Practical Assignment

BM40A1500 Data Structures and Algorithms

### 1. Implementing the Hash Table

#### 1.1 Structure of the hash table

Present the structure of the hash table.

My hash table contains a list which contains *n* linked lists when *n* is the size of the hash table. It means that I have an own linked list for each slot of my hash table. Each value added to the hash table has a key which is a number from 0 to *n*-1. It defines the slot of the value in the hash table.

Class ‘Node’ is for each node of each linked list. Class ‘LinkedList’ is for the *n* linked lists. Class ‘HashTable’ contains the hash table itself.

#### 1.2 Hash function

What hashing function you used and why?

I used string folding as my hashing function. I started from it as it was mentioned in the task description. It worked very well, and I couldn’t find any better solutions, so I decided to use it. When using string folding, the words in the task split evenly between the slots which is important when using big data.

#### 1.3 Methods

What methods (including the required) your hash table has? Explain briefly how do they work?

The method *hashValue* is for calculating the key for each value in the hash table. It takes the value as an input parameter and modifies it to string. Then the method calculates and returns the key for the value using the formula for string folding.

The method *insert* is for adding new values to the hash table. It calls the

### 2. Testing and Analyzing the Hash Table

#### 2.1 Running time analysis of the hash table

* What is the running time of adding a new value in your hash table and why?
* What is the running time of finding a new value in your hash table and why?
* What is the running time of removing a new value in your hash table and why?

Use Θ notation. Consider what factors influence the running time of the methods.

### 3. The Pressure Test

Table 1. Results of the pressure test.

|  |  |
| --- | --- |
| **Step** | **Time (s)** |
| Initializing the hash table |  |
| Adding the words |  |
| Finding the common words |  |

#### 3.1 Comparison of the data structures

Which data structure was faster in adding the words from the file and why? In which data structure was the search faster and why?

#### 3.2 Further improvements

Are you able to make the program faster?

* Try to change the size of the hash table.
* How well is the data distributed in the hash table?

### List of references

Add here the references and source that you used.