Odpowiedzi na pytania:

1. Czym się różnią testy funkcjonalne od niefunkcjonalnych?

Testy funkcjonalne sprawdzają jak sama nazwa wskazuje funkcje które realizuje system zaś testy niefunkcjonalne testują charakterystykę oprogramowania np. ergonomie, wydajność. W skrócie można powiedzieć że testy funkcjonalne sprawdzają czy funkcjonalność którą ma realizować system działa a testy niefunkcjonalne jak działa cały system.

2. Co to są 'smoke testy' i 'testy regresji'? Kiedy je stosujemy?

Testy regresji mają na celu weryfikacje istniejącej funkcjonalności systemu po jego modyfikacji czyli sprawdzamy czy zmiany nic nie „uszkodziły". Stosujemy je gdy nastąpi jakaś zmiana w systemie np. usuniecie/dodanie nowych funkcji, poprawianie błędów. Należy odpowiednio szacować jaki zakres systemu ma być testowany, spróbować znaleźć złoty środek miedzy kosztami(czasem) a tym aby nie przeoczyć istotnego błędu.

Przeznaczeniem smoke testów jest sprawdzenie każdej części sytemu bez zagłębiania się w logikę jego działania. Smoke testy określają czy możliwe jest wykonanie testów poszczególnych części systemu, wykonujemy je wiec zawsze na początku procesu testowania systemu na podstawie listy przypadków testowych lub przy użyciu odpowiednich narzędzi do testów automatycznych.

3. Co jest celem testowania?

Celem testowania jest zminimalizowanie ryzyka wystąpienia błędu czyli uzyskanie maksymalnej (ograniczonej przez koszty czyli czas potrzebny na testowanie) pewności że oprogramowanie działa zgodnie z założeniami.

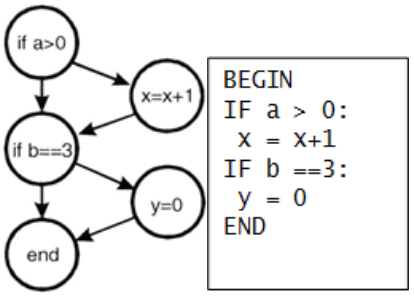
4. Jak tester może się upewnić, że błąd został naprawiony?

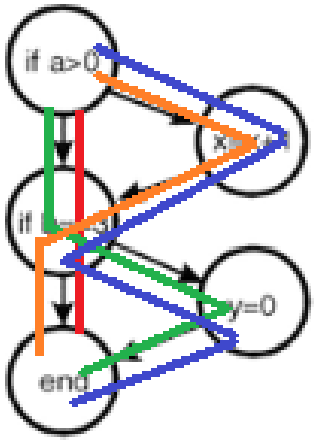
Aby sprawdzić czy błąd został naprawiony należy wykonać retesty czyli powtórne wykonanie przypadków testowych które wykazały błędy

5. Testujesz aplikacje termometr która wykonuje pomiar temperatury. Co byś zrobił aby przetestować zachowanie aplikacji przy skrajnych wartościach -50C i 200C ?

Jak rozumiem chodzi o testowanie aplikacji nie samego urządzenia dokonującego pomiaru temperatury. Ponieważ w warunkach biurowych ciężkie by było uzyskanie takich temperatur należałoby sprawdzić w jakiś sposób przekazywane są dane z miernika temperatury i na tej podstawie stworzyć środowisko symulujące miernik i przekazujące do programu dane w tym podana przez nas wartości temperatury.

6. Ile przypadków testowych potrzeba, aby pokryć wszystkie możliwości?





Potrzeba 4 przypadków testowych:

a. Dane Wejściowe: a>0 i b==3,
b. Dane Wejściowe: a>0 i b!=3,
c. Dane Wejściowe: a<0 i b==3,
d. Dane Wejściowe: a<0 i b!=3.

