|  |
| --- |
|  |
| МИНОБРНАУКИ РОССИИ |
| *Федеральное государственное бюджетное образовательное учреждение высшего образования* ***«МИРЭА – Российский технологический университет»***  **РТУ МИРЭА** |

**Институт информационных технологий (ИТ)**

**Кафедра инструментального и прикладного программного обеспечения (ИиППО)**

**Дисциплина «Программирование на языке Джава»**

**ОТЧЕТ**

**ПО ПРАКТИЧЕСКОМУ ЗАНЯТИЮ №19, №20**

Выполнил студент группы ИНБО-02-20 Самойленко М. А.

Принял Степанов П.В.

Практические работы выполнены «\_\_\_»\_\_\_\_\_\_\_2020г.

«\_\_\_\_\_\_\_\_\_\_\_\_\_\_» «\_\_\_»\_\_\_\_\_\_\_2020г.

Отметка о выполнении

**Москва – 2020 г.**

**Практическое занятие №19-20**

**Задание:**

1. Напишите генератор «красивых» автомобильных номеров. Используйте правила генерации номеров для получения более 2 млн номеров:

* X, Y, Z — различные буквы автомобильного номера ("А", "В", "Е", "К", "М", "Н", "О", "Р", "С", "Т", "У", "Х"), N — цифры, R — регион (от 01 до 199);
* XNNNYZR — пример, A111BC197, Y777HC66

1. Используя генератор «красивых» методов проведите поиск номера, введенного из консоли, с помощью методов:

* прямым перебором по ArrayList, (array.contains())
* бинарным поиском по сортированному ArrayList, (Collections.binarySearch())
* поиском в HashSet, (setHash.contains())
* поиском в TreeSet. (setTree.contains())

1. Измерьте и сравните длительность каждого метода поиска. Формат вывода результатов поиска:

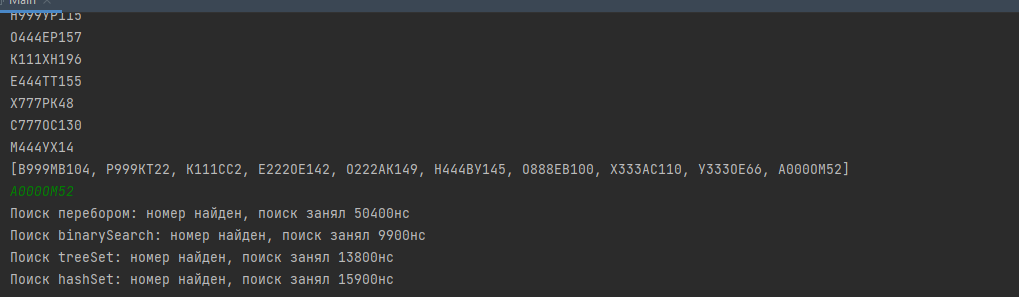
* Поиск перебором: номер найден/не найден, поиск занял 34нс
* Бинарный поиск: номер найден/не найден, поиск занял 34нс
* Поиск в HashSet: номер найден/не найден, поиск занял 34нс

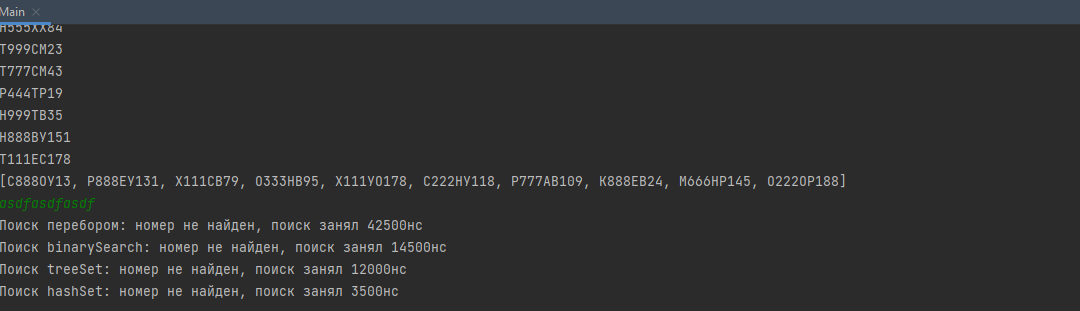
Поиск в TreeSet: номер найден/не найден, поиск занял 34нс

