https://github.com/Calandrinon/Formal-Languages-And-Compilers/tree/master/Labs/Lab-3

The SymbolTable class:
The symbol table is a hash table implemented with a Python dictionary.

- ascii_hash(input) -> int

  The ascii_hash method represents the hash function used for our hash table. It computes the sum of all the ASCII codes of its input.

- is_key_in_table(key) -> bool

  This method checks if a given key exists in the hash table and returns an appropriate boolean.

- add_element(element) -> (key, element position in the chain) or -1

  Adds an element to the hash table by computing the corresponding key (with the hash function) and checking 2 cases:

  1. if the corresponding key is not already in the table, a list with the element is created on the key of the hash table
  2. if the key is in the table, but the element to be added is not, then this means that we have a hash collision, so we append the element to the list on the existing key

  For 1 and 2, the returned values are tuples with the key and the position of the element in the list associated with the key. In case the key is already in the table and the element is contained by the list associated with the key, then the returned value is -1 (because the element doesn't have to be added again).

- search_element(element) -> ((key, index in the chain), bool) or False

  Searches an element in the hash table. If the hash function result for the element exists as a key in the table, it returns a tuple containing the key and the index in the list(chain) associated with the key.