|  |
| --- |
|  |
| 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 ЗАДАНИЮ №12**

Выполнил студент группы ИНБО-02-20 Веселкин.И.А

Принял Степанов П.В.

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

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

Отметка о выполнении

**Москва – 2021 г.**

## **Задание**

**Задание 1.**

Разработать класс Person, в котором имеется функция, возвращающая Фамилию И.О. Функция должна учитывать возможность отсутствия значений в полях Имя и Отчество. Программу оптимизировать с точки зрения быстродействия.

**Задание 2.**

Доработать класс адреса, который из полученной строки формата "Страна[d] Регион[d] Город[d] Улица[d] Дом[d] Корпус[d] Квартира" ([d] – разделитель, например, «запятая») выбирает соответствующие части и записывает их в соответствующие поля класса Address. Учесть, что в начале и конце разобранной части адреса не должно быть пробелов. Все поля адреса строковые. Разработать проверочный класс не менее чем на четыре адресных строки. В программе предусмотреть две реализации этого метода:

а) разделитель – только запятая (использовать метод split()); Внимание, при разработке нужно учесть, что

б) разделитель – любой из символов ,.;- (класс StringTokenizer).

**Задание 3.**

Реализуйте класс Shirt: Метод toString() выводит объяснение и значение полей построчно.

Дан также строковый массив: shirts[0] = "S001,Black Polo Shirt,Black,XL"; shirts[1] = "S002,Black Polo Shirt,Black,L"; shirts[2] = "S003,Blue Polo Shirt,Blue,XL"; shirts[3] = "S004,Blue Polo Shirt,Blue,M"; shirts[4] = "S005,Tan Polo Shirt,Tan,XL"; shirts[5] = "S006,Black T-Shirt,Black,XL"; shirts[6] = "S007,White T-Shirt,White,XL"; shirts[7] = "S008,White T-Shirt,White,L"; shirts[8] = "S009,Green T-Shirt,Green,S"; shirts[9] = "S010,Orange T-Shirt,Orange,S"; shirts[10] = "S011,Maroon Polo Shirt,Maroon,S";

Преобразуйте строковый массив в массив класса Shirt и выведите его на консоль.

**Задание 4.**

Разработайте класс, который получает строковое представление телефонного номера в одном из двух возможных строковых форматов:

+<Код страны><Номер 10 цифр>, например “+79175655655” или

“+104289652211” или

8<Номер 10 цифр> для России, например “89175655655”

и преобразует полученную строку в формат:

+<Код страны><Три цифры>–<Три цифры>–<Четыре цифры>

**Задание 5.**

В методе main считай с консоли имя файла, который содержит слова, разделенные пробелом. В методе getLine() используя StringBuilder расставьте все слова в таком порядке, чтобы последняя буква данного слова совпадала с первой буквой следующего не учитывая регистр. Каждое слово должно участвовать 1 раз.

## **Ход Работы**

Реализация задания 1:

public class Person {  
 private String name, surname, patronymic;  
  
 public Person(String surname) {  
 this.surname = surname;  
 }  
  
 public Person(String name, String surname, String patronymic) {  
 this.name = name;  
 this.surname = surname;  
 this.patronymic = patronymic;  
 }  
 public String getName(){  
 StringBuilder sb = new StringBuilder(surname);  
 if(name != null && ! name.equals(""))  
 sb.append(" ").append(name);  
 if(patronymic != null && ! patronymic.equals(""))  
 sb.append(" ").append(patronymic);  
 return sb.toString();  
 }  
  
 public static void main(String[] args) {  
 Person p1 = new Person("Cоловьёв");  
 Person p2 = new Person("Родион", "Васнецов", "Иванович");  
 Person p3 = new Person("Николай", "Романов", "Александрович");  
 Person p4 = new Person("Николай","Андропов","Владимирович");  
  
 System.*out*.println(p1.getName());  
 System.*out*.println(p2.getName());  
 System.*out*.println(p3.getName());  
 System.*out*.println(p4.getName());  
  
 }

Реализация задания 2:

import java.util.StringTokenizer;  
  
class Address {  
 private String country, region, city, street, house, housing, flat;  
  
 public Address(String stringToParse, boolean multiDivider){  
 if(stringToParse == null) throw new NullPointerException();  
 String[] s;  
 if(multiDivider){  
 StringTokenizer st = new StringTokenizer(stringToParse, ",.;");  
 s = new String[st.countTokens()];  
 int i = 0;  
 while(st.hasMoreTokens()) s[i++] = st.nextToken();  
 }else{  
 s = stringToParse.split(",");  
 }  
  
 if(s.length < 7) throw new IllegalArgumentException("Строка содержит недостаточно данных!");  
 country = s[0].trim();  
 region = s[1].trim();  
 city = s[2].trim();  
 street = s[3].trim();  
 house = s[4].trim();  
 housing = s[5].trim();  
 flat = s[6].trim();  
 }  
  
