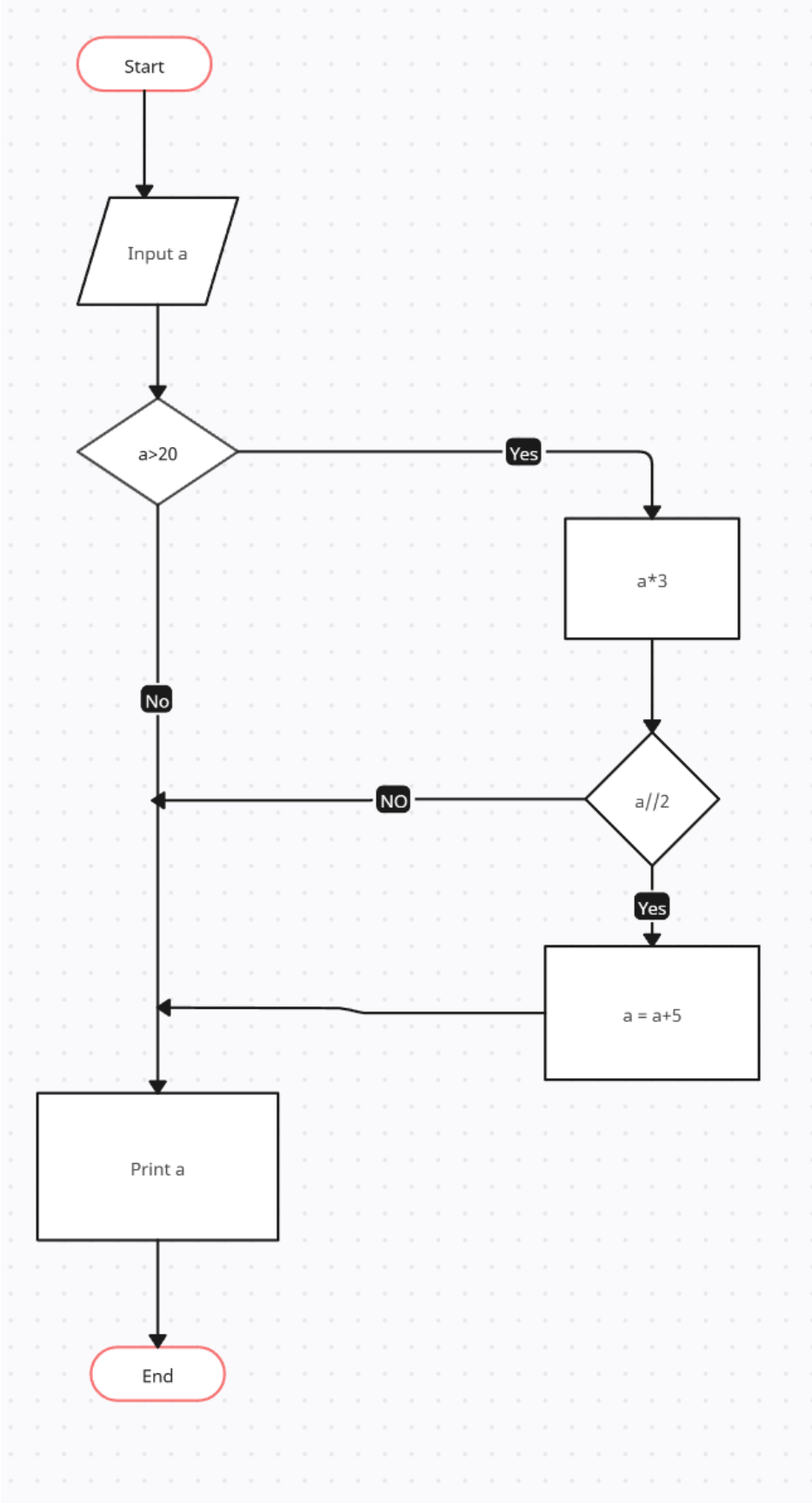
1.3

Kỹ thuật bao phủ quyết định (Decision Coverage) trong kiểm thử hộp trắng là một phương pháp để đảm bảo rằng tất cả các nhánh quyết định trong mã nguồn đã được kiểm thử ít nhất một lần. Phương pháp này tập trung vào các điểm quyết định trong chương trình, thường là các câu lệnh điều kiện hoặc lệnh rẽ nhánh (như if, else, switch, for, và while).

Cách thức hoạt động của Decision Coverage

1. **Xác định các điểm quyết định** trong mã nguồn. Mỗi điểm quyết định có thể tạo ra các kết quả khác nhau, tùy thuộc vào điều kiện trong quyết định đó (true hoặc false).
2. **Thiết lập các trường hợp kiểm thử** sao cho mỗi nhánh (true và false) của từng điểm quyết định đều được thực thi ít nhất một lần.
3. **Đo lường bao phủ quyết định** bằng cách tính tỷ lệ giữa số quyết định đã được kiểm thử với tổng số quyết định trong mã nguồn.

$$Decision\ Coverage = \frac{Number\ of\ Decision\ Outcomes\ Excercised}{Total\ Number\ of\ Decision\ Outcomes}$$


Start
Input a
a>20
Yes
a*3
No
a//2
NO
Yes
a = a+5
Print a
End

```python
a= int(input('a='))
if a>20:
    a= a*3
    if a%2 ==0:
        a= a+5
print(a)
```

Test Case	Value of A	Output	Decision Coverage
1	5	5	33.33%
2	30	95	33.33%
3	42	99	33.33%

2.1

1. Tự động kiểm thử trong Scrum

- Mục tiêu chính:
  - Đảm bảo tính năng được phát triển trong mỗi Sprint hoạt động đúng như kỳ vọng.
  - Tăng tốc độ kiểm thử hồi quy khi phần mềm được cải tiến liên tục.
  - Hỗ trợ DevOps qua tích hợp liên tục (CI/CD).
- Lợi ích:
  - Phát hiện lỗi sớm trong quá trình phát triển.
  - Tăng độ tin cậy của phần mềm trong các đợt phát hành thường xuyên.

---

2. Tích hợp Selenium trong Scrum Testing

Môi trường:

- Các trình duyệt được hỗ trợ (Chrome, Firefox, Edge).
- Môi trường chạy kiểm thử (local hoặc cloud như Selenium Grid, BrowserStack).

2.2

https://youtu.be/DfqBi1K2V3Y

https://youtu.be/-t0LuXaR5eQ