**Федеральное агентство связи**

**Государственное бюджетное образовательное учреждение высшего**

**образование**

**Ордена Трудового Красного Знамени**

**«Московский технический университет связи и информатики»**

**Кафедра «МКиИТ»**

**дисциплина «Объектно-ориентированное программирование»**

### [ПРАКТИЧЕСКИЕ ЗАНЯТИЯ](https://lms.mtuci.ru/lms/course/view.php?id=416#section-2)

## Задачи для практических занятий. Уровень сложности: 5/6

Подготовила студентка

группы БВТ1901: Нкурикийе Х

Проверил: Мосева М.С.

Москва 2021

import java.nio.charset.StandardCharsets;

import java.util.\*;

import java.security.\*;

public class task5 {

public static int[] encrypt(String a) { // 5.1

int[] res = new int[a.length()];

int lastChar = 0;

char[] charArray = a.toCharArray();

for (int i = 0; i < charArray.length; i++) {

int charSym = charArray[i];

res[i] = charSym - lastChar;

lastChar = charSym;

}

return res;

}

public static String decrypt(int[] a) {

StringBuilder res = new StringBuilder();

int lastCode = 0;

for (int j: a) {

char newAscii = (char) (lastCode + j);

lastCode = lastCode + j;

res.append(newAscii);

}

return res.toString();

}

public static boolean canMove(String a, String bStr, String cStr) { // 5.2

char[] b = bStr.toCharArray();

char[] c = cStr.toCharArray();

if ((b.length == 2 && c.length == 2) && ((b[0] >= 'A' && b[0] <= 'H') || (b[0] >= 'a' && b[0] <= 'h')) &&

((c[0] >= 'A' && c[0] <= 'H') || (c[0] >= 'a' && c[0] <= 'h')) && (b[1] >= '1' && b[1] <= '8')

&& (c[1] >= '1' && c[1] <= '8')) {

if (a.trim().toLowerCase().equals("пешка")) {

if (b[0] == c[0]) {

if (b[1] == '1') {

return false;

}

if (b[1] == '2' && c[1] == '4') {

return true;

}

int begin = (int) b[1];

int end = (int) c[1];

return begin + 1 == end;

} else {

return false;

}

} else if (a.trim().toLowerCase().equals("конь")) {

return ((Math.abs((b[1] - c[1])) == 2 && Math.abs(b[0] - c[0]) == 1) ||

(Math.abs((b[0] - c[0])) == 2 && Math.abs(b[1] - c[1]) == 1));

} else if (a.trim().toLowerCase().equals("слон")) {

return (Math.abs(b[0] - c[0]) == Math.abs(b[1] - c[1]));

} else if (a.trim().toLowerCase().equals("ладья")) {

return b[0] == c[0] || b[1] == c[1];

} else if (a.trim().toLowerCase().equals("ферзь")) {

return (b[0] == c[0] || b[1] == c[1]) || (Math.abs(b[0] - c[0]) == Math.abs(b[1] - c[1]));

} else if (a.trim().toLowerCase().equals("король")) {

return ((b[0] == c[0]) && (b[1] + 1 == c[1]) || (b[1] == c[1]) && (b[0] + 1 == c[0]) ||

(b[0] == c[0]) && (b[1] - 1 == c[1]) || (b[1] == c[1]) && (b[0] - 1 == c[0]));

} else {

return false;

}

}

else {

return false;

}

}

public static boolean canComplete(String a, String b) { // 5.3

for (char let: a.toCharArray()) {

int curSymPos = b.indexOf(let);

if ((curSymPos != -1)) {

b = b.substring(curSymPos+1);

}

else return false;

}

return true;

}

public static int sumDigProd(int[] arrOfNums) { // 5.4

int sum = 0;

for (int num: arrOfNums) {

sum += num;

}

while (sum / 10 != 0){

int allNumProd = 1;

String temp = Integer.toString(sum);

for (int i = 0; i < temp.length(); i++) {

allNumProd = Character.getNumericValue(temp.charAt(i)) \* allNumProd;

}

sum = allNumProd;

}

return sum;

}

public static ArrayList<String> sameVowelGroup(String[] a) { // 5.5

ArrayList<String> AnsWords = new ArrayList<>(Collections.singletonList(a[0]));

ArrayList<Character> usedVowelsDefault = new ArrayList<>();

for (char symFromMainWord: a[0].toLowerCase().toCharArray()) {

if (!usedVowelsDefault.contains(symFromMainWord)) {

switch (symFromMainWord) {

case 'a' -> usedVowelsDefault.add('a');

case 'e' -> usedVowelsDefault.add('e');

case 'i' -> usedVowelsDefault.add('i');

case 'o' -> usedVowelsDefault.add('o');

case 'u' -> usedVowelsDefault.add('u');

}

}

}

for (int i = 1; i < a.length; i++) {

ArrayList<Character> usedVowelsCur = new ArrayList<>();

for (char checkSym: a[i].toLowerCase().toCharArray()) {

if (!usedVowelsCur.contains(checkSym)) {

switch (checkSym) {

case 'a' -> usedVowelsCur.add('a');

case 'e' -> usedVowelsCur.add('e');

case 'i' -> usedVowelsCur.add('i');

case 'o' -> usedVowelsCur.add('o');

case 'u' -> usedVowelsCur.add('u');

}

}

}

if (usedVowelsDefault.containsAll(usedVowelsCur) && usedVowelsCur.containsAll(usedVowelsDefault)) AnsWords.add(a[i]);

}

return AnsWords;

}

public static boolean validateCard(long a) { // 5.6

String cardNum = Long.toString(a);

int cardNumLength = cardNum.length();

if((cardNumLength > 13) && (cardNumLength < 20)) {

int contralNum = cardNum.charAt(cardNumLength-1) - '0';

int sumOfNums = 0;

for (int i = cardNumLength-2; i > -1; i--) {

int curNum;

if (i % 2 == 0) {

curNum = (cardNum.charAt(i) - '0') \* 2;

if (curNum > 9) sumOfNums += curNum - 9;

else sumOfNums += curNum;

