**Immagine che contiene elettronica, testo, computer, schermata

Descrizione generata automaticamenteSAFR – ITSS23/24**

*Fractions java class methods from Apache Commons Lang Library*

Componenti:

* Samuel Giovanni Paparella: [s.paparella24@studenti.uniba.it](mailto:s.paparella24@studenti.uniba.it), 764513;
* Francesco Pappagallo: [f.pappagallo5@studenti.uniba.it](mailto:f.pappagallo5@studenti.uniba.it), 758237.

**Indice**

1 Introduzione ………………………………………………………………………………… 3

2 Capitolo 1 ……………………………………………………………………………………. 4

3 Capitolo 2 ………………………………………………………………………………….. 36

1. **Introduzione**

In un mondo sempre più digitale e interconnesso, la qualità del software è diventata un elemento critico per il successo delle aziende e per la soddisfazione degli utenti finali. Effettuare software testing rappresenta è fondamentale per garantire che il software sia affidabile, sicuro e in particolare conforme ai requisiti specificati.

Esploreremo i diversi tipi di test, le tecniche di progettazione dei test, gli strumenti di automazione e le best practice per la gestione dei processi di test.

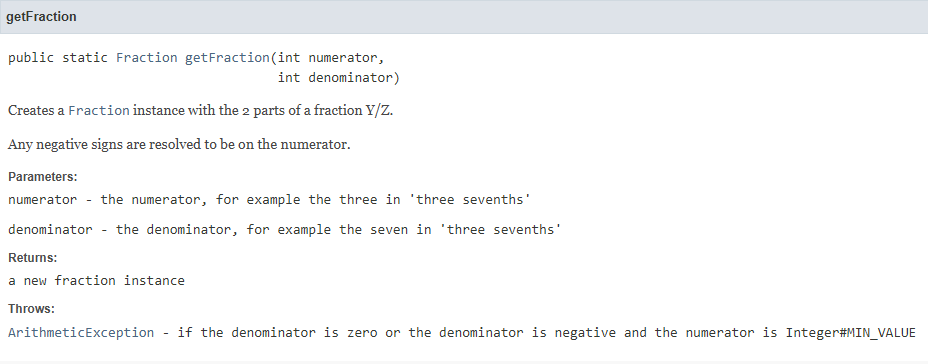
1. **Capitolo 1 (HW1)**

Nell'ambito dell'Homework 1, abbiamo effettuato specification – based testing sul costruttore della classe Fraction, getReducedFraction e greatestCommonDivisor. Questi metodi riguardano operazioni come calcoli numerici di base e altre funzionalità.

Il nostro obiettivo sarà comprendere e testare attentamente il comportamento di questi metodi rispetto alle specifiche fornite al fine di garantire che operino correttamente in una varietà di scenari.

**Step 1: Understanding the requirements**

* **Fraction constructor:**

****

Il costruttore della classe Fraction accetta un numeratore e un denominatore come input e inizializza un **oggetto** **Fraction** con tali valori. Prima di procedere all'inizializzazione, il costruttore effettua controlli per garantire che il denominatore non sia zero e gestisce casi speciali come denominatori negativi.

* Se il denominatore è zero, viene lanciata un'eccezione;
* Se il numeratore è zero, la frazione viene normalizzata a 0/1;
* Se il denominatore è negativo, nega anche il numeratore per mantenere la coerenza.

Infine, vengono impostati il numeratore e il denominatore con i valori forniti. In questo modo, il costruttore garantisce che l'oggetto **Fraction** sia correttamente inizializzato e rappresenti una frazione valida.

**Nota**: precisiamo che in questa fase ci siamo accorti che il costruttore della classe Fraction appartenente al codice sorgente di partenza, ovvero il codice preso direttamente dalla libreria Apache Commons Lang ([Fraction (Apache Commons Lang 3.14.0 API)](https://commons.apache.org/proper/commons-lang/apidocs/org/apache/commons/lang3/math/Fraction.html)), non effettuava assolutamente nessuno dei controlli soprammenzionati. In questo modo era possibile generare una frazione che avesse valori invalidi come numeratore zero, denominatore zero, entrambi pari a zero e così via.

Pertanto, abbiamo corretto il codice sorgente di partenza inserendo gli opportuni controlli degli input nel costruttore.

* **getReducedFraction:**

Immagine che contiene testo, schermata, Carattere, numero

Descrizione generata automaticamente

Il metodo accetta un numeratore e un denominatore come input e restituisce una **frazione ridotta ai minimi termini**.

Inizialmente, il metodo esegue controlli per assicurarsi che il denominatore non sia zero e gestisce casi speciali come denominatori negativi e numeratori nulli.

