МИНИСТЕРСТВО ЦИФРОВОГО РАЗВИТИЯ, СВЯЗИ И МАССОВЫХ

КОММУНИКАЦИЙ РОССИЙСКОЙ ФЕДЕРАЦИИ

Ордена Трудового Красного Знамени федеральное государственное бюджетное образовательное учреждение высшего образования   «Московский технический университет связи и информатики»



Кафедра"Системное программирование"

Лабораторная работа № 3 Вариант № 9

По дисциплине

“Информационные технологии и программирование”

                                      Выполнил: студент гр. БВТ2201

Шамсутдинов Р.Ф.

Москва, 2023 г

Цели и задачи:

Реализовать хеш таблицу на языке программирования Java.

Ход работы:

Создаем класс Item. Он будет реализовавать элемент связного списка с полями ключ, значение и следующий элемент.

public class Item<K, V>{  
 private K key;  
 private V value;  
 private Item<K, V> next;  
 public Item(K key, V value){  
 this.key = key;  
 this.value = value;  
 }  
 public K getKey() {  
 return key;  
 }  
 public V getValue() {  
 return value;  
 }  
 public Item<K, V> getNext() {  
 return next;  
 }  
 public void setNext(Item<K, V> next) {  
 this.next = next;  
 }  
 public void setValue(V value) {  
 this.value = value;  
 }  
 @Override  
 public String toString(){  
 return "key = " + key.toString() + ", value = " + value.toString();  
 }  
}

Создаем класс HashTable который будет реализовывать хеш таблицу, то есть массив из экземпляров класса Item, а также методы к ним.

public class HashTable<K, V> {  
 private final int SIZE = 3;  
 private Item<K, V>[] table;  
  
 public HashTable(){  
 table = new Item[SIZE];  
 }  
  
 public void put(K key, V value){  
 int hash = key.hashCode() % SIZE;  
 if (hash < 0) {  
 hash = hash \* -1;  
 }  
 Item<K, V> i = table[hash];  
 if (i == null) {  
 table[hash] = new Item<K, V>(key, value);  
 return;  
 }  
 if (i.getKey() == key){  
 i.setValue(value);  
 return;  
 }  
 Item<K,V> next = i.getNext();  
 if (next == null) {  
 i.setNext(new Item<K, V>(key, value));  
 }  
 while(next != null){  
 if (next.getKey() == key){  
 next.setValue(value);  
 return;  
 }  
 if (next.getNext() == null){  
 next.setNext(new Item<K, V>(key, value));  
 return;  
 }  
 next = next.getNext();  
 }  
 }  
  
 public V get(K key){  
 int hash = key.hashCode() % SIZE;  
 if (hash < 0) {  
 hash = hash \* -1;  
 }  
 Item<K, V> i = table[hash];  
 if (i == null){  
 return null;  
 }  
 if (i.getKey() == key){  
 return i.getValue();  
 }  
 Item<K, V> next = i.getNext();  
 while (next != null){  
 if (next.getKey() == key){  
 return next.getValue();  
 }  
 next = next.getNext();  
 }  
 return null;  
 }  
  
 public Item<K, V> remove(K key){  
 int hash = key.hashCode() % SIZE;  
 if (hash < 0) {  
 hash = hash \* -1;  
 }  
 Item<K, V> i = table[hash];  
 if (i == null){  
 return null;  
 }  
 if (i.getKey() == key){  
 table[hash] = i.getNext();  
 i.setNext(null);  
 return i;  
 }  
 Item<K, V> prev = i;  
 Item<K, V> next = i.getNext();  
 while (next != null){  
 if (next.getKey() == key){  
 prev.setNext(next.getNext());  
 next.setNext(null);  
 return next;  
 }  
 prev = next;  
 next = next.getNext();  
 }  
 return null;  
 }  
  
 public int size(){  
 int size = 0;  
 for (int index = 0; index < SIZE; index++){  
 if (table[index] != null){  
 size ++;  
 Item<K, V> next = table[index].getNext();  
 while (next != null){  
 size ++;  
 next = next.getNext();  
 }  
 }  
 }  
 return size;  
 }  
 public boolean isEmpty(){  
 for (int index = 0; index < SIZE; index++){  
 if (table[index] != null){  
 return false;  
 }  
 }  
 return true;  
 }  
 @Override  
 public String toString(){  
 String s = "";  
 for (int index = 0; index < SIZE; index++){  
 if (table[index] != null){  
 s += "Index in table = " + index + "\n";  
 s += "key = " + table[index].getKey().toString() + ", value = " + table[index].getValue().toString() + "\n";  
 Item<K, V> next = table[index].getNext();  
 while (next != null){  
 s += "key = " + next.getKey().toString() + ", value = " + next.getValue().toString() + "\n";  
 next = next.getNext();  
 }  
 }  
 }  
 return s;  
 }  
}