}

else sumOfNums += cardNum.charAt(i) - '0';

}

return (10 - (sumOfNums % 10)) == contralNum;

}

else

return false;

}

public static String numToEng(int a) { // 5.7 eng

String s = Integer.toString(a);

if (s.length() == 1) {

return oneNumber(a);

}

else if (s.length() == 2) {

return twoNumber(a);

}

else {

return threeNumber(a);

}

}

public static String oneNumber(int a) {

return switch (a) {

case 0 -> "zero";

case 1 -> "one";

case 2 -> "two";

case 3 -> "three";

case 4 -> "four";

case 5 -> "five";

case 6 -> "six";

case 7 -> "seven";

case 8 -> "eight";

case 9 -> "nine";

default -> " ";

};

}

public static String twoNumber(int a) {

String res = " ";

String juu = " ";

if (a < 20){

switch (a) {

case 10 -> res = "ten";

case 11 -> res = "eleven";

case 12 -> res = "twelve";

case 13 -> res = "thirteen";

case 14 -> res = "fourteen";

case 15 -> res = "fifteen";

case 16 -> res = "sixteen";

case 17 -> res = "seventeen";

case 18 -> res = "eighteen";

case 19 -> res = "nineteen";

}

}

else {

int b = a / 10;

a %= 10;

switch (b) {

case 2 -> juu = "twenty ";

case 3 -> juu = "thirty ";

case 4 -> juu = "fourty ";

case 5 -> juu = "fifty ";

case 6 -> juu = "sixty ";

case 7 -> juu = "seventy ";

case 8 -> juu = "eighty ";

case 9 -> juu = "ninety ";

}

res = juu + oneNumber(a);

if (a == 0){

res = juu;

}

}

return res;

}

public static String threeNumber(int a) {

String res = " ";

int c = a % 100;

a = a / 100;

res = oneNumber(a) + " hundred " + twoNumber(c);

return res;

}

public static String numToRus(int a) { // 5.7 rus

String s = Integer.toString(a);

if (s.length()==1) {

return oneNumberRus(a);

}

else if (s.length()==2) {

return twoNumberRus(a);

}

else {

return threeNumberRus(a);

}

}

public static String oneNumberRus(int a) {

return switch (a) {

case 0 -> "ноль";

case 1 -> "один";

case 2 -> "два";

case 3 -> "три";

case 4 -> "четыре";

case 5 -> "пять";

case 6 -> "шесть";

case 7 -> "семь";

case 8 -> "восемь";

case 9 -> "девять";

default -> " ";

};

}

public static String twoNumberRus(int a) {

String res = " ";

String juu = " ";

if (a < 20) {

switch (a) {

case 10 -> res = "десять";

case 11 -> res = "одиннадцать";

case 12 -> res = "двенадцать";

case 13 -> res = "тринадцать";

case 14 -> res = "четырнадцать";

case 15 -> res = "пятнадцать";

case 16 -> res = "шестнадцать";

case 17 -> res = "семнадцать";

case 18 -> res = "восемнадцать";

case 19 -> res = "девятнадцать";

}

}

else {

int b = a % 10;

a /= 10;

switch (a) {

case 2 -> juu = "двадцать ";

case 3 -> juu = "тридцать ";

case 4 -> juu = "сорок ";

case 5 -> juu = "пятьдесят ";

case 6 -> juu = "шестьдесят ";

case 7 -> juu = "семьдесят ";

case 8 -> juu = "восемьдесят ";

case 9 -> juu = "девяносто ";

}

res = juu + oneNumberRus(b);

if (b == 0){

res = juu;

}

}

return res;

}

public static String threeNumberRus(int a) {

String res = " ";

String sotnya =" ";

int b = a % 100;

a /= 100;

switch (a) {

case 1 -> sotnya = "сто ";

case 2 -> sotnya = "двести ";

case 3 -> sotnya = "триста ";

case 4 -> sotnya = "четыреста ";

case 5 -> sotnya = "пятьсот ";

case 6 -> sotnya = "шестьсот ";

case 7 -> sotnya = "семьсот ";

case 8 -> sotnya = "восемьсот ";

case 9 -> sotnya = "девятьсот ";

}

if (b == 0) {

res = sotnya;

}

res = sotnya + twoNumberRus(b);

return res;

}

public static String getSha256Hash(String a) { // 5.8

try {

MessageDigest digest = MessageDigest.getInstance("SHA-256");

byte[] hash = digest.digest(a.getBytes(StandardCharsets.UTF\_8));

StringBuilder hexString = new StringBuilder();

for (byte b: hash) {

String hex = Integer.toHexString(0xff & b);

if (hex.length() == 1) hexString.append('0');

hexString.append(hex);

}

return hexString.toString();

} catch (Exception e) {

return "Error";

}

}

public static String correctTitle(String a) { // 5.9

String[] words = a.toLowerCase().split("\\s+");

StringBuilder ans = new StringBuilder();

for (int i = 0; i < words.length; i++){

if (words[i].equals("in") || words[i].equals("of") || words[i].equals("the") || words[i].equals("and"))

ans.append(words[i]).append(" ");

else {

char[] numArr = words[i].toCharArray();

ans.append(Character.toUpperCase(numArr[0])).append(words[i].substring(1)).append(" ");

}

if (i == words.length - 1)

ans = new StringBuilder(ans.substring(0, ans.length() - 1));

}

return ans.toString();

}

public static String hexLattice(int a) { // 5.10

int pointInFirstLine = 0;

int workedNum = a;

int subNum = 1;

while (workedNum > 0) {

pointInFirstLine++;

workedNum -= subNum;

subNum = pointInFirstLine \* 6;