Successivamente, semplifica la frazione calcolando il Massimo Comune Divisore tra il numeratore e il denominatore e dividendo entrambi per questo valore.

Infine, restituisce un oggetto **Fraction** che rappresenta la frazione ridotta ai minimi termini.

* **greatestCommonDivisor:**

Calcola il Massimo Comune Divisore del valore assoluto di due numeri, utilizzando il *binary gcd* che evita operazioni di divisione e modulo. Come input il metodo prende due valori interi u e v, che sono due numeri diversi da zero, e come output restituisce il **Massimo Comune Divisore** che non è mai zero.

**Step 2: Explore what the program does for various inputs**

* **Fraction constructor:**

Abbiamo eseguito una suite di test utilizzando una varietà di input al fine di coprire un'ampia gamma di scenari possibili. Verifichiamo l'affidabilità del metodo utilizzando valori che creano frazioni valide e identificando le situazioni in cui tali frazioni non possano essere considerate valide.

Immagine che contiene testo, schermata, software

Descrizione generata automaticamente

* **getReducedFraction:**

Abbiamo eseguito una suite di test utilizzando una varietà di input al fine di coprire un'ampia gamma di scenari possibili. Verifichiamo che il metodo sia in grado di restituire una frazione ridotta ai minimi termini valida utilizzando valori interi positivi e negativi generati manualmente. In particolare, verifichiamo che il metodo restituisca una frazione ridotta ai minimi termini corretta anche con input negativi.Inizio moduloImmagine che contiene testo, schermata, software

Descrizione generata automaticamenteImmagine che contiene testo, schermata, software

Descrizione generata automaticamente

Immagine che contiene testo, schermata

Descrizione generata automaticamente

* **greatestCommonDivisor:**

Abbiamo eseguito una suite di test utilizzando una varietà di input al fine di coprire un'ampia gamma di scenari possibili. Verifichiamo la correttezza del calcolo del Massimo Comune Divisore (GCD) utilizzando valori sia positivi che negativi. Inoltre, identifichiamo i casi in cui il GCD risulta essere invalido.

Immagine che contiene testo, schermata, schermo

Descrizione generata automaticamente

**Step 3: Explore inputs, outputs and identify partitions**

* **Fraction constructor:**

**Individual inputs**

Il costruttore della classe Fraction accetta due numeri interi in input così classificabili: tutti gli input sono di tipo **intero** in base alle specifiche di Apache e vanno in un range da -2^31 (Integer.MIN\_VALUE) a 2^31 (Integer.MAX\_VALUE).

* Numerator: diverso da zero, zero, Integer.MAX\_VALUE, Integer.MIN\_VALUE;
* Denominator: diverso da zero, zero, Integer.MAX\_VALUE, Integer.MIN\_VALUE.

**Combinations of inputs**

Miriamo a testare una vasta gamma di scenari per garantire una copertura completa dei possibili input. Questi scenari includono:

* Numeratore e denominatore diversi da zero;
* Numeratore diverso da zero e denominatore = 0;
* Numeratore diverso da zero e denominatore = Integer.MAX\_VALUE;
* Numeratore diverso da zero e denominatore = Integer.MIN\_VALUE;
* Numeratore = 0 e denominatore diverso da zero;
* Numeratore = 0 e denominatore = 0;
* Numeratore = 0 e denominatore = Integer.MAX\_VALUE;
* Numeratore = 0 e denominatore = Integer.MIN\_VALUE;
* Numeratore = Integer.MAX\_VALUE e denominatore diverso da zero;
* Numeratore = Integer.MAX\_VALUE e denominatore = 0;
* Numeratore = Integer.MAX\_VALUE e denominatore = Integer.MAX\_VALUE;
* Numeratore = Integer.MAX\_VALUE e denominatore = Integer.MIN\_VALUE;
* Numeratore = Integer.MIN\_VALUE e denominatore diverso da zero;
* Numeratore = Integer.MIN\_VALUE e denominatore = 0;
* Numeratore = Integer.MIN\_VALUE e denominatore = Integer.MAX\_VALUE;
* Numeratore = Integer.MIN\_VALUE e denominatore = Integer.MIN\_VALUE.

In questo modo, possiamo garantire che il metodo sia in grado di gestire diversi casi di input in modo accurato.

**Classes of (expected) outputs**

Tutti gli output validi sono frazioni **intere** in base alle specifiche di Apache e vanno in un range da -2^31 a 2^31.

* **getReducedFraction:**

**Individual inputs**

Anche in questo caso il metodo accetta due interi, quindi gli individual inputs sono:

tutti gli input sono di tipo **intero** in base alle specifiche di Apache e vanno in un range da -2^31 (Integer.MIN\_VALUE) a 2^31 (Integer.MAX\_VALUE).

