Test Plan

Rubrica telefonica

Componenti Gruppo 17:

Di Marino Domenico
Matricola: 0612707421

Adinolfi Giovanni
Matricola: 0612708352

Di Crescenzo Francesco
Matricola: 0612708640

INDICE DEL DOCUMENTO

1.0 Introduzione

Scopo: Lo scopo del test plan è quello di organizzare il processo di testing, specificando quali sono i test da eseguire, le risorse che devono essere utilizzate e i criteri di accettazione. Così facendo è possibile minimizzare i rischi identificando a priori potenziali problemi che potrebbero rivelarsi invece solo dopo il rilascio del prodotto.

Contesto: Il progetto prevede la realizzazione di una rubrica telefonica in grado di creare contatti, eliminarli, aggiungere fino a un massimo di 3 numeri telefonici ed email per ciascun contatto e, tramite interfaccia, importare ed esportare i dati da un file CSV.

2.0 Obiettivi dei test

Pass/Fail criteria: I risultati dei test devono corrispondere esattamente al risultato atteso per essere considerati validi. Un test è considerato superato quando il confronto tra il risultato ottenuto e quello atteso risulta positivo.

Ogni test ha l'obiettivo di verificare che ogni modulo del sistema gestisca correttamente tutti i casi possibili. Sono stati condotti test unitari su ciascun metodo delle classi che costituiscono la rubrica telefonica.

2.1 Unit Test

Di seguito sono riportati i test effettuati durante il progetto:

- Classe Rubrica;
- Classe Contatto;
- Classe Email;
- Classe Telefono.

2.1.1 Classe Rubrica

- Per il metodo **aggiungiContatto** viene testato un caso generico per verificare che le variabili istanziate vengano correttamente aggiunte nella rubrica

- Per il metodo **modificaContatto** sono stati dati input che verificassero la corretta esecuzione nei seguenti casi:
    - nome di un contatto
    - cognome di un contatto
    - numero di un contatto
    - email di un contatto
    - nome e cognome null
    - inserimento di nome null in un contatto con cognome null
    - inserimento di cognome null in un contatto con nome null
    - nome e cognome

- Per il metodo **eliminaContatto** viene testato un caso generico per verificare che le variabili istanziate vengano correttamente eliminate

- Per il metodo **ricercaContatto** viene testato un caso generico per verificare che le variabili istanziate vengano correttamente trovate tramite una sottostringa data in input

- Per il metodo **sort** viene testato un caso generico per verificare che le variabili istanziate vengano correttamente ordinate

- Per il metodo **importaRubrica** sono stati dati input che verificassero la corretta esecuzione nei seguenti casi:
    - file csv con contenuto nel formato previsto dall'applicazione
    - file csv con contenuto diverso dal formato previsto dall'applicazione
    - file diverso da csv

- Per il metodo **esportaRubrica** viene testato un caso generico per verificare che le variabili istanziate vengano correttamente esportate nel file csv

- Per il metodo **getContatti** viene testato un caso generico per verificare che le variabili istanziate vengano correttamente restituite dalla rubrica

- Per il metodo **toString** viene testato un caso generico per verificare che la stampa venisse fatta correttamente

2.1.2 Classe Contatto

- Per il metodo **Constructor** sono stati dati input che verificassero la corretta esecuzione nei seguenti casi:
    - nome, cognome, telefono, email
    - nome e cognome
    - cognome con nome null
    - nome con cognome null
    - valori null
    - nome e cognome null
    - nome null

- Per il metodo **getNome** viene testato un caso generico per verificare che la variabile istanziata venga correttamente restituita dal contatto

- Per il metodo **getCognome** viene testato un caso generico per verificare che la variabile istanziata venga correttamente restituita dal contatto

- Per il metodo **getNumeri** viene testato un caso generico per verificare che la variabile istanziata venga correttamente restituita dal contatto

- Per il metodo **getEmail** viene testato un caso generico per verificare che la variabile istanziata venga correttamente restituita dal contatto

- Per il metodo **setNome** sono stati dati input che verificassero la corretta esecuzione nei seguenti casi:
    - nome
    - nome null con cognome null

- Per il metodo **setCognome** sono stati dati input che verificassero la corretta esecuzione nei seguenti casi:
    - cognome
    - cognome null con nome null

- Per il metodo **aggiungiNumero** sono stati dati input che verificassero la corretta esecuzione nei seguenti casi:
    - aggiunta classica di un numero
    - tentativo di aggiunta di un numero con 3 numeri già presenti

- Per il metodo **aggiungiEmail** sono stati dati input che verificassero la corretta esecuzione nei seguenti casi:
    - inserimento di un'email
    - tentativo di inserire un email in una lista piena

- Per il metodo **compareTo** sono stati dati input che verificassero la corretta esecuzione nei seguenti casi:
    - due contatti con nome e cognome diversi