}

if ((((a - 1) % 6 == 0) || (a == 1)) && (workedNum == 0)) {

StringBuilder ans = new StringBuilder();

int dotsInLine = pointInFirstLine;

for (int i = pointInFirstLine; i > 0; i--){

ans.append(" ".repeat(i - 1));

ans.append(" o".repeat(Math.max(0, dotsInLine)));

dotsInLine++;

ans.append("\n");

}

for (int i = dotsInLine-1; i > pointInFirstLine; i--){

ans.append(" ".repeat(Math.max(0, (dotsInLine) - i)));

ans.append(" o".repeat(Math.max(0, i - 1)));

ans.append("\n");

}

return ans.toString();

}

else

return "Invalid";

}

public static void main(String[] args) {

Scanner in = new Scanner(System.in).useLocale(Locale.ENGLISH);

Scanner sc = new Scanner(System.in);

System.out.println("Введите номер задачи (от 41 до 50):");

int n = in.nextInt();

switch (n) {

case 41 -> {

System.out.println("encrypt() - 1");

System.out.println("decrypt() - 2");

int n1 = in.nextInt();

if (n1 == 1) {

System.out.println("Введите строку:");

String s1 = sc.nextLine();

int[] res = encrypt(s1);

System.out.println("Результат: ");

for (int i: res) {

System.out.print(i + " ");

}

}

else if (n1 == 2) {

System.out.println("Введите количество элементов массива:");

int len1 = in.nextInt();

int[] m1 = new int[len1];

for (int i = 0; i < m1.length; i++) {

System.out.println("Введите " + (i + 1) + " элемент массива:");

m1[i] = in.nextInt();

}

System.out.println("Результат: " + decrypt(m1));

}

else {

System.out.println("Возможно выбрать только 1 или 2");

}

}

case 42 -> {

System.out.println("canMove()");

System.out.println("Введите фигуру:");

String a2 = sc.nextLine();

System.out.println("Введите положение фигуры на шахматной доске:");

String b2 = sc.nextLine();

System.out.println("Введите целевую позицию:");

String c2 = sc.nextLine();

System.out.println("Результат: " + canMove(a2, b2, c2));

}

case 43 -> {

System.out.println("canComplete()");

System.out.println("Введите подстроку:");

String a3 = sc.nextLine();

System.out.println("Введите входную строку:");

String b3 = sc.nextLine();

System.out.println("Результат: " + canComplete(a3, b3));

}

case 44 -> {

System.out.println("sumDigProd()");

System.out.println("Сколько будет чисел?");

int a4 = in.nextInt();

int[] m4 = new int[a4];

for (int i = 0; i < a4; i++) {

System.out.println("Введите " + (i+1) + " число: ");

m4[i] = in.nextInt();

}

System.out.println("Результат: " + sumDigProd(m4));

}

case 45 -> {

System.out.println("sameVowelGroup()");

System.out.println("Введите количество элементов массива:");

int len5 = in.nextInt();

String[] m5 = new String[len5];

for (int i = 0; i < m5.length; i++) {

System.out.println("Введите " + (i + 1) + " элемент массива:");

m5[i] = sc.nextLine();

}

System.out.print("Результат: " + sameVowelGroup(m5));

}

case 46 -> {

System.out.println("validateCard()");

System.out.println("Введите число:");

long a6 = in.nextLong();

System.out.println("Результат: " + validateCard(a6));

}

case 47 -> {

System.out.println("numToEng() - 1");

System.out.println("numToRus() - 2");

int a7 = in.nextInt();

System.out.println("Введите число (от 0 до 999):");

int s7 = in.nextInt();

if (a7 == 1) {

System.out.println("Результат: " + numToEng(s7));

}

else {

System.out.println("Результат: " + numToRus(s7));

}

}

case 48 -> {

System.out.println("getSha256Hash()");

System.out.println("Введите строку:");

String a8 = sc.nextLine();

System.out.println("Результат: " + getSha256Hash(a8));

}

case 49 -> {

System.out.println("correctTitle()");

System.out.println("Введите строку:");

String a9 = sc.nextLine();

System.out.println("Результат: " + correctTitle(a9));

}

case 50 -> {

System.out.println("hexLattice()");

System.out.println("Введите число:");

int a10 = in.nextInt();

System.out.println("Результат:");

System.out.println(hexLattice(a10));