* Numerator: diverso da zero, zero, Integer.MAX\_VALUE, Integer.MIN\_VALUE;
* Denominator: diverso da zero, zero, Integer.MAX\_VALUE, Integer.MIN\_VALUE.

**Combinations of inputs**

Miriamo a testare una vasta gamma di inputs al fine di ottenere una copertura completa dei possibili scenari di input. In particolare, testiamo i casi:

* Numeratore e denominatore diversi da zero;
* Numeratore diverso da zero e denominatore = 0;
* Numeratore diverso da zero e denominatore = Integer.MAX\_VALUE;
* Numeratore diverso da zero e denominatore = Integer.MIN\_VALUE;
* Numeratore = 0 e denominatore diverso da zero;
* Numeratore = 0 e denominatore = 0;
* Numeratore = 0 e denominatore = Integer.MAX\_VALUE;
* Numeratore = 0 e denominatore = Integer.MIN\_VALUE;
* Numeratore = Integer.MAX\_VALUE e denominatore diverso da zero;
* Numeratore = Integer.MAX\_VALUE e denominatore zero;
* Numeratore = Integer.MAX\_VALUE e denominatore = Integer.MAX\_VALUE;
* Numeratore = Integer.MAX\_VALUE e denominatore = Integer.MIN\_VALUE;
* Numeratore = Integer.MIN\_VALUE e denominatore diverso da zero;
* Numeratore = Integer.MIN\_VALUE e denominatore = 0;
* Numeratore = Integer.MIN\_VALUE e denominatore = Integer.MAX\_VALUE;
* Numeratore = Integer.MIN\_VALUE e denominatore = Integer.MIN\_VALUE.

In questo modo, possiamo garantire che il metodo funzioni in modo robusto e in grado di gestire diversi casi di input in modo accurato.

**Classes of (expected) outputs**

Il nostro obiettivo è garantire che il metodo restituisca una frazione ridotta ai minimi termini, rispettando regole matematiche precise per evitare errori.

Tutti gli output validi sono frazioni **intere** in base alle specifiche di Apache e vanno in un range da -2^31 a 2^31.

* **greatestCommonDivisor:**

**Individual inputs**

In questo caso, questo metodo accetta due interi, quindi andiamo ad individuare gli individual inputs:

|  |  |
| --- | --- |
| PARAMETRO u |  |
| u = 0 |  |
| u = 1  u positivo |  |
| u primo |  |
| u negativo |  |

|  |  |
| --- | --- |
| PARAMETRO v |  |
| v = 0 |  |
| v = 1  v positivo |  |
| v primo |  |
| v negativo |  |

**Combinations of inputs**

Miriamo a testare una vasta gamma di inputs al fine di ottenere una copertura completa dei possibili scenari di input:

* u 0 e v 0;
* u 0 e v positivo;
* u 0 e v negativo;
* u positivo e v 0;
* u negativo e v 0;
* u 0 e v MIN\_VALUE;
* u MIN\_VALUE e v 0;
* u MIN\_VALUE e v MIN\_VALUE;
* u 1 e v positivo;
* u -1 e v positivo;
* u -1 e v negativo;
* u positivo e v 1;
* u negativo e v 1;
* u negativo e v -1;
* u 1 e v 1;
* u -1 e v 1;
* u -1 e v -1;
* u 1 e v MIN\_VALUE;
* u MIN\_VALUE e v 1;
* u -1 e v MIN\_VALUE;
* u MIN\_VALUE e v -1;
* u 1 e v MAX\_VALUE;
* u MAX\_VALUE e v 1;
* u -1 e v MAX\_VALUE;
* u MAX\_VALUE e v -1;
* u MIN\_VALUE e v MAX\_VALUE;
* u MAX\_VALUE e v MIN\_VALUE;
* u MAX\_VALUE e v MAX\_VALUE;
* u positivo e v MIN\_VALUE;
* u MIN\_VALUE e v positivo;
* u negativo e v MIN\_VALUE;
* u MIN\_VALUE e v negativo;
* u positivo e v MAX\_VALUE;
* u MAX\_VALUE e v positivo;
* u negativo e v MAX\_VALUE;
* u MAX\_VALUE e v negativo;
* u positivo e v MIN\_VALUE;
* u MIN\_VALUE e v positivo;
* u negativo e v MIN\_VALUE;
* u MIN\_VALUE e v negativo;
* u positivo e v MAX\_VALUE;
* u MAX\_VALUE e v positivo;
* u negativo e v MAX\_VALUE;
* u MAX\_VALUE e v negativo;
* u MIN\_VALUE + 1 v MIN\_VALUE + 1;
* u MIN\_VALUE + 1 v -1;
* u MIN\_VALUE + 1 v 1;
* u MIN\_VALUE + 1 v MAX\_VALUE - 1;
* u MAX\_VALUE - 1 v MIN\_VALUE + 1;
* u MAX\_VALUE - 1 v -1;
* u MAX\_VALUE - 1 v 1;
* u MAX\_VALUE - 1 v MAX\_VALUE - 1;

