National University of Computer and Emerging Sciences



Coverage Report

CS4036 Project

|  |  |
| --- | --- |
| **Members** | **Names** |
| 19F-0170 | Danish Ahmad |
| 19F-0228 | Muhammad Zain |
| 19F-0311 | Talha Ahmad |

# 

# BLL Functions

## *<toNarrators>*

*/\*\**

*\**

*\* @author zain*

*\**

*\*/*

public ArrayList<String> narrateHadithsToGivenNarrator(String inputName) {

ArrayList<String> Row = new ArrayList<String>();

ArrayList<String> narratorsList = new ArrayList<String>();

Row = dao.getHadiths();*// ----> get all rows here*

for (int i = 0; i < Row.size(); i++) {

String[] NarratorsInOneRow = SplitNarrators(Row, i);

for (int j = 0; j < NarratorsInOneRow.length - 1; j++) {

String getOne = NarratorsInOneRow[j];

if (getOne.equals(inputName) && !narratorsList.contains(NarratorsInOneRow[j + 1])) {

narratorsList.add(NarratorsInOneRow[j + 1]);

}

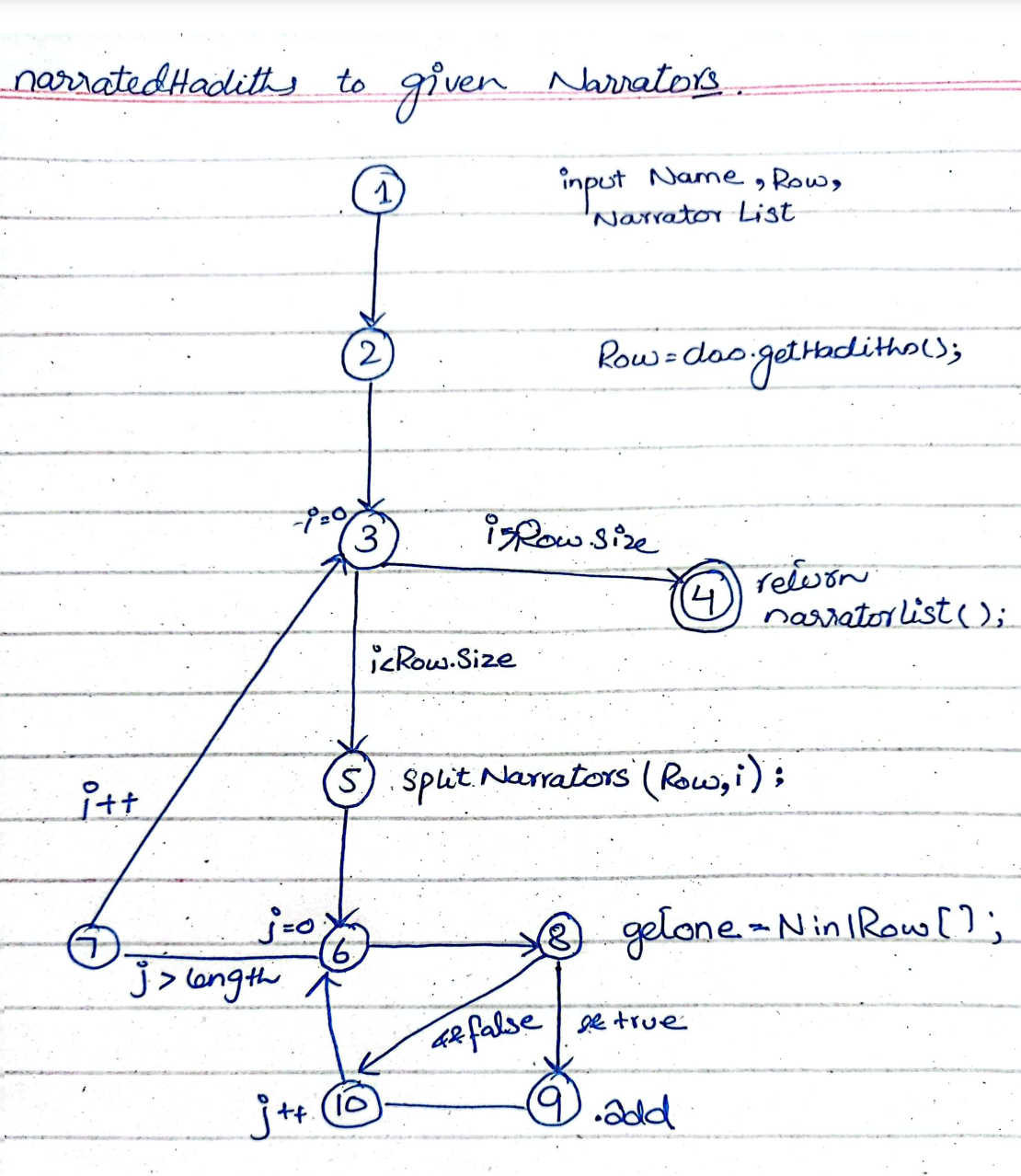
}

}

return narratorsList;

