|  |
| --- |
|  |
| 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 ПРАКТИЧЕСКИМ РАБОТАМ №19-20**

Выполнил студент группы ИНБО-02-20 Бойко К.В.

Принял Степанов П.В.

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

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

Отметка о выполнении

**Москва – 2021 г.**

1. **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

2. Используя генератор «красивых» номеров сгенерируйте N-номеров и 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())

3. 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

1. **Ход Работы**

(основной код доступен по ссылке <https://github.com/Zyablikitsme/Java_practice>)

В ходе выполнения работы были получены следующие исходные коды:

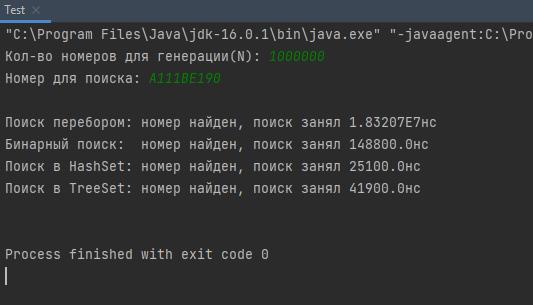
**Generator.java**

package ru.mirea.java.practice19\_20;  
  
import java.util.Random;  
  
public class Generator {  
 private String[] autoNumbLetters = {"А", "В", "Е", "К", "М", "Н", "О", "Р", "С", "Т", "У", "Ч"};  
  
 public String generate(){  
 Random randomizer = new Random();  
  
 String firstLetter = autoNumbLetters[randomizer.nextInt(autoNumbLetters.length)];  
 int middleNumbers = randomizer.nextInt(10);  
 String secondLetter = autoNumbLetters[randomizer.nextInt(autoNumbLetters.length)];  
 String thirdLetter = autoNumbLetters[randomizer.nextInt(autoNumbLetters.length)];  
 int region = randomizer.nextInt(200);  
  
 return (firstLetter + String.*valueOf*(middleNumbers).repeat(3) + secondLetter + thirdLetter + region);  
 }  
}

**Test.java**

package ru.mirea.java.practice19\_20;  
  
import java.util.\*;  
  
public class Test {  
 static public void arrTest(ArrayList<String> arr, String autoNumberForFind){  
 System.out.print("\nПоиск перебором: номер");  
  
 long nanos = System.nanoTime();  
  
 if (arr.contains(autoNumberForFind)) System.out.print(" найден,");  
 else System.out.print(" не найден,");  
  
 System.out.print(" поиск занял " + (double) (System.nanoTime() - nanos) + "нс");  
 }  
  
 static public void arrBinaryTest(ArrayList<String> arr, String autoNumberForFind){  
 System.out.print("\nБинарный поиск: номер");  
  
 Collections.sort(arr);  
  
 long nanos = System.nanoTime();  
  
 int findedArrIndex = Collections.binarySearch(arr, autoNumberForFind);  
  
 if (findedArrIndex >= 0) System.out.print(" найден,");  
 else System.out.print(" не найден,");  
  
 System.out.print(" поиск занял " + (double) (System.nanoTime() - nanos) + "нс");  
 }  
  
 static public void hashSetTest(HashSet<String> hashS, String autoNumberForFind){  
 System.out.print("\nПоиск в HashSet: номер");  
  
 long nanos = System.nanoTime();  
  
 if (hashS.contains(autoNumberForFind)) System.out.print(" найден,");  
 else System.out.print(" не найден,");  
  
 System.out.print(" поиск занял " + (double) (System.nanoTime() - nanos) + "нс");  
 }  
  
 static public void treeSetTest(TreeSet<String> treeS, String autoNumberForFind){  
 System.out.print("\nПоиск в TreeSet: номер");  
  
 long nanos = System.nanoTime();  
  
 if (treeS.contains(autoNumberForFind)) System.out.print(" найден,");  
 else System.out.print(" не найден,");  
  
 System.out.print(" поиск занял " + (double) (System.nanoTime() - nanos) + "нс\n\n");  
 }  
  
 public static void main(String[] args) {  
 Scanner inp = new Scanner(System.in);  
 Generator gen = new Generator();  
  
 System.out.print("Кол-во номеров для генерации(N): ");  
 int N = inp.nextInt();  
  
 System.out.print("Номер для поиска: ");  
 String autoNumberForFind = inp.next();  
  
 ArrayList<String> arrFirst = new ArrayList<String>();  
 for (int i = 0; i < N; i++) arrFirst.add(gen.generate());  
  
 ArrayList<String> arrSecond = arrFirst;  
 HashSet<String> hashS = new HashSet<String>(arrFirst);  
 TreeSet<String> treeS = new TreeSet<String>(arrFirst);  
  
 arrTest(arrFirst, autoNumberForFind);  
 arrBinaryTest(arrSecond, autoNumberForFind);  
 hashSetTest(hashS, autoNumberForFind);  
 treeSetTest(treeS, autoNumberForFind);  
 }  
}

**Выходные данные программы** (при запуске Test.java)**:**



1. **Вывод**

В ходе данной работы были получены знания по генерации случайных чисел с помощью класса Random, были изучены методы поиска различных структур данных в Java.