**Classes of (expected) outputs**

Tutti gli output validi sono numeri **interi positivi.**

**Step 4: Identify boundary cases (aka corner case)**

* **Fraction constructor:**

Basandoci sulla definizione di frazione, che coinvolge due numeri interi assegnati rispettivamente a denominatore e numeratore, abbiamo individuato situazioni significative che richiedono una gestione appropriata:

1. **Il denominatore è prossimo allo zero**: se il denominatore fosse zero, porterebbe ad un valore indefinito. In base ai requisiti di Apache *lancia un’eccezione di tipo ArithmeticException*. Se il denominatore fosse un valore prossimo allo zero (-1 e 1), il metodo funziona correttamente;
2. **Il numeratore è zero e il denominatore è un valore limite o prossimo a quest’ ultimo**: se il numeratore è uguale a zero, il metodo restituisce il valore costante Fraction.ZERO;
3. **Valori limite numeratore/denominatore**: se il numeratore o il denominatore o entrambi fossero uguali a Intger.MAX\_VALUE oppure Integer.MIN\_VALUE non dovrebbe sollevare eccezioni;
4. **Overflow/Underflow on point ed off point**: se il numeratore o il denominatore o entrambi fossero prossimi a Integer.MAX\_VALUE oppure Integer.MIN\_VALUE non dovrebbe sollevare eccezioni (Integer.MAX\_VALUE – 1 e Integer.MIN\_VALUE + 1);

* **GetReducedFraction:**

Basandoci sulla definizione di frazione, che coinvolge due numeri interi assegnati rispettivamente a denominatore e numeratore, abbiamo individuato situazioni significative che riguardano la riduzione delle frazioni:

* + - 1. **Il denominatore è prossimo zero**: se il denominatore fosse zero, porterebbe ad un valore indefinito. In base ai requisiti di Apache *lancia un’eccezione di tipo ArithmeticException.* Se il denominatore fosse un valore prossimo allo zero (-1 e 1), il metodo funziona correttamente;
      2. **Il numeratore è zero e il denominatore è un valore limite o prossimo a quest’ ultimo**: se il numeratore è uguale a zero, il metodo restituisce il valore costante Fraction.ZERO;
      3. **Valori limite numeratore/denominatore**: se il numeratore o il denominatore o entrambi fossero uguali a Intger.MAX\_VALUE oppure Integer.MIN\_VALUE non dovrebbe sollevare eccezioni;
      4. **Overflow/Underflow on point ed off point**: se il numeratore o il denominatore o entrambi fossero prossimi a Integer.MAX\_VALUE oppure Integer.MIN\_VALUE non dovrebbe sollevare eccezionin (Integer.MAX\_VALUE – 1 e Integer.MIN\_VALUE + 1);
      5. **Numeratore è un numero pari molto grande o molto piccolo, Il denominatore è MIN\_VALUE**: Se il numeratore e il denominatore sono numeri pari il metodo divide entrambi i termini per 2;
* **GreatestCommonDivisor:**

Basandoci sulla definizione di Massimo comune divisore (GCD), che coinvolge due numeri interi assegnati a due parametri **u** e **v**, ci siamo soffermati sulle situazioni fondamentali gestite da questo metodo:

1. Calcolo del GCD tra 1, -1 e MIN\_VALUE;
2. Calcolo del GCD tra 1, -1 e MAX\_VALUE;
3. Calcolo del GCD tra 1, -1 e MIN\_VALUE + 1;
4. Calcolo del GCD tra 1, -1 e MAX\_VALUE - 1;
5. Calcolo del GCD tra 2, -2 e MIN\_VALUE;
6. Calcolo del GCD tra MIN\_VALUE e MAX\_VALUE e viceversa;
7. Calcolo del GCD tra MIN\_VALUE + 1 e MAX\_VALUE - 1 e viceversa;
8. Calcolo del GCD tra numero pari (positivo e negativo) e MIN\_VALUE;
9. Calcolo del GCD tra numero pari (positivo e negativo) e MAX\_VALUE;
10. Calcolo del GCD tra numero dispari (positivo e negativo) e MIN\_VALUE;
11. Calcolo del GCD tra numero dispari (positivo e negativo) e MAX\_VALUE;

**Step 5: Devise test cases**

In questo step andiamo a verificare per ogni metodo, quali parametri devono essere combinati tra loro e quali no:

* **Fraction:**

**T1:** Numeratore positivo e denominatore positivo;

Numeratore positivo e denominatore negativo;