}

### CFG



### TR set for Edge-Pair Coverage

TR= { (1,2,3), (2,3,4), (2,3,5), (3,5,6), (5,6,8), (5,6,7), (6,8,9), (6,8,10), (8,9,10), (8,10,6), (9,10,6), (10,6,8), (10,6,7), (6,7,3), (7,3,4), (7,3,5) }

### Test Paths

| [1,2,3,4] | [1,2,3], [2,3,4] |
| --- | --- |
| [1,2,3,5,6,8,9,10,6 ,7,3,4] | [1,2,3], [2,3,5], [3,5,6], [5,6,8], [6,8,9], [8,9,10], [9,10,6], [10,6,7], [6,7,3], [7,3,4] |
| [1,2,3,5,6,8,10,6,8,10,6,7,3,4 | [1,2,3], [2,3,5], [3,5,6], [5,6,8], [6,8,10], [8,10,6], [10,6,8], [10,6,7], [6,7,3], [7,3,4] |

## *<fromNarrators>:*

*/\*\**

*\**

*\* @author danish*

*\**

*\*/*

public ArrayList<String> narrateHadithsFromGivenNarrator(String inputName) {

ArrayList<String> Row = new ArrayList<String>();

ArrayList<String> narratorsList = new ArrayList<String>();

Row = dao.getHadiths();*// ----> get all rows here*

for (int i = 0; i < Row.size(); i++) {

String[] NarratorsInOneRow = SplitNarrators(Row, i);

for (int j = 0; j < NarratorsInOneRow.length - 1; j++) {

String getOne = NarratorsInOneRow[j];

if (getOne.equals(inputName) && !narratorsList.contains(NarratorsInOneRow[j + 1])) {

narratorsList.add(NarratorsInOneRow[j - 1]);

