FINAL EXAM RETAKE

ВРЕМЕ:

ЗАДАЧА 1: - 17 мин.

ЗАДАЧА 2: – 36 мин.

ЗАДАЧА 3: – 28 мин

Общо: 1:22 мин

Задача 1:

import java.util.Scanner;  
  
public class first {  
 public static void main(String[] args) {  
 Scanner scanner = new Scanner(System.*in*);  
  
 String text = scanner.nextLine();  
  
 String command = scanner.nextLine();  
 while (!command.equals("Decode")) {  
 String[] tokens = command.split("\\|");  
 String action = tokens[0];  
 StringBuilder sb = new StringBuilder();  
  
 switch (action) {  
 case "Move":  
 int numberToMove = Integer.*parseInt*(tokens[1]);  
 String moveLetters = text.substring(0, numberToMove);  
 String endPart = text.substring(numberToMove);  
  
 sb.append(endPart);  
 sb.append(moveLetters);  
 text = sb.toString();  
 break;  
  
 case "Insert":  
 int index = Integer.*parseInt*(tokens[1]);  
 String value = tokens[2];  
  
 sb = new StringBuilder(text);  
 sb.insert(index, value);  
 text = sb.toString();  
 break;  
  
 case "ChangeAll":  
 String substring = tokens[1];  
 String replacement = tokens[2];  
  
 text = text.replace(substring, replacement);  
  
 break;  
 }  
  
 command = scanner.nextLine();  
 }  
 System.*out*.printf("The decrypted message is: %s", text);  
 }  
}

Задача 2:

import java.util.Scanner;  
import java.util.regex.Matcher;  
import java.util.regex.Pattern;  
  
public class second {  
 public static void main(String[] args) {  
 Scanner scanner = new Scanner(System.*in*);  
  
 String text = scanner.nextLine();  
  
 String regex = "([\\#]|[\\|])(?<itemName>[A-Za-z\\ ]+)(\\1)(?<date>[\\d]{2}\\/[\\d]{2}\\/[\\d]{2})(\\1)(?<calories>\\d+)(\\1)";  
 Pattern pattern = Pattern.*compile*(regex);  
 Matcher matcher = pattern.matcher(text);  
 Pattern patternCalories = Pattern.*compile*(regex);  
 Matcher matcherCalories = patternCalories.matcher(text);  
  
 int sumOfCalories = 0;  
 int lastDays = 0;  
  
 while (matcherCalories.find()) {  
 sumOfCalories += Integer.*parseInt*(matcherCalories.group("calories"));  
  
 }  
 if (sumOfCalories == 0) {  
 lastDays = 0;  
 } else {  
 lastDays = sumOfCalories / 2000;  
 }  
  
 System.*out*.printf("You have food to last you for: %d days!%n", lastDays);  
  
 while (matcher.find()) {  
 System.*out*.printf("Item: %s, Best before: %s, Nutrition: %s%n", matcher.group("itemName"),  
 matcher.group("date"), matcher.group("calories"));  
 }  
 }  
}

Задача 3:

import java.util.Scanner;  
import java.util.TreeMap;  
  
public class third {  
 public static void main(String[] args) {  
 Scanner scanner = new Scanner(System.*in*);  
  
 TreeMap<String, String> composerRegister = new TreeMap<>();  
 TreeMap<String, String> keyRegister = new TreeMap<>();  
  
 int numberOfPieces = Integer.*parseInt*(scanner.nextLine());  
 for (int i = 0; i < numberOfPieces; i++) {  
 String[] elements = scanner.nextLine().split("\\|");  
 String piece = elements[0];  
 String composer = elements[1];  
 String key = elements[2];  
  
 composerRegister.putIfAbsent(piece, composer);  
 keyRegister.putIfAbsent(piece, key);  
 }  
  
 String command = scanner.nextLine();  
 while (!command.equals("Stop")) {  
 String[] tokens = command.split("\\|");  
 String action = tokens[0];  
 String piece = tokens[1];  
  
 switch (action) {  
 case "Add":  
 String composer = tokens[2];  
 String key = tokens[3];  
  
 if (!composerRegister.containsKey(piece)) {  
 composerRegister.put(piece, composer);  
 keyRegister.put(piece, key);  
 System.*out*.printf("%s by %s in %s added to the collection!%n", piece, composer, key);  
 } else {  
 System.*out*.printf("%s is already in the collection!%n", piece);  
 }  
 break;  
  
 case "Remove":  
 if (!composerRegister.containsKey(piece)) {  
 System.*out*.printf("Invalid operation! %s does not exist in the collection.%n", piece);  
 } else {  
 composerRegister.remove(piece);  
 keyRegister.remove(piece);  
 System.*out*.printf("Successfully removed %s!%n", piece);  
 }  
 break;  
  
 case "ChangeKey":  
 String newKey = tokens[2];  
 if (!composerRegister.containsKey(piece)) {  
 System.*out*.printf("Invalid operation! %s does not exist in the collection.%n", piece);  
 } else {  
 keyRegister.put(piece, newKey);  
 System.*out*.printf("Changed the key of %s to %s!%n", piece, newKey);  
 }  
  
 break;  
 }  
 command = scanner.nextLine();  
 }  
 composerRegister  
 .entrySet()  
 .stream()  
 .sorted((f, s) -> {  
 int result = f.getKey().compareTo(s.getKey());  
 if (result == 0) {  
 result = f.getValue().compareTo(s.getValue());  
 }  
 return result;  
 })  
 .forEach(entry -> {  
 System.*out*.printf("%s -> Composer: %s, Key: %s%n", entry.getKey(), entry.getValue(), keyRegister.get(entry.getKey()));  
 });  
 }  
}