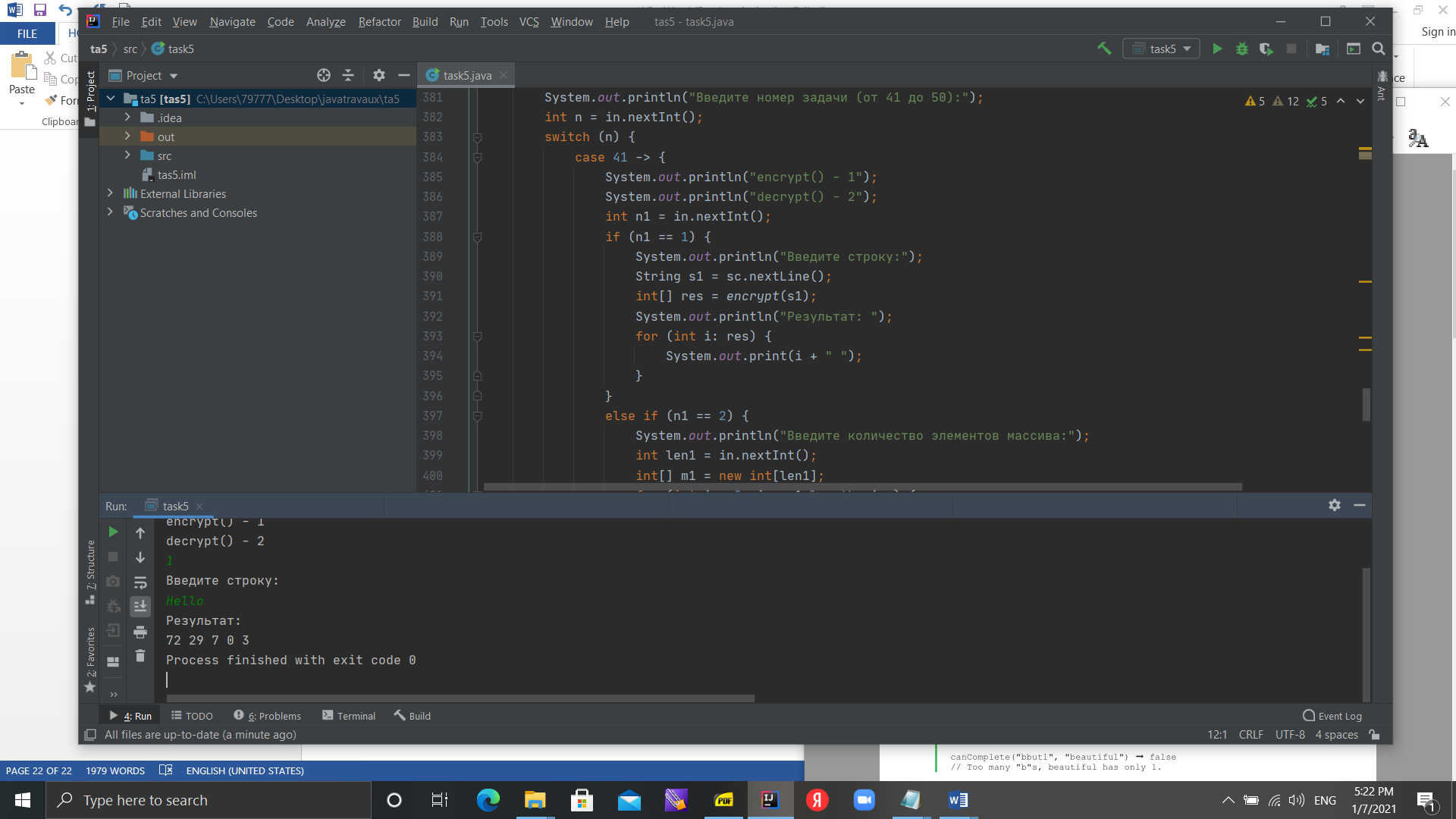
}

}

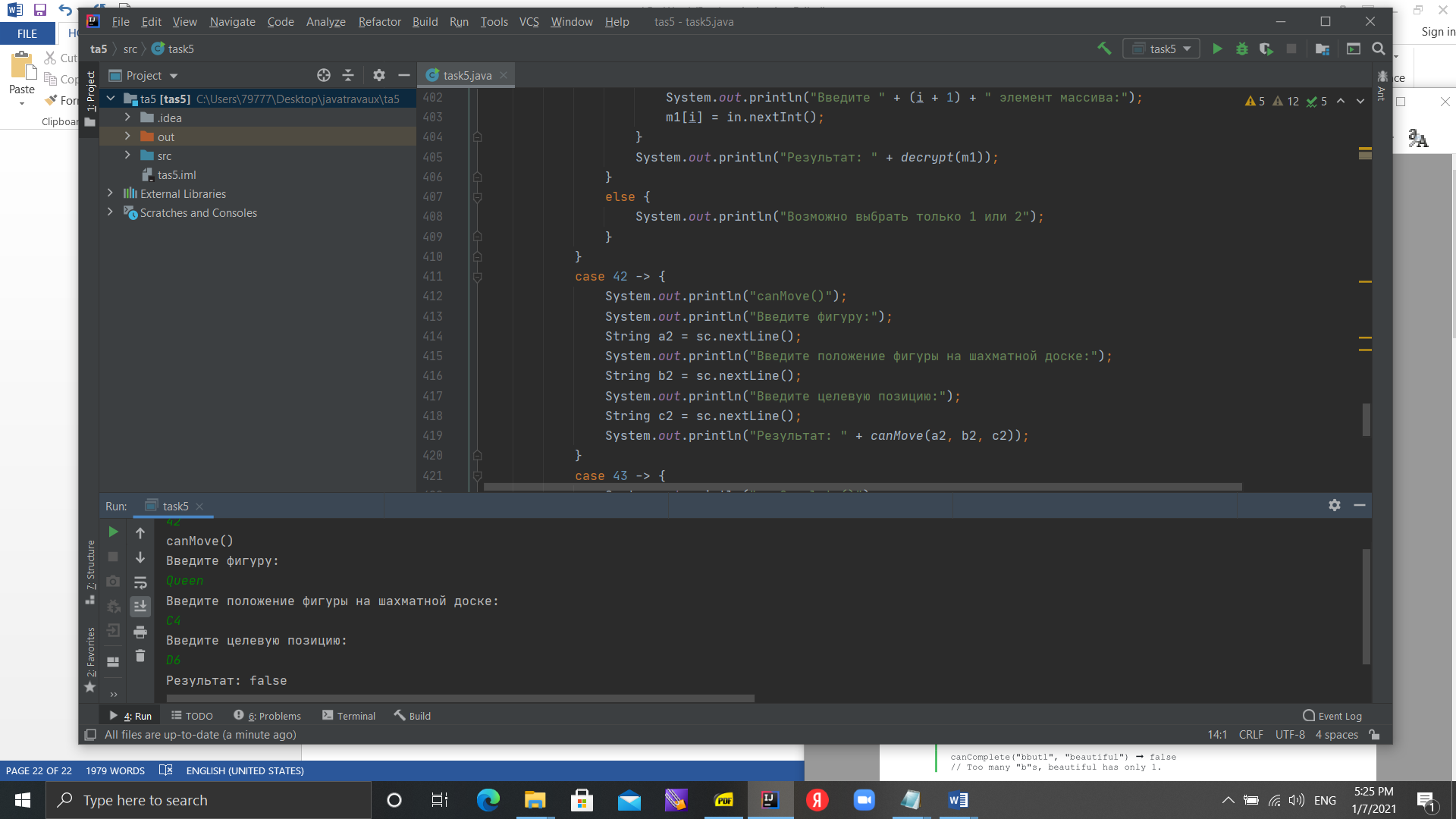