}

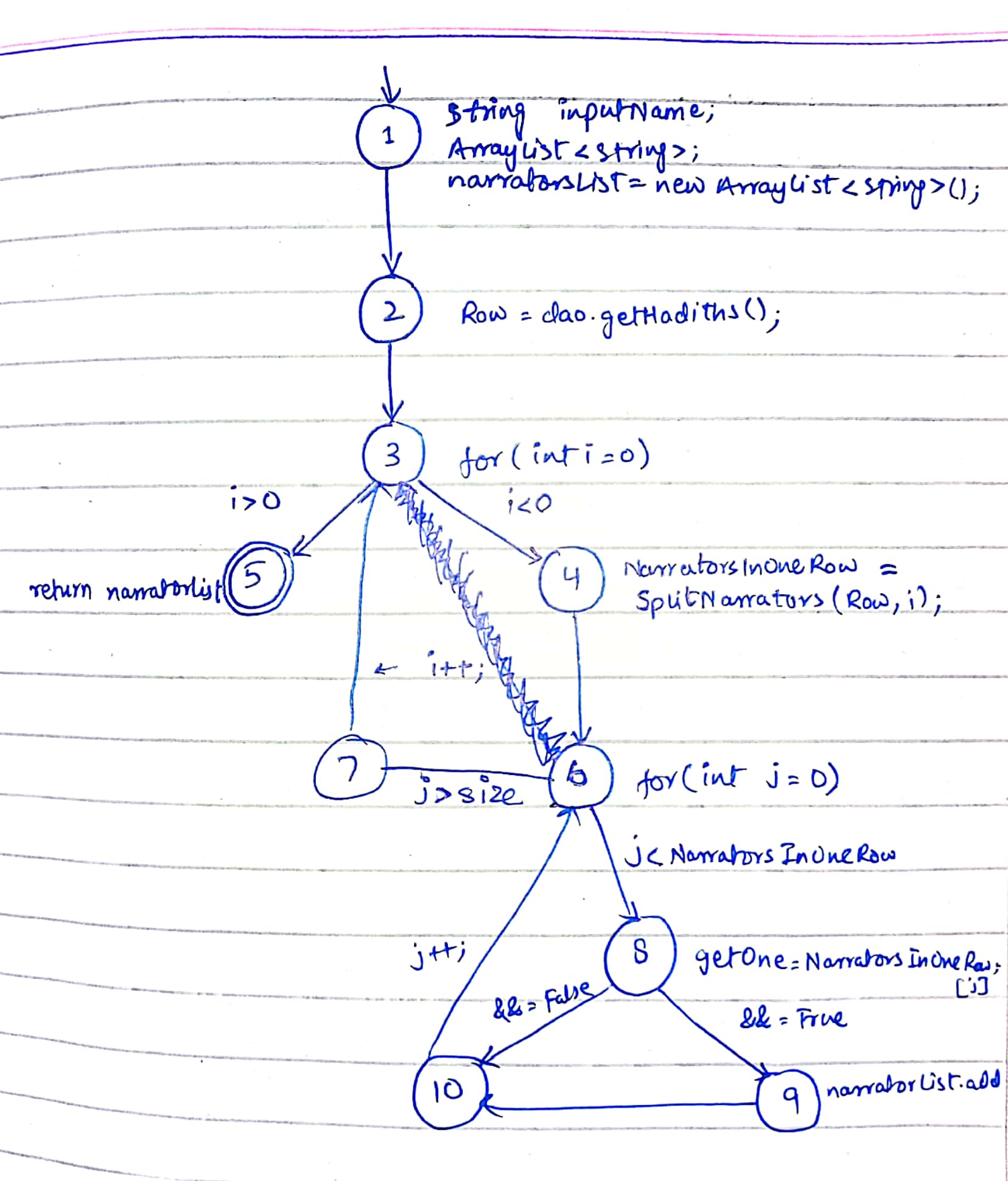
}

}

return narratorsList;

}

### CFG



### TR set for Edge-Pair Coverage

TR= { (1,2,3), (2,3,4), (2,3,5), (3,4,6), (4,6,7), (4,6,8), (6,7,3), (7,3,4), (7,3,5), (6,8,9), (6,8,10), (8,9,10), (8,10,6), (9,10,6), (10,6,7), (10,6,8) }

### Test Paths

| [1,2,3,5] | [1,2,3], [2,3,5] |
| --- | --- |
| [1,2,3,4,6,7,3,4,6,8,9,10,6,7,3,5] | [1,2,3], [2,3,4], [3,4,6], [4,6,7], [4,6,8], [6,7,3], [7,3,4], [7,3,5], [6,8,9], [8,9,10], [9,10,6], [10,6,7] |
| [1,2,3,4,6,8,10,6,8,10,6,7,3,5] | [1,2,3], [2,3,4], [3,4,6], [4,6,8], [6,7,3], [7,3,5], [6,8,10], [8,10,6], [10,6,7], [10,6,8] |

## *<uniqueNarrators>:*

*/\*\**

*\**

*\* @author Talha*

*\**

*\*/*

@Override

public List<String> getUniqueNarratorsAtLevelN(String book, int level) {

List<Hadees> dataList = dao.getDataByBook(book);

Map<String, Set<String>> narratorLevels = new HashMap<>();

for (Hadees data : dataList) {

String sanad = data.getSanad();

String[] narrators = sanad.split(",");

for (int i = 0; i < narrators.length; i++) {

String narrator = narrators[i].trim();

Set<String> levels = narratorLevels.get(narrator);

if (levels == null) {

levels = new LinkedHashSet<>();

narratorLevels.put(narrator, levels);

}

levels.add(data.getBook() + "-" + data.gethadithNum() + "-" + (i + 1));

}

}

List<String> uniqueNarrators = new ArrayList<>();

for (Map.Entry<String, Set<String>> entry : narratorLevels.entrySet()) {

String narrator = entry.getKey();

Set<String> levels = entry.getValue();

if (levels.size() == level) {

uniqueNarrators.add(narrator);

