|  |
| --- |
|  |
| 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 Баринов.И.В.

Принял Степанов П.В.

Практические работы выполнены «\_\_\_»\_\_\_\_\_\_\_2021г.

«\_\_\_\_\_\_\_\_\_\_\_\_\_\_» «\_\_\_»\_\_\_\_\_\_\_2021г.

Отметка о выполнении

**Москва – 2021 г.**

## **Задание**

Написать генератор «красивых» автомобильных номеров с помощью правил генерации номеров

Сгенерировать N-номеров и провести поиск номера, введенного из консоли

Измерить и сравнить длительность каждого метода поиска.

## **Ход Работы**

В ходе выполнения работы были получены следующие исходные коды:

package Practise19\_20;  
  
import java.util.\*;  
  
public class Main {  
 static Map<Integer, String> *letters* = new HashMap<>();  
 static int[] *digits* = new int[]{0, 1, 2, 3, 4, 5, 6, 7, 8, 9};  
 public static String genNumber() {  
 int dig = (int) (Math.*random*() \* 10 + 0);  
 String generatedNum = *letters*.get((int)(Math.*random*() \* 12 + 0)) + *digits*[dig]+*digits*[dig]+*digits*[dig] +  
 *letters*.get((int)(Math.*random*() \* 12 + 0))+*letters*.get((int)(Math.*random*() \* 12 + 0))+ (int)(Math.*random*() \* 199 + 1) ;  
 return generatedNum;  
 }  
  
 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 < 10; i++){  
 String k = *genNumber*();  
 arrayList.add(k);  
 treeSet.add(k);  
 hashSet.add(k);  
 }  
 System.*out*.println(arrayList);  
  
 Scanner in = new Scanner(System.*in*);  
 boolean flag;  
 long beginTime, elapsedTime;  
 int time;  
 System.*out*.println("Введите искомый номер:");  
 System.*out*.println();  
 String k1 = in.next();  
 beginTime = System.*nanoTime*();  
 flag = arrayList.contains(k1);  
 elapsedTime = System.*nanoTime*() - beginTime;  
 if (flag) 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*();  
 time = Collections.*binarySearch*(arrayList, k1);  
 elapsedTime = System.*nanoTime*() - beginTime;  
 if (time >=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*();  
 flag = treeSet.contains(k1);  
 elapsedTime = System.*nanoTime*() - beginTime;  
 if (flag) 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*();  
 flag = hashSet.contains(k1);  
 elapsedTime = System.*nanoTime*() - beginTime;  
 if (flag) System.*out*.println("Поиск hashSet: номер найден, поиск занял " + elapsedTime + "нс");  
 else System.*out*.println("Поиск hashSet: номер не найден, поиск занял " + elapsedTime +"нс");  
  
 }  
}

## **Вывод**

В ходе выполнения работы я решил поставленные задачи и научился измерять время работы методов на Java

GitHub: https://github.com/BarinovIvan/RuPractises/tree/master/src/Practise19\_20