}

}

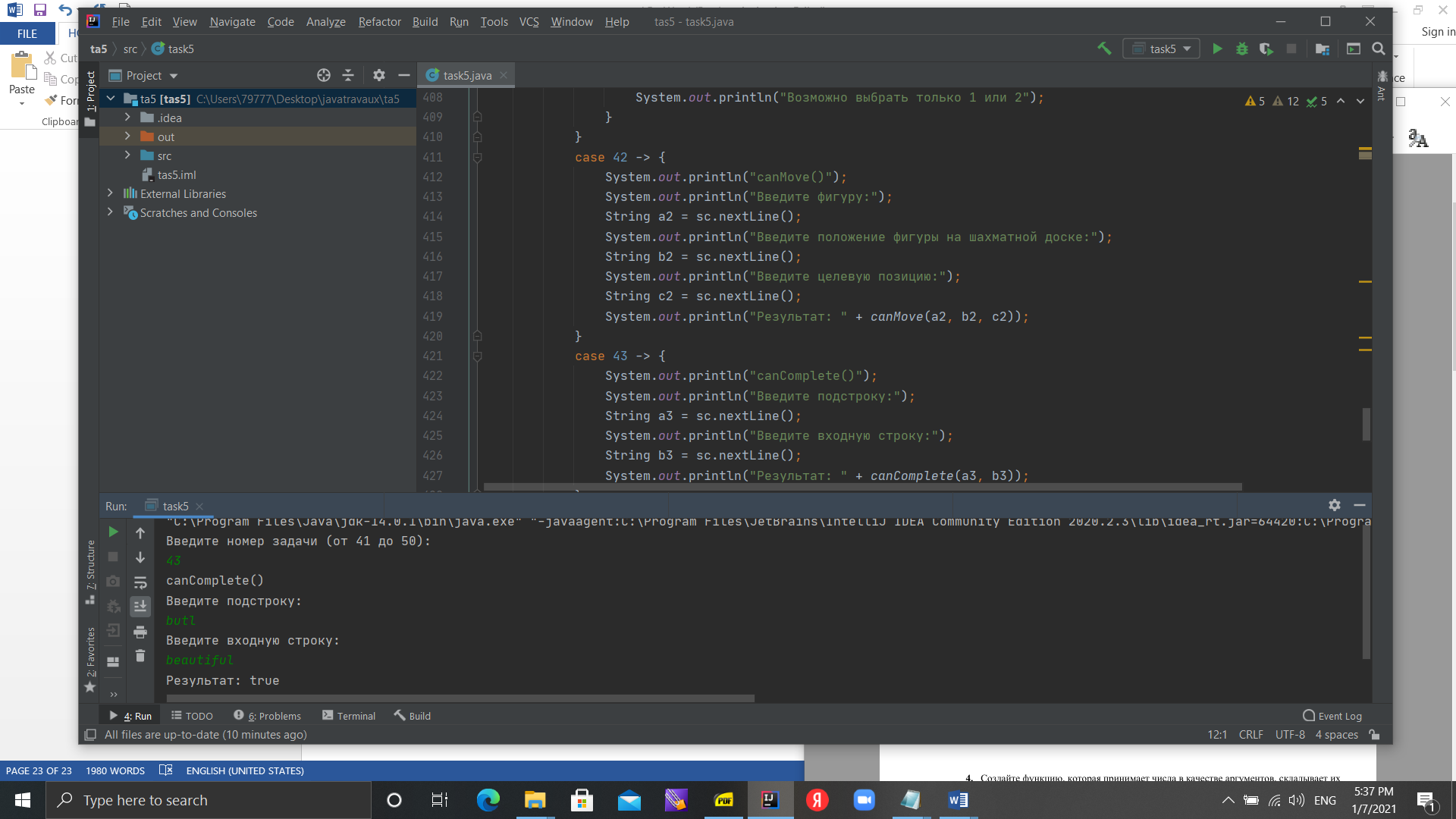
41//5.1



