***class Main***

package com.company.bank\_account;  
  
import java.util.stream.Collectors;  
  
import java.util.\*;  
import java.util.function.Consumer;  
import java.util.stream.Stream;  
  
*/\*\*  
 \* Задание.  
 \* 1. Разработать класс «Банковский счет», который содержит информацию о денежных средствах  
 \* на данном счете, а так же информацию о валюте данного счета (реализацию валют выбрать самостоятельно,  
 \* от String до выделенного класса или класса со статическими полями, рекомендуется реализовать аналог  
 \* перечисления enumeration). Добавить в класс все необходимые методы.  
 \* 2. Выполнить задания, указанные ниже, используя stream API  
 \* 1) Сформировать список счетов со случайными значениями баланса, но строго заданными значениями валют  
 \* при инициализации (можно использовать stream.of(<валюты>).map(<приведение к счету)>)  
 \* 2) Вывести список на экран через forEach (в дальнейшем использовать данную команду для промежуточного вывода)  
 \* 3) Вывести отсортированный список по балансу  
 \* 4) Вывести отсортированный список по названию валют  
 \* 5) Вывести все долларовые счета  
 \* 6) Вывести топ-3 рублевых счетов  
 \* 7) Вывести сумму всех счетов валюты «евро»  
 \* 8) Посчитать среднее значение (среднее арифметическое) для рублевых счетов  
 \* 9) Получить set с валютами (существующие в списке валюты) из списка счетов  
 \* 10) Конвертировать все счета в доллары (использовать замыкание для передачи коэффициентов, коэффициенты желательно хранить во вспомогательном классе, здесь хорошо будет реализовать анонимный класс и реализацию соответствующего интерфейса).  
 \*/*public class Main {  
  
 public static void main(String[] *args*) {  
 // 1. Формирование списка счетов со случайными значениями баланса, но строго заданными значениями валют  
 List<BankAccount> *account\_1* = Stream.*generate*(() -> new BankAccount("RUB", Math.*random*() \* 5000, (int) (Math.*random*() \* (1000)))).limit(5).toList();  
 List<BankAccount> *account\_2* = Stream.*generate*(() -> new BankAccount("USD", Math.*random*() \* 5000, (int) (Math.*random*() \* (1000)))).limit(4).toList();  
 List<BankAccount> *account\_3* = Stream.*generate*(() -> new BankAccount("EUR", Math.*random*() \* 5000, (int) (Math.*random*() \* (1000)))).limit(3).toList();  
  
 // Объединение счетов в один список  
 Stream<BankAccount> *combinedStream* = Stream.*concat*(  
 Stream.*concat*(*account\_1*.stream(), *account\_2*.stream()),  
 *account\_3*.stream());  
 List<BankAccount> *account* = *combinedStream* .collect(Collectors.*toList*());  
  
 // 2. Вывод списка на экран  
 System.*out*.println("\nп.1 и 2 Формирование счетов и вывод списка счетов на экран:");  
 Consumer<BankAccount> *printAccount* = System.*out*::println; // переменная, которая содержит лямбду  
 *account*.forEach(*printAccount*);  
  
 // 3. Вывод отсортированного списка по балансу  
 System.*out*.println("\n3. Вывод отсортированного списка по балансу:");  
 *account*.stream().sorted(Comparator.*comparingDouble*(BankAccount::getBalance))  
 .forEach(*printAccount*);  
  
 // 4. Вывод отсортированного списка по названию валют  
 System.*out*.println("\n4. Вывод отсортированного списка по названию валют:");  
 *account*.stream().sorted(Comparator.*comparing*(BankAccount::getCurrency))  
 .forEach(*printAccount*);  
  
 // 5. Вывод всех USD счетов  
 System.*out*.println("\n5. Вывод всех USD счетов:");  
 *account*.stream()  
 .filter((*acc*) -> "USD".equals(*acc*.getCurrency()))  
 .forEach(*printAccount*);  
  
