Corlaci Andreea- Lab 2

Documentation

Git link: https://github.com/CorlaciAndreea/Flcd-lab

Symbol table implementation: a hash table which has as attributes: table- the actual symbol table.

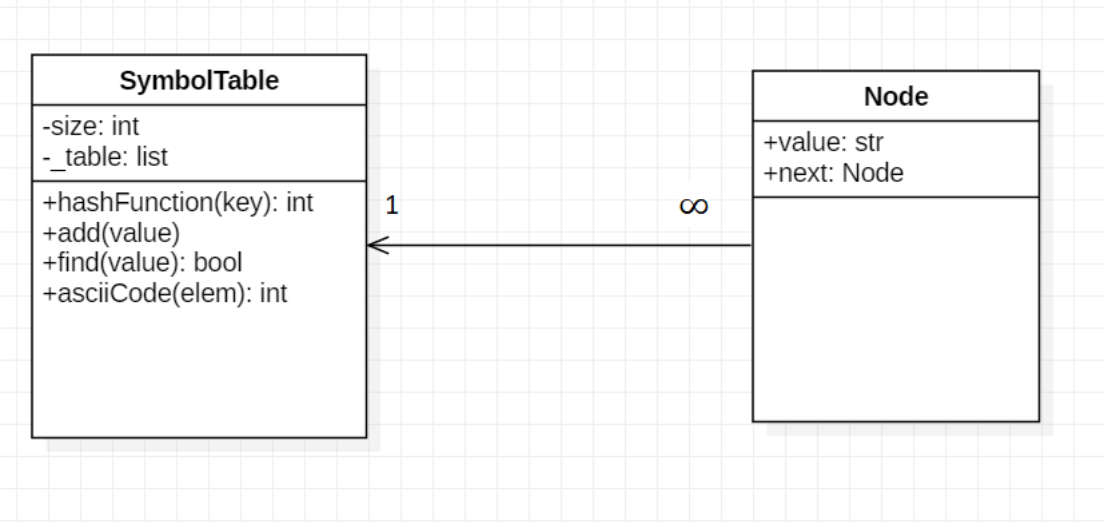
For collisions, I used a linked list.

I created an auxiliary class Node to keep the actual value of it and the value of the next node.

In the **HashTable class** I implemented the following functions:

- Init(size): Creates the table as a list of lists
  Input: key-integer
  Output: the hashtable
- hashFuntion (key): Maps the key to a value in the hashtable
  input: key-integer
  output: key% len(hashtable)
- asciiCode(elem): return the ascii code of a given elem
  pre: elem is a string
  input: elem-string
  output: asciiCode: int
- add(value): Inserts a new value in the table.If the value is string we compute the ascii code and then hash it to find the position, else we use the actual value. If the position is empty, creates a new node with the value, if not, it adds the new value to the linked list associated with that position
  input: value- string or integer
  output: -
- find(value): searches for an element with the given value
  input: value
  output: true – if the value is in the table
  false-otherwise
- __str__: print the hashtable
  Input: -
  Output: a string with the hashtable

Diagram:


SymbolTable
-size: int
-_table: list
+hashFunction(key): int
+add(value)
+find(value): bool
+asciiCode(elem): int
1
∞
Node
+value: str
+next: Node