7. Dany jest input „wiek”, który przyjmuje wartości od 18 do 60. Twoim zadaniem jest przetestować go za pomocą techniki wartości brzegowych. Jakie wartości wpisujesz do inputu? Podaj wszystkie liczby, które wpisujesz.

Zakładając że są to wartości tylko całkowite będą to: 17,18,19,59,60,61.

8. Dołączasz do projektu w trakcie develepmentu aplikacji, do której nie ma dokumentacji. Jakie pytania zadasz analitykowi, zanim przystąpisz do testów logowania?

Zakładając że testy odbywałyby się metodą czarnej skrzynki spytałbym o:

- bazę loginów które mógłbym wykorzystać do testowania,
- jakie znaki dopuszczalne w poprawnych loginach i hasłach,
- czy istnieją jakieś wymogi ich dotyczące (minimalna ilość znaków, musi zawierać cyfry bądź znaki specjalne),
- czy występuje blokada formularza bo określonej liczbie niepoprawnych logowań jeśli tak to na jakiej zasadzie działa i jak ją zdjąć;
- walidację pól, czy możliwe jest wpisanie niepoprawnych znaków, jeżeli tak czy zostaną one zweryfikowanie przed wysłaniem do bazy (można by sprawdzić odporność na wstrzykiwanie kodu),
- jakie komunikaty wyświetlają się przy niepoprawnych próbach logowania, o to na ile są „user friendly” czyli czy pojawi się informacja „niepoprawny login/hasło” czy po prostu „niepoprawne dane”,
- po poprawnym zalogowaniu jak wygląda dostęp do funkcjonalności wymagającej uwierzytelnienia, po jakim czasie uwierzytelnienie traci ważność, czy po zamknięciu aplikacji użytkownik zostaje zalogowany, czy zostanie wylogowany po jakim czasie bezczynności.

Więcej pytań pojawiło by się na pewno w trakcie trwania testów ☺

9. Czym się różni metoda GET od POST?

Metoda GET przekazuje dane poprzez adres URI (np. *../index.php?arg=1*) a metoda POST przekazuje dane przez nagłówek HTTP.

10. Czy HTTP jest protokołem zmiennostanowym?

Protokół HTTP jest protokołem bezstanowym, po zakończeniu transakcji z klientem nie przechowuje żadnych informacji o tej transakcji.

11. Czym różni się LEFT JOIN od INNER JOIN?

Wynikiem INNER JOIN będą wiersze z tabel które spełniają warunek, te wiersze które nie zostaną „połączone" nie pojawią się wyniku. W LEFT JOIN uwzględnia w wyniku dane z pierwszej tabeli (lewej) której nie posiadają odpowiedników w drugiej tabeli. Puste kolumny z drugiej tabeli w takim przypadku zostaną wypełnione wartościami NULL

12. W jakim katalogu, standardowo Linux trzyma pliki konfiguracyjne:

Odpowiedz: /etc

13. Jak przetestowałbyś bashową komendę cp? (argumenty funkcji można pominąć)

Zakładając że pomijamy opcje(przełączniki) wykonałbym następujące przypadki testowe:

- skopiowanie istniejącego pliku bez podawania drugiego argumentu,
- skopiowanie nieistniejącego pliku bez podawania drugiego argumentu,
- skopiowanie istniejącego pliku do istniejącego katalogu,
- skopiowanie istniejącego pliku do nieistniejącego katalogu,
- skopiowanie nieistniejącego pliku do istniejącego katalogu,
- skopiowanie nieistniejącego pliku do nieistniejącego katalogu.

Powyższe kombinacje wykonałbym dodatkowo próbując nadpisać jeden plik na drugi (nadpisanie istniejącego/nieistniejącego pliku na istniejący/nieistniejący itd.).

Przetestowałbym również powyższe przypadki testowe dla plików/folderów do których użytkownik nie ma uprawnień oraz dla różnych zapisów plików/folderów (pełnych ścieżek, samych nazw, odpowiednich i nieodpowiednich zapisów dla katalogów nadrzędnych i podrzędnych).