 // 6. Вывод топ-3 рублевых счетов  
 System.*out*.println("\n6. Вывод топ-3 рублевых счетов:");  
 *account*.stream()  
 .filter((*acc*) -> "RUB".equals(*acc*.getCurrency()))  
 .sorted(Comparator.*comparingDouble*(BankAccount::getBalance).reversed())  
 .limit(3)  
 .forEach(*printAccount*);  
  
 // 7. Вывод суммы всех счетов валюты EUR  
 System.*out*.println("\n7. Вывод суммы всех счетов валюты EUR:");  
 BankAccount *commonAccount* = *account*.stream().filter((*acc*) -> "EUR".equals(*acc*.getCurrency()))  
 .reduce(new BankAccount("EUR", 0),  
 (*sum*, *x*) -> new BankAccount("EUR", *sum*.getBalance() + *x*.getBalance(), 1)  
 );  
 System.*out*.println("Все EUR на " + *commonAccount*);  
  
 // 8. Вывод среднего значения (среднее арифметическое) для рублевых счетов  
 System.*out*.println("\n8. Вывод среднего значения (среднее арифметическое) для рублевых счетов:");  
 OptionalDouble *average* = *account*.stream().filter((*acc*) -> "RUB".equals(*acc*.getCurrency())).  
 mapToDouble(BankAccount::getBalance).average();//преобразовываем банковские счета в баланс  
 if (*average*.isPresent()) {  
 System.*out*.printf("Среднее значениее %1.2f %s", *average*.getAsDouble(), " RUB");  
 }  
 System.*out*.println();  
  
 // 9. Set с валютами (существующие в списке валюты) из списка счетов  
 System.*out*.println("\n9. Set с валютами (существующие в списке валюты) из списка счетов:");  
 Set<BankAccount> *currency* = *account*.stream()  
 .filter((*acc*) -> "USD".equals(*acc*.getCurrency()) || "EUR".equals(*acc*.getCurrency()))  
 .collect(Collectors.*toSet*());  
 *currency*.forEach(*printAccount*);  
  
 // 10. Конвертировать все счета в доллары  
 System.*out*.println("\n10. Конвертирование всех счетов в USD:");  
 final double *EUR* = 1.08; // курс Евро к Доллару  
 final double *RUB* = 0.016;// курс Рубля к Доллару  
 *account*.stream().filter(*acc* -> !Objects.*equals*(*acc*.getCurrency(), "USD"))  
 .forEach(*acc* -> {  
 if (*acc*.getCurrency().equals("RUB")) {  
 *acc*.setBalance((*acc*.getBalance() \* *RUB*));  
 *acc*.setCurrency("USD");  
 }  
 if (*acc*.getCurrency().equals("EUR")) {  
 *acc*.setBalance((*acc*.getBalance() \* *EUR*));  
 *acc*.setCurrency("USD");  
 }  
 });  
 *account*.forEach(*printAccount*);  
 }//main  
}//class close

***class BankAccount***

package com.company.bank\_account;  
  
//Класс - "Банковский счёт"  
public class BankAccount {  
  
 //поля  
 private double balance; // баланс  
 private int number; // номер счета  
 private String currency; // валюта  
  
  
 //Конструкторы  
 // 1. Конструктор без параметров  
 public BankAccount() {  
 currency = "RUB";  
 number = 1;  
 balance = 0;  
 }  
  
 // 2. Конструктр с 2 параметрами  
 public BankAccount(String *currency*, double *value*) {  
 this.balance = *value*;  
 this.currency = *currency*;  
 }  
  
 // 3. Конструктр с 3 параметрами  
 public BankAccount(String *currency*, double *value*, int *number*) {  
 this.balance = *value*;  
 this.number = *number*;  
 this.currency = *currency*;  
 }  
  
 //getters and setters  
  
 public double getBalance() {  
 return balance;  
 }  
  
 public void setBalance(double *value*) {  
 this.balance = *value*;  
 }  
  
 public String getCurrency() {  
 return currency;  
 }  
  
 public void setCurrency(String *currency*) {  
 this.currency = *currency*;  
 }  
  
 //метод toString  
  
 @Override  
 public String toString() {  
 return String.*format*("Банковский счёт %d: %1.2f %s", number, balance, currency);  
 }  
}//class close

***Тест:***