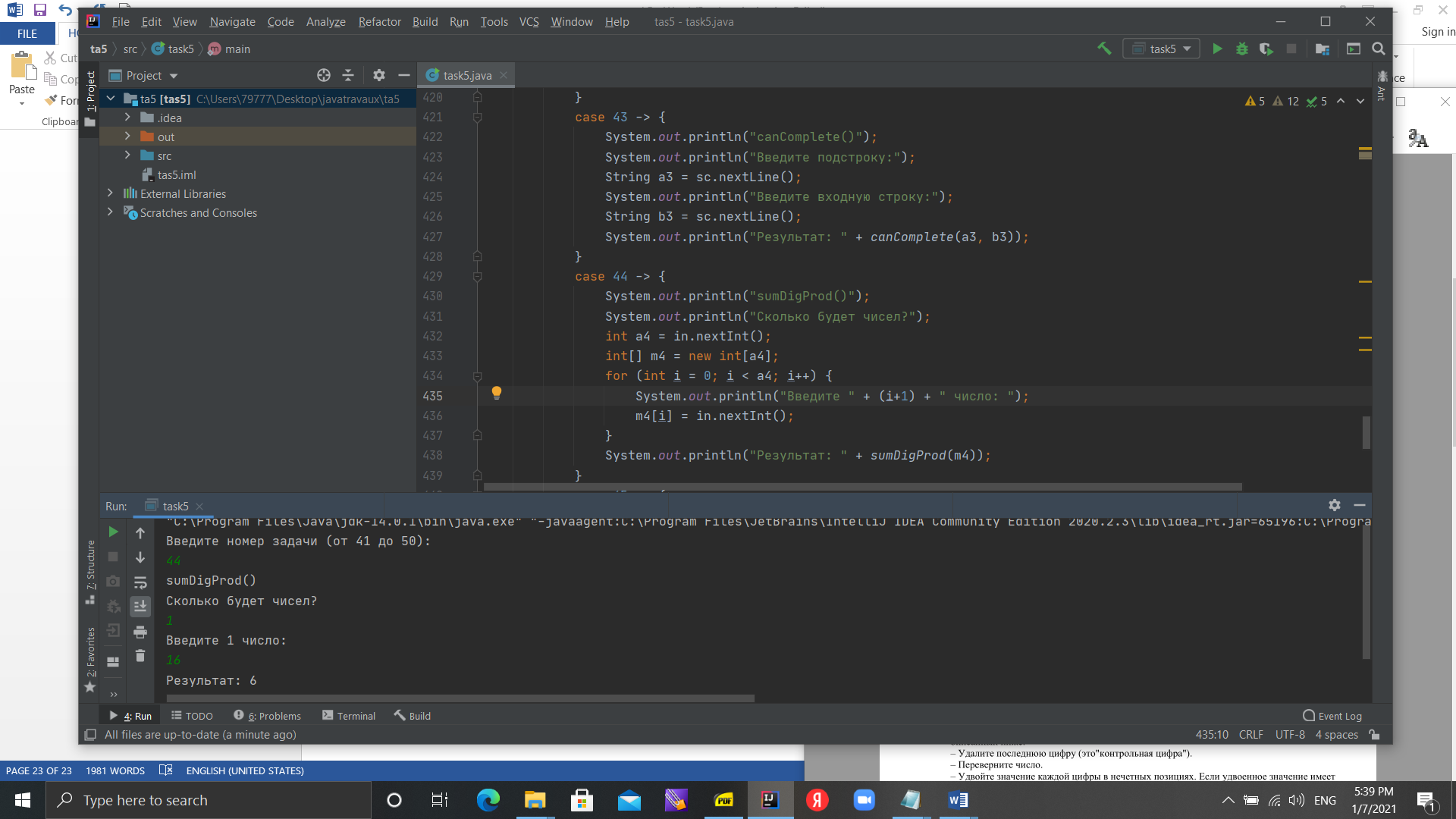
42//5.2



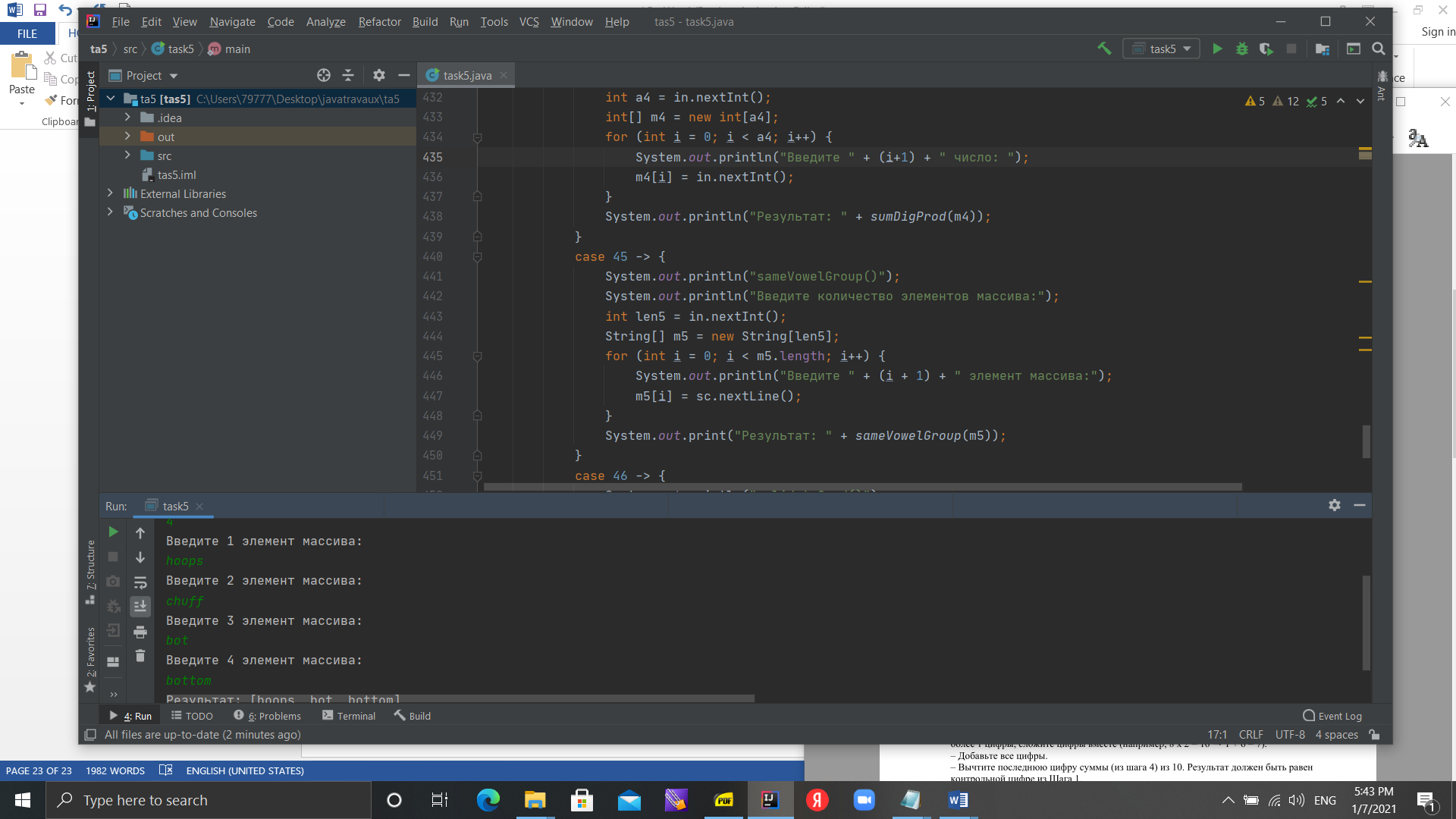