**Код программы:**

package ru.mirea.java.practice19.practice20;  
  
import java.util.\*;  
  
public class Main {  
 static Map<Integer, String> *letters* = new HashMap<>();  
 static int[] *masDigits* = new int[]{0, 1, 2, 3, 4, 5, 6, 7, 8, 9};  
 public static String genNumber() {  
 int dig = (int) (Math.*random*() \* 10 + 0);  
 String ret = *letters*.get((int)(Math.*random*() \* 12 + 0)) + *masDigits*[dig]+*masDigits*[dig]+*masDigits*[dig] +  
 *letters*.get((int)(Math.*random*() \* 12 + 0))+*letters*.get((int)(Math.*random*() \* 12 + 0))+ (int)(Math.*random*() \* 199 + 1) ;  
 return ret;  
 }  
  
 public static void main(String[] args) {  
 *letters*.put(0, "А");  
 *letters*.put(1, "В");  
 *letters*.put(2, "Е");  
 *letters*.put(3, "К");  
 *letters*.put(4, "М");  
 *letters*.put(5, "Н");  
 *letters*.put(6, "О");  
 *letters*.put(7, "Р");  
 *letters*.put(8, "С");  
 *letters*.put(9, "Т");  
 *letters*.put(10, "У");  
 *letters*.put(11, "Х");  
 for (int i = 0; i < 10; i++) {  
 System.*out*.println(*genNumber*());  
 }  
 ArrayList<String> arrayList = new ArrayList<>();  
 TreeSet<String> treeSet = new TreeSet<>();  
 HashSet<String> hashSet = new HashSet<>();  
  
 for (int i = 0; i < 100000; i++){  
 String k = *genNumber*();  
 arrayList.add(k);  
 treeSet.add(k);  
 hashSet.add(k);  
 }  
 System.*out*.println(arrayList);  
  
 Scanner in = new Scanner(System.*in*);  
 boolean t;  
 long beginTime, elapsedTime;  
 int tt;  
 String k1 = in.next();  
 beginTime = System.*nanoTime*();  
 t = arrayList.contains(k1);  
 elapsedTime = System.*nanoTime*() - beginTime;  
 if (t) System.*out*.println("Поиск перебором: номер найден, поиск занял " + elapsedTime + "нс");  
 else System.*out*.println("Поиск перебором: номер не найден, поиск занял " + elapsedTime +"нс");  
 Collections.*sort*(arrayList);  
 beginTime = System.*nanoTime*();  
 tt = Collections.*binarySearch*(arrayList, k1);  
 elapsedTime = System.*nanoTime*() - beginTime;  
 if (tt >=0) System.*out*.println("Поиск binarySearch: номер найден, поиск занял " + elapsedTime + "нс");  
 else System.*out*.println("Поиск binarySearch: номер не найден, поиск занял " + elapsedTime +"нс");  
 beginTime = System.*nanoTime*();  
 t = treeSet.contains(k1);  
 elapsedTime = System.*nanoTime*() - beginTime;  
 if (t) System.*out*.println("Поиск treeSet: номер найден, поиск занял " + elapsedTime + "нс");  
 else System.*out*.println("Поиск treeSet: номер не найден, поиск занял " + elapsedTime +"нс");  
 beginTime = System.*nanoTime*();  
 t = hashSet.contains(k1);  
 elapsedTime = System.*nanoTime*() - beginTime;  
 if (t) System.*out*.println("Поиск hashSet: номер найден, поиск занял " + elapsedTime + "нс");  
 else System.*out*.println("Поиск hashSet: номер не найден, поиск занял " + elapsedTime +"нс");  
  
 }  
}

**Тесты**





**Выводы**

Я научился создавать хеш-таблицу, бинарные деревья поиска с помощью встроенных библиотек.