 @Override  
 public String toString() {  
 return "Адрес: " +  
 "страна='" + country + '\'' +  
 ", регион='" + region + '\'' +  
 ", город='" + city + '\'' +  
 ", улица='" + street + '\'' +  
 ", дом='" + house + '\'' +  
 ", корпус='" + housing + '\'' +  
 ", квартира='" + flat + '\'';  
 }  
  
 public static void main(String[] args) {  
 Address a1 = new Address("Россия, МО, Москва, Севастопольский проспект, 78, 7, 28", false);  
 Address a2 = new Address("Россия, Ленинградская область, Санкт-Петербург, Невсий проспект, 234, 1, 23", true);  
 Address a3 = new Address("США, Нью-Йорк, Нью-Йорк, Вуд стрит, 9211, 1, 32", false);  
 Address a4 = new Address("Россия, Калининградская область, Калининград, Серпуховская, 21, 1, 65", true);  
  
 System.*out*.println(a1);  
 System.*out*.println(a2);  
 System.*out*.println(a3);  
 System.*out*.println(a4);  
  
 }  
}

Реализация задания 3:

public class PhoneParser {  
 private String code, city, triplet, quadruple;  
  
 public PhoneParser(String phone){  
 if(phone.startsWith("8")){  
 code = "+7";  
 }else{  
 code = phone.substring(0, phone.length()-10);  
 }  
 city = phone.substring(phone.length()-10, phone.length()-7);  
 triplet = phone.substring(phone.length()-7, phone.length()-4);  
 quadruple = phone.substring(phone.length()-4);  
 }  
  
 @Override  
 public String toString() {  
 return code +"(" + city+ ")" + "-" +"("+ triplet +")" + "-" +"("+ quadruple +")" ;  
 }  
  
 public static void main(String[] args) {  
 System.*out*.println(new PhoneParser("88005553535"));  
 System.*out*.println(new PhoneParser("+78005553535"));  
 System.*out*.println(new PhoneParser("+1238005553535"));  
 System.*out*.println(new PhoneParser("+118005553535"));  
 }

}

Реализация задания 4:

public class Shirt {  
 private String article, fullName, color, size;  
  
 public Shirt(String stringToParse){  
 String[] s = stringToParse.split(",");  
 article = s[0];  
 fullName = s[1];  
 color = s[2];  
 size = s[3];  
 }  
  
 @Override  
 public String toString() {  
 return "Shirt:\n" +  
 "\tarticle='" + article + '\'' +  
 "\n\tfullName='" + fullName + '\'' +  
 "\n\tcolor='" + color + '\'' +  
 "\n\tsize='" + size + '\'';  
 }  
  
 public static void main(String[] args) {  
 String[] strs = new String[]{  
 "S001,Black Polo Shirt,Black,XL",  
 "S002,Black Polo Shirt,Black,L",  
 "S003,Blue Polo Shirt,Blue,XL",  
 "S004,Blue Polo Shirt,Blue,M",  
 "S005,Tan Polo Shirt,Tan,XL",  
 "S006,Black T-Shirt,Black,XL",  
 "S007,White T-Shirt,White,XL",  
 "S008,White T-Shirt,White,L",  
 "S009,Green T-Shirt,Green,S",  
 "S010,Orange T-Shirt,Orange,S",  
 "S011,Maroon Polo Shirt,Maroon,S"  
 };  
 Shirt[] shirts = new Shirt[strs.length];  
 for (int i = 0; i < strs.length; i++) {  
 shirts[i] = new Shirt(strs[i]);  
 System.*out*.println(shirts[i]);  
 }  
 }  
}

Реализация задания 5:

import java.io.FileReader;  
import java.io.IOException;  
import java.util.\*;  
  
public class WordConcatenator {  
 public static void main(String[] args) {  
 Scanner sc = new Scanner(System.*in*);  
 String path = sc.nextLine();  
 StringBuilder s = new StringBuilder();  
 try(FileReader fr = new FileReader(path)){  
 int c;  
 while((c=fr.read())!=-1){  
 s.append((char) c);  
 }  
 }catch (IOException e){  
 e.printStackTrace();  
 }  
 System.*out*.println(*getLine*(s.toString()));  
 }  
  
 private static String getLine(String s) {  
 Queue<String> toAdd = new ArrayDeque<>(Arrays.*asList*(s.split(" ")));  
 Set<String> added = new HashSet<>();  
 added.add(toAdd.peek());  
 StringBuilder sb = new StringBuilder(toAdd.remove());  
 boolean wasAdded = true;  
  
 while(!toAdd.isEmpty() && wasAdded){  
 wasAdded = false;  
 int i = 0;  
 for (; i < toAdd.size(); i++) {  
 String cur = toAdd.remove();  
 if(added.contains(cur)) continue;  
 if(cur.startsWith(sb.substring(sb.length()-1).toLowerCase()) ||  
 cur.startsWith(sb.substring(sb.length()-1).toUpperCase())){  
 sb.append(" ").append(cur);  
 wasAdded = true;  
 added.add(cur);  
 break;  
 }  
 toAdd.add(cur);  
 }  
 for(; i< toAdd.size(); i++){  
 String cur = toAdd.remove();  
 if(added.contains(cur)) continue;  
 toAdd.add(cur);  
 }  
 }  
 return sb.toString();  
 }  
}

# **Вывод**

В ходе выполнения работы я закрепил знания в области обработки строк, научился применять методы класса String и других классов для обработки строк.

С полным кодом программы можно ознакомится по ссылке: https://github.com/gost214/Practika-INBO-02-20-Veselkin/tree/main/pract12