Numeratore negativo e denominatore positivo;

Numeratore negativo e denominatore negativo;

**T2:** Numeratore 1 e denominatore 0;

Numeratore MAX\_VALUE e denominatore 0;

Numeratore MIN\_VALUE e denominatore 0;

**T3:** Numeratore 0 e denominatore 1;

Numeratore 0 e denominatore negativo;

Numeratore 0 e denominatore MAX\_VALUE;

Numeratore 0 e denominatore MIN\_VALUE;

Numeratore 0 e denominatore MAX\_VALUE -1;

Numeratore 0 e denominatore MIN\_VALUE +1;

**T4:** Numeratore MIN\_VALUE e denominatore -1;

Numeratore 1 e denominatore MIN\_VALUE;

Numeratore MIN\_VALUE e denominatore MIN\_VALUE;

Numeratore MAX\_VALUE e denominatore MIN\_VALUE;

**T5:** Numeratore MIN\_VALUE e denominatore 1;

Numeratore MIN\_VALUE e denominatore MAX\_VALUE;

Numeratore MAX\_VALUE e denominatore -1;

Numeratore MAX\_VALUE e denominatore 1;

Numeratore MAX\_VALUE e denominatore MAX\_VALUE;

Numeratore MIN\_VALUE + 1 e denominatore MIN\_VALUE + 1;

Numeratore MIN\_VALUE + 1 e denominatore -1;

Numeratore MIN\_VALUE + 1 e denominatore 1;

Numeratore MIN\_VALUE + 1 e denominatore MAX\_VALUE - 1;

Numeratore MAX\_VALUE - 1e denominatore MIN\_VALUE + 1;

Numeratore MAX\_VALUE - 1 e denominatore -1;

Numeratore MAX\_VALUE - 1 e denominatore 1;

Numeratore MAX\_VALUE - 1 e denominatore MAX\_VALUE - 1;

* **GetReducedFraction:**

**T6:** Numeratore positivo e denominatore positivo;

Numeratore positivo e denominatore negativo;

Numeratore negativo e denominatore positivo;

Numeratore negativo e denominatore negativo;

Numeratore positivo e denominatore positivo;

Numeratore negativo e denominatore positivo;

Numeratore positivo e denominatore negativo;

Numeratore negativo e denominatore negativo;

Numeratore positivo primo e denominatore positivo primo;

Numeratore positivo primo e denominatore negativo primo;

Numeratore negativo primo e denominatore positivo primo;

Numeratore negativo primo e denominatore positivo primo;

Numeratore positivo dispari e denominatore positivo pari;

Numeratore positivo dispari e denominatore negativo pari;

Numeratore negativo dispari e denominatore positivo pari;

Numeratore negativo dispari e denominatore positivo pari;

Numeratore positivo pari e denominatore positivo dispari;

Numeratore positivo pari e denominatore negativo dispari;

Numeratore negativo pari e denominatore positivo dispari;

Numeratore negativo pari e denominatore positivo dispari;

**T7:** Numeratore 1 e denominatore 0;

Numeratore MAX\_VALUE e denominatore 0;

Numeratore MIN\_VALUE e denominatore 0;

**T8:** Numeratore 0 e denominatore 1;

Numeratore 0 e denominatore MAX\_VALUE;

Numeratore 0 e denominatore MIN\_VALUE;

Numeratore 0 e denominatore MAX\_VALUE - 1;

Numeratore 0 e denominatore MIN\_VALUE + 1;

**T9:** Numeratore 2 e denominatore MIN VALUE;

Numeratore -2 e denominatore MIN VALUE;

Numeratore MAX\_VALUE - 1 e denominatore MIN VALUE;

Numeratore MIN\_VALUE + 2 e denominatore MIN VALUE;

**T10:** Numeratore MIN\_VALUE e denominatore -1;

Numeratore 1 e denominatore MIN\_VALUE;

Numeratore MAX\_VALUE e denominatore MIN\_VALUE;

**T11:** Numeratore MIN\_VALUE e denominatore 1;

Numeratore MIN\_VALUE e denominatore MAX\_VALUE;

Numeratore MAX\_VALUE e denominatore -1;

Numeratore MAX\_VALUE e denominatore 1;

Numeratore MAX\_VALUE e denominatore MAX\_VALUE;

Numeratore MIN\_VALUE + 1 e denominatore MIN\_VALUE + 1;

Numeratore MIN\_VALUE + 1 e denominatore -1;

Numeratore MIN\_VALUE + 1 e denominatore 1;

Numeratore MIN\_VALUE + 1 e denominatore MAX\_VALUE - 1;

Numeratore MAX\_VALUE - 1 e denominatore MIN\_VALUE + 1;