43//5.3

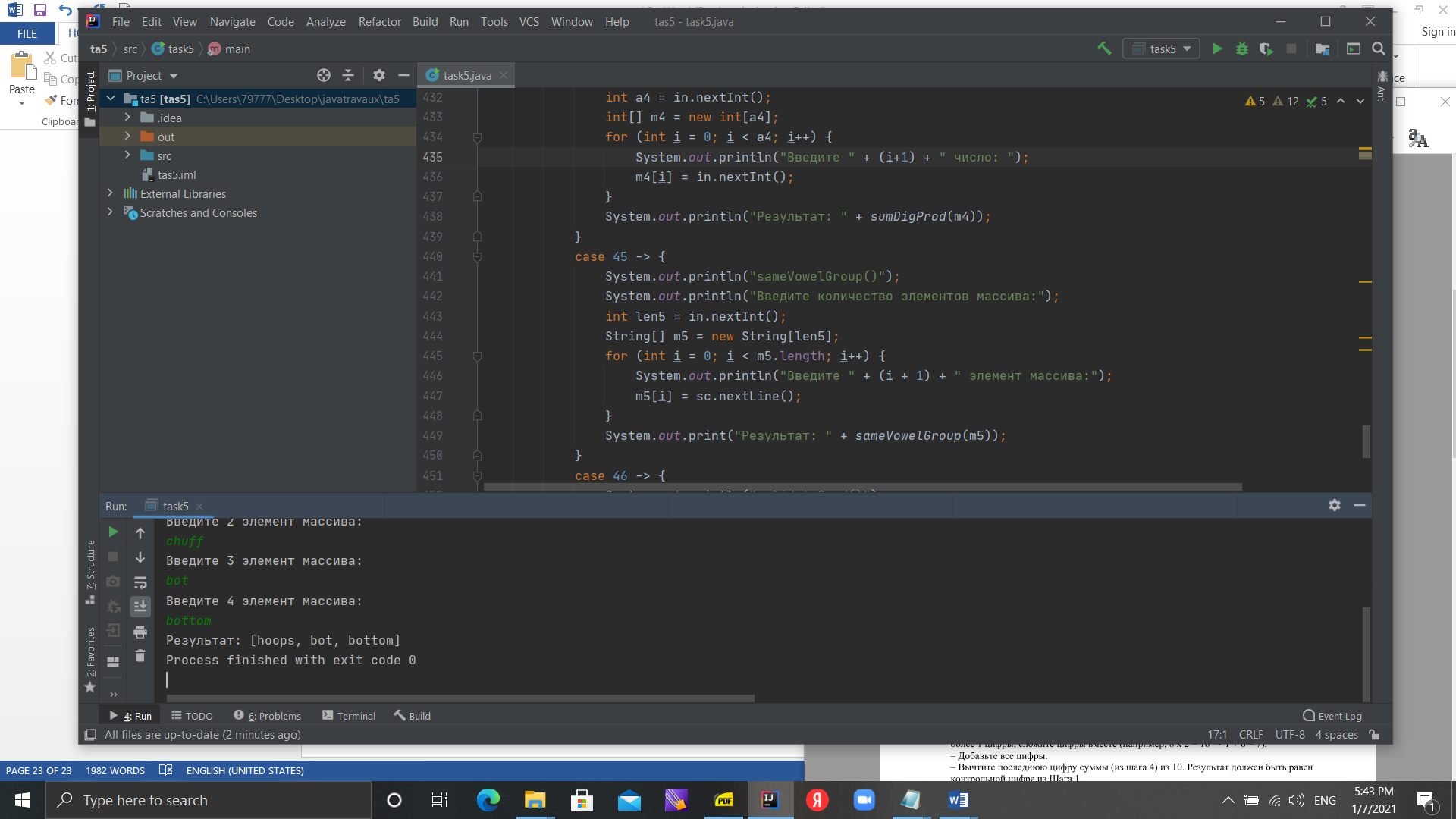


44//5.4