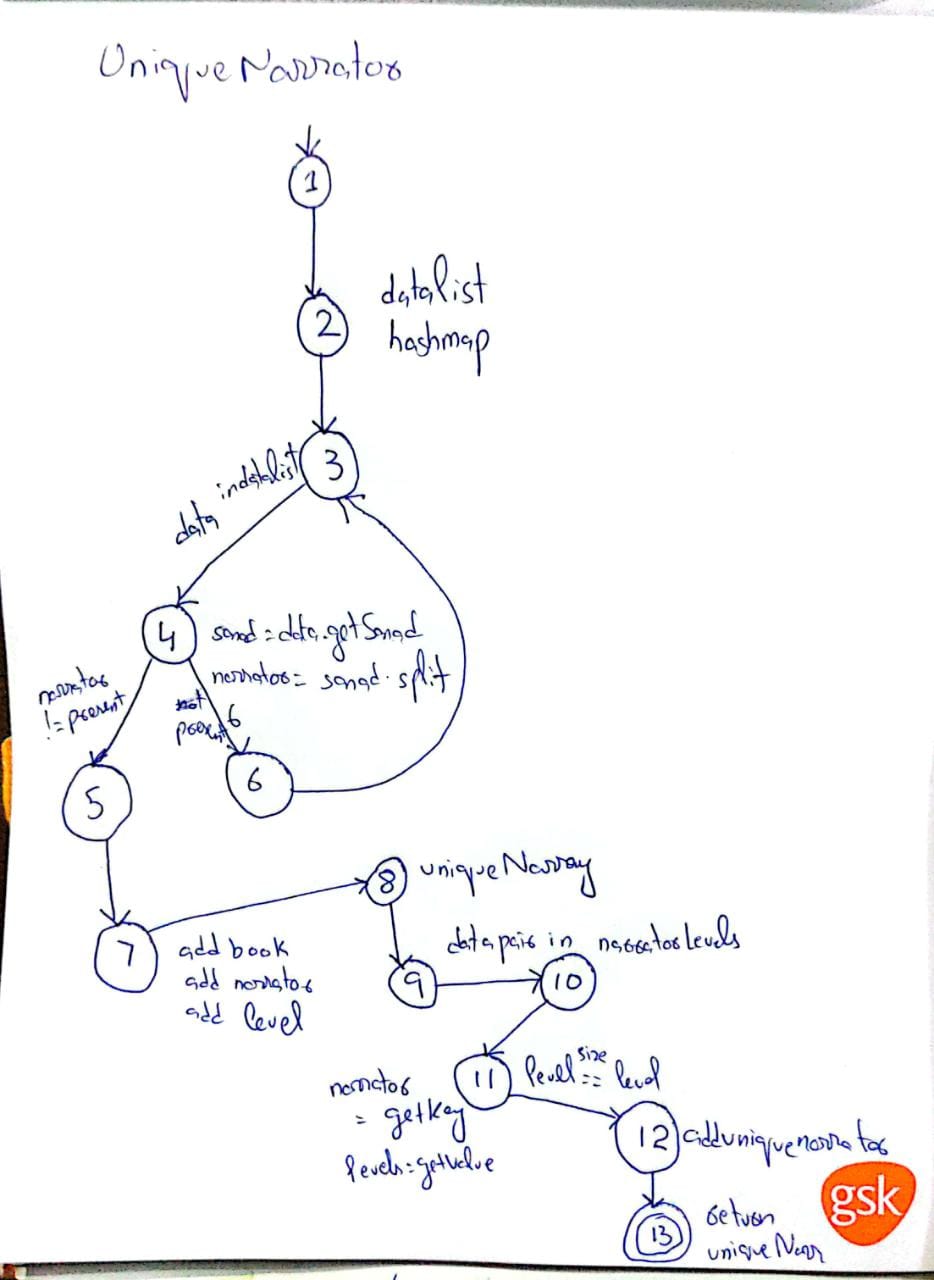
}

}

return uniqueNarrators;

}

### CFG



### TR set for Edge-Pair Coverage

TR = { {1,2,3}, {2,3,4}, {3,4,5}, {3,4,6}, {4,5,7}, {4,6,3}, {6,3,4}, {5,7,8}, {7,8,9}, {8,9,10}, {9,10,11}, {10,11,12}, {11,12,13} }

### Test Paths

| [1,2,3,4,6,3,4,6,3,4,5,7,8,9,10,11,12,13] | [1,2,3], [2,3,4], [3,4,5], [3,4,6], [4,5,7], [4,6,3], [6,3,4], [5,7,8], [7,8,9], [8,9,10], [9,10,11], [10,11,12], [11,12,13] |
| --- | --- |
| [1,2,3,4,5,7,8,9,10,11,12,13] | [1,2,3], [2,3,4], [3,4,5], [4,5,7], [5,7,8], [7,8,9], [8,9,10], [9,10,11], [10,11,12], [11,12,13] |