Numeratore MAX\_VALUE - 1 e denominatore -1;

Numeratore MAX\_VALUE - 1 e denominatore 1;

Numeratore MAX\_VALUE - 1 e denominatore MAX\_VALUE - 1;

* **GreatestCommonDivisor:**

**T12: Calcolo classico del GCD**

u positivo e v positivo;

u negativo e v positivo;

u positivo e v negativo;

u negativo e v negativo;

**Calcolo del GCD tra numeri primi non uguali tra loro**

u positivo e v positivo;

u negativo e v positivo;

u positivo e v negativo;

u negativo e v negativo;

**Calcolo del GCD tra numero pari e numero dispari**

u positivo e v positivo;

u negativo e v positivo;

u positivo e v negativo;

u negativo e v negativo;

**Calcolo del GCD tra numero dispari e numero pari**

u positivo e v positivo;

u negativo e v positivo;

u positivo e v negativo;

u negativo e v negativo;

**T13:** u 0 e v 0;

**T14:** u 0 e v positivo;

u 0 e v negativo;

u positivo e v 0;

u negativo e v 0;

**T15:** u 0 e v MIN\_VALUE;

u MIN\_VALUE e v 0;

u MIN\_VALUE e v MIN\_VALUE;

**T16: Calcolo del GCD tra 1 e numero intero**

u 1 e v positivo;

u -1 e v positivo;

u -1 e v negativo;

**Calcolo del GCD tra numero intero e 1**

u positivo e v 1;

u negativo e v 1;

u negativo e v -1;

**Calcolo del GCD con entrambi i parametri uguali a 1**

u 1 e v 1;

u -1 e v 1;

u -1 e v -1;

**T17: Calcolo del GCD tra 1 e MIN\_VALUE**

u 1 e v MIN\_VALUE;

u MIN\_VALUE e v 1;

u -1 e v MIN\_VALUE;

u MIN\_VALUE e v -1;

**Calcolo del GCD tra 1 e MAX\_VALUE**

u 1 e v MAX\_VALUE;

u MAX\_VALUE e v 1;

u -1 e v MAX\_VALUE;

u MAX\_VALUE e v -1;

**Calcolo del GCD tra MIN\_VALUE e MAX\_VALUE**

u MIN\_VALUE e v MAX\_VALUE;

u MAX\_VALUE e v MIN\_VALUE;

u MAX\_VALUE e v MAX\_VALUE;

**Calcolo del GCD tra numero pari e MIN\_VALUE**

u positivo e v MIN\_VALUE;

u MIN\_VALUE e v positivo;

u negativo e v MIN\_VALUE;

u MIN\_VALUE e v negativo;

**Calcolo del GCD tra numero pari e MAX\_VALUE**

u positivo e v MAX\_VALUE;

u MAX\_VALUE e v positivo;

u negativo e v MAX\_VALUE;

u MAX\_VALUE e v negativo;

**Calcolo del GCD tra numero dispari e MIN\_VALUE**

u positivo e v MIN\_VALUE;

u MIN\_VALUE e v positivo;

u negativo e v MIN\_VALUE;

u MIN\_VALUE e v negativo;

**Calcolo del GCD tra numero dispari e MAX\_VALUE**

u positivo e v MAX\_VALUE;

u MAX\_VALUE e v positivo;

u negativo e v MAX\_VALUE;

u MAX\_VALUE e v negativo;

**Calcolo del GCD con MIN\_VALUE + 1**

u MIN\_VALUE + 1 v MIN\_VALUE + 1;

u MIN\_VALUE + 1 v -1;

u MIN\_VALUE + 1 v 1;

u MIN\_VALUE + 1 v MAX\_VALUE - 1;

**Calcolo del GCD con MAX\_VALUE - 1**

u MAX\_VALUE - 1 v MIN\_VALUE + 1;

u MAX\_VALUE - 1 v -1;

u MAX\_VALUE - 1 v 1;

u MAX\_VALUE - 1 v MAX\_VALUE - 1;

**Step 6: Automate test cases**

* **Fraction:** Automatizziamo i test nel codice esplicitati nello step 5.

Immagine che contiene testo, schermata, software

Descrizione generata automaticamente

**Immagine che contiene testo, schermata

Descrizione generata automaticamente**

**Immagine che contiene testo, schermata

Descrizione generata automaticamente**

* **GetReducedFraction:** Automatizziamo i test nel codice esplicitati nello step 5.

Immagine che contiene testo, schermata

Descrizione generata automaticamente

Immagine che contiene testo, schermata

Descrizione generata automaticamente

Immagine che contiene testo, schermata, Carattere

Descrizione generata automaticamente

Immagine che contiene testo, schermata, Carattere

Descrizione generata automaticamente

* **GreatestCommonDivisor:** Automatizziamo i test nel codice esplicitati nello step 5.

