https://github.com/biancaszekely32/FLCD

Documentation

This code implements a symbol table, which is a data structure associating values (symbols) with keys (positions). The symbol table supports operations like adding symbols, finding the position of a symbol, and retrieving symbols by their respective positions.

The symbol table is built on top of a hash table. This hash table uses a list-based approach, where each position in the table contains another list. This ensures that values hashing to the same position are stored in the same list. The hash table can be used with both integer and string symbols, dynamically adjusting its operations based on the provided symbol type. Additionally, to handle collisions, the separate chaining technique is used.

Classes:

- Pair - represents a pair of values (position and symbol). It is used to return positions from the hash table.
  - *Pair(Integer pos, Integer symbol):* Constructor to initialize a pair with a position and a symbol.
  - *getPos():* Returns the position.
  - *getSymbol():* Returns the symbol.
  - *toString():* Returns a string representation of the pair.

- HashTable - implements the hash table data structure.
  - *HashTable(Integer size):* Constructor to initialize a hash table with a specified size.
  - *hash(T key):* Computes the hash value for a given key based on its type:
    - For integers: It finds the remainder when dividing the integer key by the size of the hash table (size). This gives a position in the hash table.
    - For strings: It iterates through each character in the string and updates the hash value using a formula. This formula helps spread out hash values to reduce collisions. The final hash value is made non-negative.
  - *getSize():* Returns the size of the hash table.
  - *findPosOfKey(T key):* Finds the position of a key in the hash table.
  - *put(T key):* Inserts a key into the hash table, returns true if the key does not already exists in the table, otherwise returns false.
  - *getByPos(int pos):* Retrieves the symbol corresponding to a given position in the table.
  - *displayTable():* Returns a string representation of the hash table.

- SymbolTable - wraps the HashTable class to provide a symbol table interface.
  - *SymbolTable(Integer size):* Constructor to initialize a symbol table with a specified size.
  - *addSymbol(T symbol):* Adds a symbol to the symbol table. Returns true if the symbol was added successfully, false if it already exists.
  - *getSymbolPosition(T symbol):* Retrieves the position of a symbol in the symbol table.
  - *getSymbolByPos(int position):* Retrieves a symbol given a position in the table.
  - *displayTable():* Returns a string representation of the hash table.