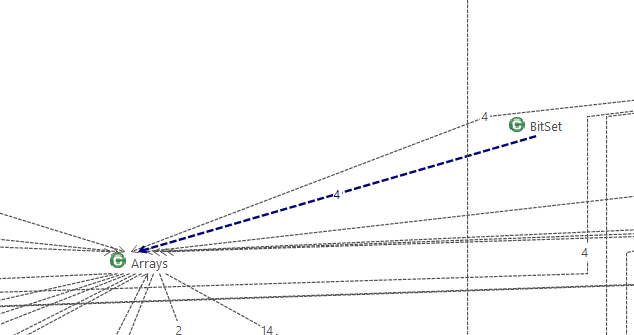
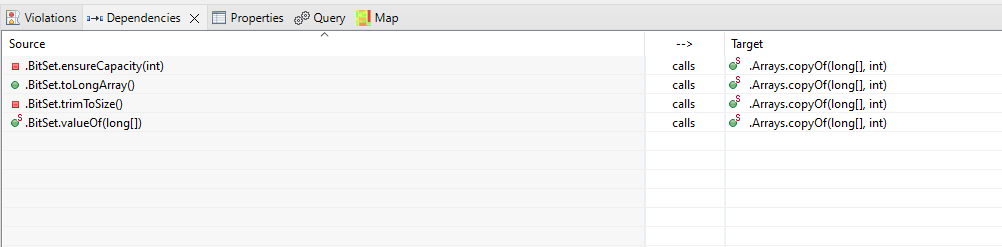
Laboratorio 2 FIS

1. ***Quante sono le dipendenze tra la classe “java.util.BitSet” e la classe “java.util.Arrays”? Di che tipo sono?***

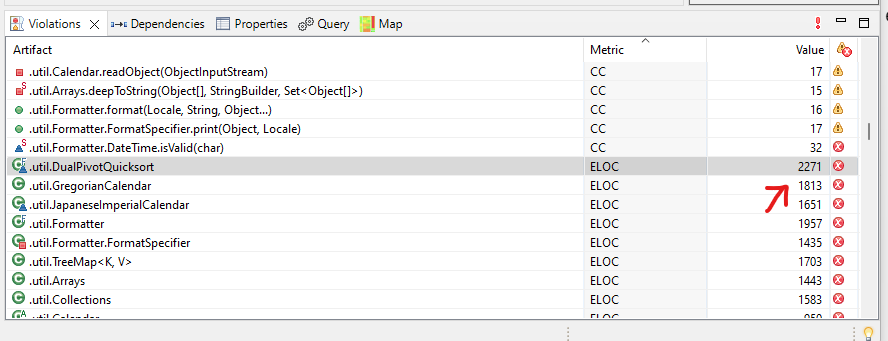




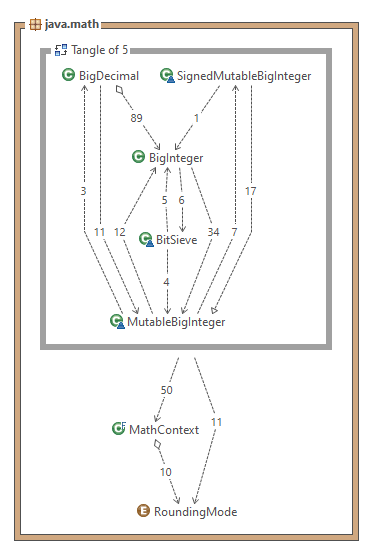
Come si puo’ vedere, le dipendenze tra queste due classi sono 4 e sono di tipo “chiamata” (da java.util.BitSet verso java.util.Arrays)

1. ***Quante classi con più di 2k ELOC ci sono nel package “java.util”? Riportare le loro dimensioni in ELOC.***

Ne e’ presente solo una, ovvero java.util.DualPivotQuicksort, che ha ELOC = 2271

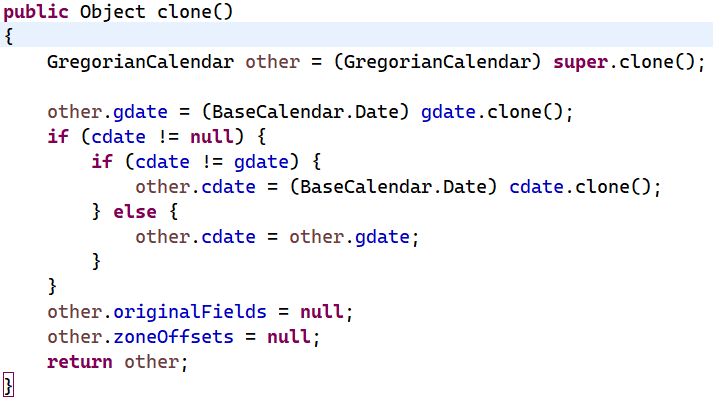


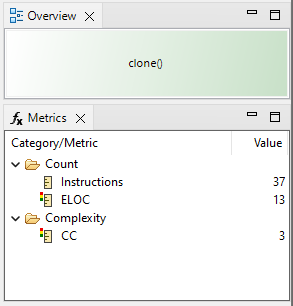
1. ***Nel package “java.math” è presente un Tangle; da quante componenti è formato?***



