https://github.com/alexandra-olahut/FLCD-assignments

FLCD Scanner Documentation

Olahut Alexandra – 935/2

Lab2

## Symbol Table

Symbol table is implemented as a hash table, in Python.

The elements are represented as a main list with given size, where the positions represent the hashed values of the keys. Collisions are solved using separate chaining, so on each position, elements are placed using the deque from Python.

The class attributes are:

- Table size (given as parameter)
- Number of elements (updated with each addition, the numbers of symbols)
- The elements

The implemented operations are:

1. add(key) – adds a new symbol in the table
2. getPosition(key) – for given symbol returns its position as a tuple

   (indexInList, positionInDeque)

3. exists(key) – checks if a symbol is in the table or not
4. size() – returns the number of symbols currently in the symbol table

When adding a new symbol, its hash value is computed and the element is added at the computed position in the list; if there are more elements on that position, the new element is added at the end of the list(deque). If the same symbol already exists in the symbol table, it is not added again.

The hash value is computed as follows:

- the symbol is converted to a string (if it is a number)
- compute the sum of ascii codes from the string
- return the sum modulo the size of the hash table