*Intro to testing*

Il testing è la valutazione dei prodotti elaborati durante la produzione del software. È il prrocesso tramite cui si scoprono i difetti del sistema.

Fa in modo che il software faccia tutto ciò che ci si aspetta (oppure ad assicurarsi che faccia tuttociò che deve e niente di più.)

**Failure**: comportamento inaspettato del sistema

**Bug** == Difetto: qualcosa che “va storto” nel codice, e che causa una failure.

Quality assurance: mira a prevenire i difetti e a rimuovere quelli già esistenti, mentre il testing mira solo a scoprirli.

*Test early, often and the right art the right way.*

Test case: componente base del test, coppia = (input, expected result).

Input: info sullo stato del sistema o valori da processare

Expected Result: spesso include cambiamenti non percettibili per l’utente

Test suite: organizzano i test case, ad esempio li raggruppano per tipo

*COVERAGE*

Misura dell’adeguatezza del testing che osserva la quantità di codice(percentage of lines) coperto da casi di test oppure in termini di quanti requisiti sono valutati nelle specifiche e nei test.

Black box testing: basandosi sulle specifiche

White box testing: basandosi sull’implementazione

JUnit: framework opensource di testing per java

Code coverage

* Optimistic: coverage == 100% && failures == errors == 0 => è possible che ci siano bug
* Pessimistic: coverage == 100% && failures != 0 => è posibile che non ci siano bug (errore nel test)
* Impossbile: coverage < 100%

Impossible Coverage

Il progetto è un counter grafico.

Il testing avviene sulla condizione iniziale della variabile contatore ed è possibile visualizzarne il risultato solamente alla chiusura dello stage dell’applicativo. Dunque, lanciando il counter e chiudendo immediatamente la finestra, il test risulta “passed”, ma la coverage non è del 100%, poiché non viene eseguito il metodo associato al setOnAction del bottone “+”.

Se invece, una volta lanciato l’applicativo, si andasse ad utilizzare il bottone “+”, per aumentare il contatore, alla chiusura della finestra, il testing darebbe come risultato “failed”, ma una coverage del 100%.

Il testing del solo metodo associato al bottone non è possibile in quanto “private”, impedendo dunque anche l’eventuale tentativo di confronto con un’altra variabile tale da tenere a sua volta il conto di quante volte si è incrementato il counter.

Optimistic Coverage

Il progetto prevede il calcolo della media aritmetica fra due double.

Nonostante la coverage del 100%, il codice dell’applicativo non è privo di bug in quanto il risultato restituito è erroneamente sempre “10.0”, tuttavia il caso di testing risulta “passed”, poiché effettuato su una delle tante coppie di valori la cui media è “10.0”.

Pessimistic Coverage

Il progetto permette di invertire una qualsiasi stringa.

Nonostante la coverage del 100%, ed il fatto che il codice dell’applicativo sia privo di bug, il caso di testing risulta “failed”, poiché erroneamente eseguito: la stringa expected dovrebbe essere “ittut a oaic” e non “ittut a oiac”.