Software Testing and Validation
A.A. 2022/2023
Testo del Progetto

Igor Melatti

Come si consegna

Il presente documento descrive le specifiche per il progetto d'esame. La consegna deve consistere in un singolo file `STV_2022_2023_matricole.zip` (se il progetto è fatto in gruppo, scrivere tutte le matricole separate dall'underscore _), che contenga un'unica directory `STV_2022_2023_matricole`, la quale a sua volta deve includere:

- un file PDF `relazione.pdf` con le seguenti caratteristiche:
  - deve indicare nome, cognome e matricola di ogni studente/studentessa del gruppo;
  - deve descrivere come il progetto sia stato svolto;
- un file PDF `presentazione.pdf` che sia la versione con le slide della relazione;
- tutti i file che fanno parte del progetto, con un'opportuna organizzazione in directory.

Il suddetto file `STV_2022_2023_matricole.zip` andrà poi inviato per email al docente `igor.melatti@univaq.it`.

È possibile consultarsi con i compagni. Tuttavia, occorre che ciascun gruppo presenti una propria soluzione personale.

Esercizio per Gruppi da 2 Studenti/Studentesse

Considerando gli esempi dati nelle lezioni sul testing funzionale (collasso spazi, collasso k spazi, radici di un'equazione di secondo grado), fornire il testing completo in ogni sua parte:

1. Implementazione per il collasso di k spazi.
2. Definizione dei test case specifications, sia come singoli test factors che nella versione combinata:
   - è possibile sia riusare che ignorare gli esempi di test case specifications dati a lezione;
   - qualora si usi un tool esterno (ad es., `pict`), indicare come viene invocato.
3. Generazione dei test case:
   - scrivere un algoritmo ed un piccolo ma funzionante programma che, a partire dalle combinazioni di test factors, generi un corrispondente test case.
4. Esecuzione dei test case:
   - è consigliabile usare `JUnit`
   - per la funzione sulle equazioni di secondo grado, far vedere l'esecuzione sia prima che dopo la correzione dell'errore.
5. Valutazione della test adequacy rispetto alle coverage viste a lezione:
   - istrumentare opportunamente il codice.
6. Uso di un model checker tra quelli visti a lezione per la generazione di test case aggiuntivi (fine pag. 328 del libro di testo) per il loop boundary adequacy criterion nel caso dei k spazi consecutivi.
   - costruire il CFG
   - modellarlo nel model checker
   - aggiungere una parte che generi tutti i possibili input con determinati limiti:
     - lunghezza stringa massimo n caratteri
     - set limitato di caratteri nella stringa
       - ad es.: a, b, A, B, spazio, andata a capo
     - massimo numero di spazi consecutivi m
     - massimo numero di occorrenze di spazi consecutivi l
       - con n, m, l parametri del modello...
   - eseguire la verifica in modo da farsi ritornare i test case per il loop boundary criterion come controesempio.

Esercizio per Singoli Studenti/Studentesse

Come sopra, ma facendo il testing della sola funzione sul collasso dei k spazi consecutivi.