Immagine che contiene testo, schermata, schermo

Descrizione generata automaticamente

Immagine che contiene testo, schermata, Carattere

Descrizione generata automaticamente

**Immagine che contiene testo, schermata

Descrizione generata automaticamente**

**Immagine che contiene testo, schermata, design

Descrizione generata automaticamente**

**Step 7: Augment the test suite with creativity and experience**

* **Fraction:** Abbiamo lavorato sul seguente metodo in maniera tale da testare più situazioni:
* Nel **T2**, abbiamo verificato che venga lanciata un'eccezione quando si assegna il valore 0 al denominatore;
* Nel **T3**, abbiamo assegnato il valore 0 al numeratore per confermare che venga restituita la costante Fraction.ZERO;
* Nel **T4**, in particolare nell'ultimo caso, abbiamo testato una combinazione aggiuntiva di valori per i due parametri selezionati: abbiamo combinato il valore minimo e il valore massimo per generare nuovamente un'eccezione;

**Immagine che contiene testo, schermata, software

Descrizione generata automaticamente**

**Immagine che contiene testo, schermata

Descrizione generata automaticamente**

* **GetReducedFraction:** Abbiamo lavorato su questo metodo per verificare nel **T10**, in particolare nella sua ultima condizione, un'altra combinazione per i nostri due parametri. Abbiamo assegnato loro i valori MIN VALUE e MAX VALUE, in modo da poter generare una nuova eccezione.  
  Abbiamo verificato l'efficacia del nostro metodo testandolo su valori pari, dispari e numeri primi, per assicurarci che le frazioni fossero ridotte correttamente.

**Immagine che contiene testo, schermata

Descrizione generata automaticamente**

**Immagine che contiene testo, schermata, Carattere

Descrizione generata automaticamente**

* **GreatestCommonDivisor:** Abbiamo lavorato sul seguente metodo in maniera tale da testare più situazioni:
* Nel **T13**, dopo aver modificato la condizione originale del metodo, abbiamo verificato la possibilità che entrambi i parametri u e v siano uguali a 0, generando di conseguenza un'eccezione;
* Nel **T15**, abbiamo sperimentato diverse combinazioni per i due parametri u e v, assegnando valori come 0 e MIN VALUE, per assicurare che venga nuovamente generata un'eccezione;

**Immagine che contiene testo, schermata, Carattere

Descrizione generata automaticamente**

**Structural testing**

Dopo aver completato il testing basato sulle specifiche, passiamo al testing strutturale. Abbiamo esaminato l'implementazione del codice ed eseguito la suite di test utilizzando uno strumento di code coverage, per identificare quali linee di codice non erano coperte dai test. Abbiamo utilizzato il **Condition + branch**, che considera non solo i possibili rami, ma anche ogni condizione di ciascuna istruzione di ramo. La suite di test dovrebbe verificare:

* che ciascuna di queste condizioni venga valutata almeno una volta come vera e come falsa
* che l'intera istruzione di ramo sia vera e falsa almeno una volta.

Di seguito riportiamo il risultato mostrato dal tool di coverage JaCoCo.

Immagine che contiene testo, schermata, Software multimediale, software

Descrizione generata automaticamente

* **Fraction:** Grazie al lavoro di testing svolto, tutte le linee di codice di questo metodo sono state coperte.
* **GetReducedFraction:** Grazie al lavoro di testing svolto, tutte le linee di codice di questo metodo sono state coperte.
* **GreatestCommonDivisor:** Per quanto riguarda questo metodo, non siamo riusciti a coprire completamente una singola riga di codice (riga 182). Anche se abbiamo fornito in input valori molto grandi, la copertura della riga risulta parziale.

1. **Capitolo 2 (HW2)**

**Introduzione**

L'obiettivo del nostro homework 2 è effettuare il property-based testing su un metodo della classe Fraction di Apache Commons Lang.

Il property-based testing è una tecnica che permette di generare automaticamente una vasta gamma di input casuali, ma in un intervallo specifico, per verificare che le proprietà specificate del software siano sempre soddisfatte.

Il metodo che abbiamo scelto per questo testing è getReducedFraction.