    - due contatti con lo stesso cognome ma diverso nome
    - due contatti con lo stesso cognome e nome
    - due contatti con lo stesso nome ma diverso cognome
    - due contatti con lo stesso nome, ma cognome assente
    - due contatti con lo stesso cognome, ma nomi assenti
    - due contatti con lo stesso cognome, il primo con nome null
    - due contatti con lo stesso cognome, il secondo con nome null
    - due contatti, il secondo con cognome null
    - due contatti, il primo con cognome null

- Per il metodo **toString** viene testato un caso generico per verificare che la stampa venisse fatta correttamente

- Per il metodo **equals** sono stati dati input che verificassero la corretta esecuzione nei seguenti casi:
    - confronta se due contatti sono uguali
    - confronta che due contatti siano diversi tra loro
    - verifica se il contatto è null
    - confronta due oggetti di classi differenti
    - verifica se due contatti hanno lo stesso nome ma non lo stesso cognome
    - verifica se due oggetti hanno lo stesso nome e cognome, ma uno dei due oggetti è stato modificato dopo la creazione
    - verifica se due oggetti hanno lo stesso nome, cognome e numeri di telefono, ma differiscono per l'email

- Per il metodo **hashCode** viene testato un caso generico per verificare che la stampa venisse fatta correttamente

2.1.3 Classe Email

- Per il metodo **Constructor** viene testato un caso generico per verificare che la variabile istanziata venga correttamente creata

- Per il metodo **aggiungiEmail** sono stati dati input che verificassero la corretta esecuzione nei seguenti casi:
    - inserimento di un'email
    - tentativo di inserire un indirizzo email con già 3 email presenti

- Per il metodo **getSize** viene testato un caso generico per verificare che la variabile istanziata venga correttamente restituita

- Per il metodo **getEmail** viene testato un caso generico per verificare che la/e variabile/i istanziata/e venga/vengano correttamente restituite

- Per il metodo **getSingolaEmail** viene testato un caso generico per verificare che la variabile istanziata venga correttamente restituita

- Per il metodo **setEmail** viene testato un caso generico per verificare che la/e variabile/i istanziata/e venga/vengano correttamente reimpostate

- Per il metodo **toString** viene testato un caso generico per verificare che la stampa venisse fatta correttamente

- Per il metodo **equals** sono stati dati input che verificassero la corretta esecuzione nei seguenti casi:
    - verifica se due email sono uguali
    - confronta una email con un oggetto null
    - confronta un'email con un oggetto differente

2.1.4 Classe Telefono

- Per il metodo **Constructor** viene testato un caso generico per verificare che la variabile istanziata venga correttamente creata

- Per il metodo **aggiungiNumero** sono stati dati input che verificassero la corretta esecuzione nei seguenti casi:
    - inserimento di un numero
    - tentativo di inserire un numero con già 3 numeri presenti

- Per il metodo **getSize** viene testato un caso generico per verificare che la variabile istanziata venga correttamente restituita

- Per il metodo **getNumeri** viene testato un caso generico per verificare che la/e variabile/i istanziata/e venga/vengano correttamente restituite

- Per il metodo **getSingoloNumero** viene testato un caso generico per verificare che la variabile istanziata venga correttamente restituita

- Per il metodo **getSetNumero** viene testato un caso generico per verificare che la/e variabile/i istanziata/e venga/vengano correttamente reimpostate

- Per il metodo **toString** viene testato un caso generico per verificare che la stampa venisse fatta correttamente

- Per il metodo **setNumeri** viene testato un caso generico per verificare che la/e variabile/i istanziata/e venga/vengano correttamente reimpostate

- Per il metodo **equals** sono stati dati input che verificassero la corretta esecuzione nei seguenti casi:
    - verifica se due numeri sono uguali
    - confronta un numero di telefono con un oggetto null
    - confronta numeri di telefono di classi differenti

2.2 Functional Test

I test sono strutturati e utilizzati con lo scopo di verificare il comportamento dei vari componenti del software durante l'esecuzione di determinati scenari.

2.3 Integration Test

Test del sistema completo: L'intero software è stato testato come un unico modulo integrato, risultante dalla combinazione di tutti i singoli moduli precedentemente sottoposti a test individuali.

2.4 Usability Test

Prototype Test: Sono stati condotti test prototipali con la partecipazione di utenti esterni, al fine di ottenere un osservazione imparziale sul funzionamento del software e verificarne l'assenza di errori.

2.5 System Test

Testing dell'intero software dopo aver apportato tutte le correzioni.

2.6 Regression Test

Ridefinizione dei test: I test sono stati rivisti e aggiornati in base ai risultati ottenuti, con l'obiettivo di correggere eventuali errori rilevati, migliorare eventuali imperfezioni e verificare che le modifiche apportate non abbiano introdotto nuovi problemi o compromesso il corretto funzionamento del software.

3.0 Risorse

Le risorse utilizzate per effettuare il testing del software sono: *NetBeans*, *JUnit 5* per l'implementazione dei test automatizzati, e JaCoCo per il code coverage.