7 zasad testowania oprogramowania

1. Testowanie ujawnia usterki

Testowanie może wykazać obecność defektów, ale nie może dowieść, że oprogramowanie jest od nich wolne. Tym samym testowanie zmniejsza prawdopodobieństwo, że w oprogramowaniu pozostaną niewykryte defekty, ale sam fakt niewykrycia defektów nie stanowi dowodu poprawności działania oprogramowania. Niemal na pewno testy nie uwzględniły wszystkich przypadków.

2. Testowanie gruntowne jest niemożliwe

Przetestowanie wszystkiego (tj. wszystkich kombinacji danych wejściowych i warunków wstępnych) jest możliwe tylko w najprostszych przypadkach. Gdybyśmy starali się przewidzieć i sprawdzić wszystkie możliwe przypadki, niepotrzebnie stacilibyśmy czas i być może nigdy nie zakończylibyśmy testowania. W związku z tym, zamiast podejmować próbę testowania gruntownego, należy odpowiednio ukierunkować wysiłki związane z testowaniem na zastosowanie analizy ryzyka, technik testowania i priorytetyzacji.

3. Wczesne testowanie oszczędza czas i pieniądze

Aby wcześnie wykryć defekty, należy rozpocząć testowanie jak najwcześniejszym etapie cyklu życia oprogramowania. Wczesne testowanie jest niekiedy nazywane „przesunięciem w lewo” (ang. shift left). Wykonywanie testów na wczesnym etapie cyklu życia oprogramowania pozwala ograniczyć lub wyeliminować kosztowne zmiany.

4. Kumulowanie się defektów

Zwykle większość defektów wykrytych podczas testowania przed przekazaniem oprogramowania do eksploatacji lub większość awarii występujących w fazie eksploatacji występuje lub ma swoje źródło w niewielkiej liczbie modułów.

Możemy zaobserwować tutaj działanie **zasady Pareto (80/20)**, czyli na 10 testowanych funkcjonalności, 8 zwykle działa prawidłowo, a 2 nieprawidłowo. Jednak jeżeli firma wprowadzi na rynek wadliwą aplikacje to 80% klientów zauważy usterki. Dlatego niezwykle ważne jest zlokalizowanie wadliwych modułów, które mogą być skupiskiem defektów aplikacji.

5. Paradoks pestycydów

Ciągłe powtarzanie tych samych testów prowadzi do sytuacji, w której przestają one w pewnym momencie wykrywać nowe defekty. Aby móc wykrywać nowe defekty, może być konieczne zmodyfikowanie dotychczasowych testów i danych testowych, a także napisanie nowych testów. Niezmieniane testy tracą z czasem zdolność do wykrywania defektów, podobnie jak pestycydy po pewnym czasie nie są zdolne do eliminowania szkodników. Aby przezwyciężyć ten paradoks należy regularnie przeglądać i modyfikować przypadki testowe (lista czynności do wykonania w celu sprawdzenia prawidłowości działania funkcjonalności).

6. Testowanie zależy od kontekstu

Testowanie wykonuje się w różny sposób w różnych kontekstach. Na przykład oprogramowanie sterujące systemami przemysłowymi, które jest krytyczne ze względów bezpieczeństwa, testuje się inaczej niż aplikację mobilną sklepu internetowego. Innym przykładem może być odmienny sposób przeprowadzania testów w projektach zwinnych i w tych prowadzonych zgodnie z modelem sekwencyjnym cyklu życia.

7. Przekonanie o braku błędów jest błędem

Nie należy oczekiwać, że jeśli oprogramowanie zostało gruntownie przetestowane i wyeliminowano znalezione dzięki temu błędy, na pewno jest już poprawne i będzie w pełni użyteczne. Przede wszystkim wciąż może zawierać usterki, których nie dało się wykryć. Należy jednak pamiętać, że nawet znalezienie i usunięcie wszystkich błędów na nic się nie zda, jeśli system będzie nieintuicyjny i trudny w obsłudze albo nie będzie spełniał potrzeb i oczekiwać użytkownika.