**Property-Based Testing**

Abbiamo quindi individuato per il nostro metodo, le proprietà principali che deve soddisfare:

* **PBT1**: Andiamo a testare l’invalidità del denominatore nel momento in cui questo risulta essere uguale a 0;
* **PBT2**: Andiamo a testare la validità della frazione quando numeratore e denominatore sono diversi da 0;
* **PBT3**: Andiamo a testare l’invalidità della frazione nel momento in cui al numeratore viene assegnato un valore dispari e il denominatore risulta essere uguale a MIN VALUE;
* **PBT4**: Andiamo a testare l’invalidità della frazione nel momento in cui il numeratore risulta essere uguale a MIN VALUE e al denominatore viene assegnato un valore negativo dispari;
* **PBT5**: Andiamo a testare la validità della frazione in più situazioni:

1. Quando numeratore è diverso da 0 e non è uguale a MIN VALUE;
2. Quando denominatore è diverso da 0 e non è uguale a MIN VALUE;

* **PBT1:**

Nel primo test, verifichiamo una condizione essenziale per le frazioni: la validità del denominatore. In particolare, controlliamo che il denominatore sia considerato invalido quando gli viene assegnato il valore 0.

**Immagine che contiene testo, schermata, software, Software multimediale

Descrizione generata automaticamente**

Per ogni test effettuato, abbiamo deciso di raccogliere delle statistiche per confermare e visualizzare su schermo i dati attesi. In questo caso, le statistiche mostrano che la condizione da noi verificata è sempre risultata vera (100%).

**Immagine che contiene testo, schermata, Carattere

Descrizione generata automaticamente**

* **PBT2:**

Nel secondo test, verifichiamo che, quando sia il denominatore sia il numeratore sono diversi da 0, la frazione risulti sempre valida, ovvero venga sempre ridotta ai minimi termini.

**Immagine che contiene testo, schermata, software

Descrizione generata automaticamente**

Anche in questo caso, vogliamo analizzare le statistiche. Possiamo osservare che per 968 dei casi esaminati nel test, la nostra condizione risulta essere sempre valida.

In aggiunta analizziamo la distribuzione dei valori del numeratore e del denominatore, i valori del MCD (Massimo Comun Divisore) e la probabilità che la frazione iniziale non venga ridotta (ossia quando il MCD è uguale a 1).

Abbiamo osservato che in 662 casi la frazione è stata ridotta ai minimi termini, mentre in 306 casi il metodo non è riuscito a ridurre la frazione poiché i termini avevano un MCD pari a 1.

**Immagine che contiene testo, schermata

Descrizione generata automaticamente**

**Immagine che contiene testo, schermata, Carattere

Descrizione generata automaticamente**

* **PBT3:**

Nel terzo test, verifichiamo l'invalidità della frazione quando il numeratore è un numero dispari e il denominatore è il valore minimo possibile (MIN VALUE).

In questo caso, ci aspettiamo che venga lanciata un'eccezione, poiché la frazione non sarà mai valida.

Immagine che contiene testo, schermata, software, Software multimediale

Descrizione generata automaticamente

Analizzando le statistiche di questo test, notiamo che sono state generate solo 504 combinazioni che soddisfano i criteri dichiarati nel test, ovvero i parametri per cui la nostra frazione risulta non valida al 100%.

Immagine che contiene testo, schermata, menu

Descrizione generata automaticamente

* **PBT4:**

Nel quarto test, verifichiamo l'invalidità della frazione quando il numeratore è il valore minimo possibile (MIN VALUE) e il denominatore è un numero dispari negativo.

In questo caso, ci aspettiamo che venga lanciata un'eccezione, poiché la frazione non sarà mai valida.

Immagine che contiene testo, schermata, software, Software multimediale

Descrizione generata automaticamente

Analogamente al test precedente, le statistiche mostrano che solo 498 combinazioni di parametri generate risultano sempre non valide al 100%. Questo accade perché il range impostato permette al test di generare anche valori pari. Tuttavia, quando il test verifica quali combinazioni rispettano la condizione, considera solo quelle in cui il parametro scelto è dispari.

Immagine che contiene testo, schermata, menu

Descrizione generata automaticamente

* **PBT5:**

Nel quinto e ultimo test, verifichiamo più situazioni per cui la nostra frazione risulti valida:

* La frazione è valida quando il numeratore ha valori diversi da MIN VALUE e 0;
* La frazione è valida quando il denominatore è negativo e ha valori diversi da 0 e MIN VALUE.

Immagine che contiene testo, schermata, software

Descrizione generata automaticamente

In questo caso, le statistiche mostrano che la condizione da noi verificata è sempre risultata vera (100%) nel 990 dei casi presi in considerazione dal test.

Anche in questo caso andiamo ad analizzare la distribuzione dei valori del numeratore e del denominatore, i valori del MCD (Massimo Comun Divisore) e la probabilità che la frazione iniziale non venga ridotta (ossia quando il MCD è uguale a 1).

Al contrario del PBT2, possiamo vedere come in tutti i casi verificati la frazione sia stata ridotta ai minimi termini.

Immagine che contiene testo, schermata

Descrizione generata automaticamente

Immagine che contiene testo, schermata, Carattere

Descrizione generata automaticamente