Come si vede nell’immagine, e’ presente un tangle composto da 5 classi

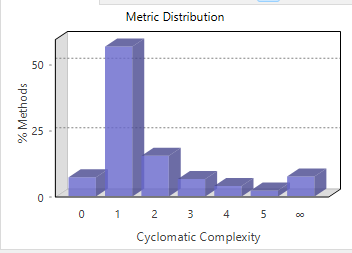
1. ***Qual è il valore assunto dalla Complessità Ciclomatica (CC) per il metodo “clone()” nella classe “java.util.GregorianCalendar”? Verificare tale valore riportando il calcolo della metrica.***





Siccome nel metodo clone() ci sono 2 if, la CC e’ 3. Infatti anche nelle metriche CC = 3

1. ***Qual è la percentuale di metodi del package “java.util” che hanno Complessità Ciclomatica (CC) maggiore (>) di 4? Qual è la CC più diffusa tra i metodi del package “java.util”?***



L’immagine sopra rappresenta la distribuzione della complessita’ ciclomatica per il package java.util, come si puo’ vedere, la percentuale dei metodi che hanno 4 < CC < ∞ e’ circa il 9.89%, e sempre dal grafico la CC piu’ diffusa e’ quella 0 < CC <= 1, con un valore del 56.8%

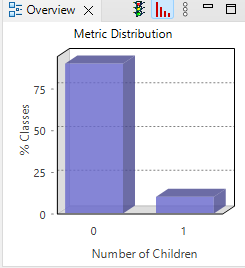
1. ***Tra i due package “java.util” e “java.math” quale tra i due mostra un miglior bilanciamento tra Instability (I) e Abstractness (A)?***

per il package java.util A = 0.3 e I = 0.5, quindi la loro differenza e’ di 0.2

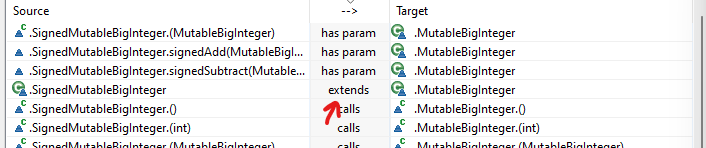
per java.math A = 0.12 e I = 0.5, con una differenza di 0.7.

Il package piu’ bilanciato e’ quindi java.util, siccome la differenza tra A e I e’ piu’ piccola rispetto a java.math

1. ***Qual è il valore della metrica NOC per la classe “java.math.MutableBigInteger”? Verificare tale valore riportando i nomi delle sottoclassi.***



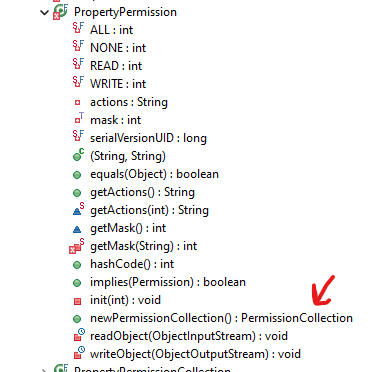
Per la classe MutableBigInteger, NOC = 0.1, questo perche’ MutableBigInteger ha un solo figlio, ovvero SignedMutableBigInteger



1. ***Qual è il valore della metrica RFC per la classe “java.util.PropertyPermission”? Verificare tale valore riportando il numero di metodi della classe + l’elenco dei nomi dei metodi di altre classi chiamati dalla classe stessa.***

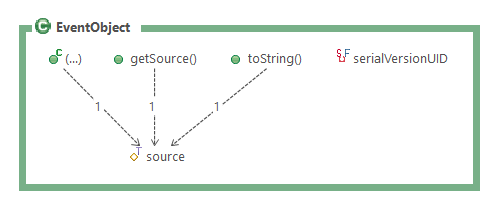
per java.util.PropertyPermission, il valore RFC = 15.2

il suo numero di metodi e’ 12, e come possiamo vedere dalla view composition, il suo unico arco uscente (dipendenza) e’ verso java.util.PropertyPermissionCollection. La dipendenza e’ una, verso pero’ due diversi target in PropertyPermissionCollection. Ovvero, di tipo “references” (alla classe stessa) e una di tipo “calls” verso il costruttore di PropertyPermissionCollection. Immagino quindi che PropertyPermission abbia un campo di tipo PropertyPermissionCollection, infatti e’ cosi’



1. ***Qual è il valore della metrica FAT per la classe “java.util.EventObject”? Verificare tale valore riportando il nome dei nodi, nel grafo delle dipendenze utilizzato per il calcolo di FAT, che hanno almeno un arco entrante.***

all’interno di java.util.EventObject, FAT = 3, questo perche’, in questa classe, le uniche dipendenze sono 3 archi entranti nel campo *source*



1. ***Qual è il metodo più difficile da testare nel package “java.math”? Quale metrica/metriche ritieni utile/i valutare per rispondere a tale domanda? (argomentare)***

Il metodo piu’ difficile da testare e’ un costruttore di java.math.BigDecimal, in particolare java.math.BigDecimal(char[], int, int), con un CC = 52. Ritengo questa metrica particolarmente utile per rispondere alla domanda perche’ aggiungendo delle condizioni in un metodo occorre testare sia quando queste condizioni sono false, sia quando sono vere, sia quando alcune sono false e altre vere. A livello di metodo non ci sono altre metriche utili per capire come questo incidera’ sul testing