Symbol Table: it is essentially a wrapper over the HashTable class

Class arguments: size (int), by default 50 -> fixes the size of the array list of the hashtable

Class methods:

addIdentifier(idName: String) -> public method that calls the addItem method from the hashtable object with the identifier name as its argument; it adds the item to hashtable and returns the position which is now occupied by the newly added item

getIdentifier (idName: String) –> public method that calls the getItem method from the hashtable object with the identifier name as its argument; it returns the item's position in the hashtable if it exists, null otherwise

toString() - > formats the content of the class in a pretty manner

HashTable:

Class arguments: size (int) -> it is passed from symbol table class

Class methods:

hashItem (elem T) : Int -> it hashes the element into an integer smaller than the size of the hashtable

getItem(elem T): Pair(Int, Int) -> checks if the elem exists and returns its position in the array as a pair of integers

addItem(elem T): Pair(Int, Int) -> checks if the elem already exists and if so it returns its position, otherwise it adds the element on the first free position from the sublist found on the hash's result row