Создаем класс Car который будет содержать информацию о машинах

public class Car{  
 private String brand;  
 private String model;  
 private String year;  
 public Car(String brand, String model, String year){  
 this.brand = brand;  
 this.model = model;  
 this.year = year;  
 }  
 public Car(){  
 this.brand = null;  
 this.model = null;  
 this.year = null;  
 }  
 public String getBrand() {  
 return brand;  
 }  
 public String getModel() {  
 return model;  
 }  
 public String getYear() {  
 return year;  
 }  
 public void setBrand(String brand) {  
 this.brand = brand;  
 }  
 public void setModel(String model) {  
 this.model = model;  
 }  
 public void setYear(String year) {  
 this.year = year;  
 }  
 @Override  
 public String toString(){  
 return brand + " " + model + " " + year;  
 }  
}

Демонстрируем реализацию хеш-таблицы в главном методе класса Main

public class Main {  
 public static void main(String[] args) {  
 HashTable<String, Car> myHashMap = new HashTable<String, Car>();  
  
 System.out.println(myHashMap.isEmpty());  
  
 Car car1 = new Car("BMW", "M5 Competition", "2022");  
 Car car2 = new Car("Porsche", "911 Turbo S", "2020");  
 Car car3 = new Car("Tesla", "model S Plaid", "2023");  
 Car car4 = new Car("Citroen", "DS 20", "1972");  
 Car car5 = new Car("Chevrolet", "Impala", "1967");  
  
 myHashMap.put("т777ор", car1);  
 myHashMap.put("в911тм", car2);  
 myHashMap.put("у100уу", car3);  
 myHashMap.put("SJW 665", car4);  
 myHashMap.put("KAZ 2Y5", car5);  
  
 System.out.println(myHashMap);  
  
 myHashMap.remove("у100уу");  
 myHashMap.remove("в911тм");  
  
 System.out.println(myHashMap);  
 System.out.println(myHashMap.get("KAZ 2Y5"));  
 System.out.println("");  
  
 System.out.println(myHashMap);  
 System.out.println("");  
 myHashMap.put("в911тм", car2);  
 System.out.println(myHashMap.get("в911тм"));  
 myHashMap.put("в911тм", car3);  
 System.out.println(myHashMap.get("в911тм"));  
 System.out.println("");  
 System.out.println(myHashMap);  
  
 System.out.println(myHashMap.isEmpty());  
 System.out.println(myHashMap.size());  
 }  
}

**Получаем результат в консоле**

true

Index in table = 0

key = в911тм, value = Porsche 911 Turbo S 2020

key = у100уу, value = Tesla model S Plaid 2023

key = KAZ 2Y5, value = Chevrolet Impala 1967

Index in table = 1

key = SJW 665, value = Citroen DS 20 1972

Index in table = 2

key = т777ор, value = BMW M5 Competition 2022

Index in table = 0

key = KAZ 2Y5, value = Chevrolet Impala 1967

Index in table = 1

key = SJW 665, value = Citroen DS 20 1972

Index in table = 2

key = т777ор, value = BMW M5 Competition 2022

Chevrolet Impala 1967

Index in table = 0

key = KAZ 2Y5, value = Chevrolet Impala 1967

Index in table = 1

key = SJW 665, value = Citroen DS 20 1972

Index in table = 2

key = т777ор, value = BMW M5 Competition 2022

Porsche 911 Turbo S 2020

Tesla model S Plaid 2023

Index in table = 0

key = KAZ 2Y5, value = Chevrolet Impala 1967

key = в911тм, value = Tesla model S Plaid 2023

Index in table = 1

key = SJW 665, value = Citroen DS 20 1972

Index in table = 2

key = т777ор, value = BMW M5 Competition 2022

false

4

Вывод: Проделав данную работу я понял как устроены хеш-таблицы и связные списки, а также реализовал на языке программирования Java

Ссылка на репозиторий: https://github.com/Sypoo1/mtuci-java-labs/tree/main/thirdLab