Github: https://github.com/Tasadan-Filip/FilipT-lftc-labs

1. Hash function

a. Hash function implementaiton:

{

int hashValue = 7;

for (int index = 0; index < stringToHash.length(); index++) {

hashValue = hashValue \* 70 + stringToHash.charAt(index);

}

return hashValue % tableSize;

}

b. Hash function explanation:

The hash function multiplies the current computed hash value with 70 (number of different characters that we could have in the string) and it adds the ASCII value of the character. In order to retrieve the list's position where the new element should be added, the hash function's value will be the remainder of the computed hash value and the size of the symbol table (because the symbol table is represented as a list of lists)

2. Collision resolution method:

The method used is "Separate chaining Hash Table" in which the hash table contains a list of lists, where each internal list contains string elements that should be kept in the table. Each element is hashed to the position of a internal list from the table's list and it gets added there if doesn't exist.

3. Class diagram