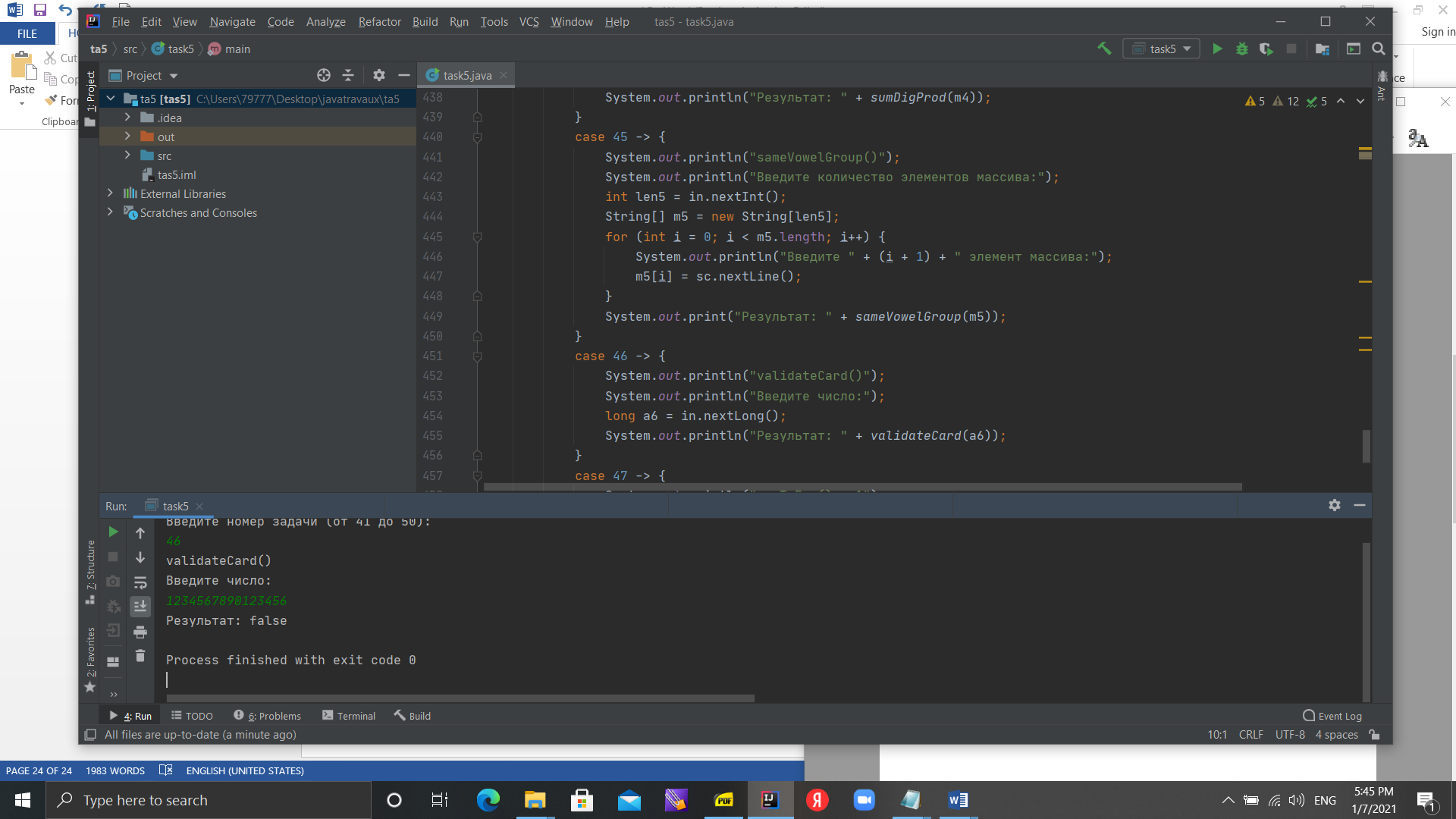


45//5.5

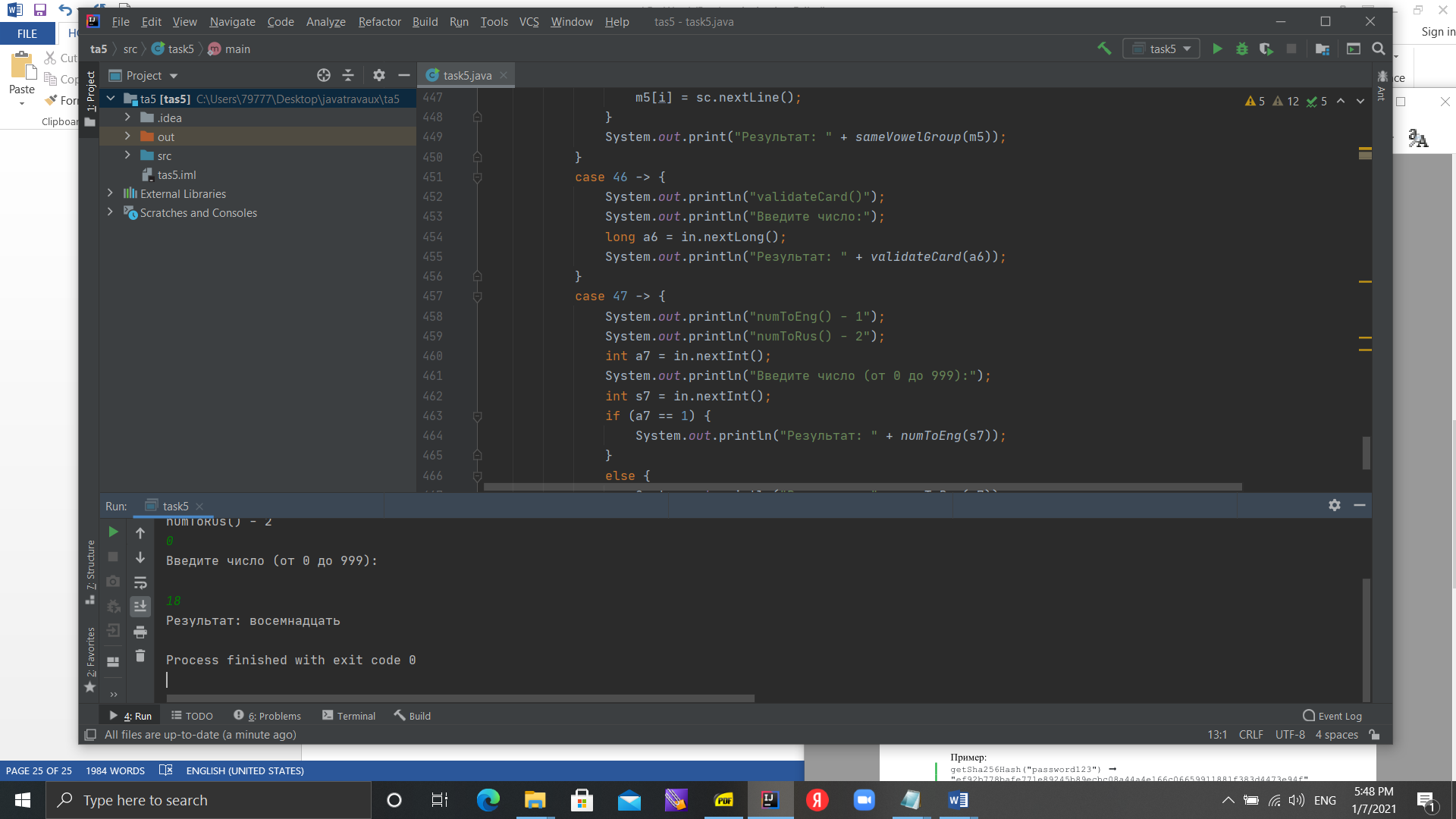




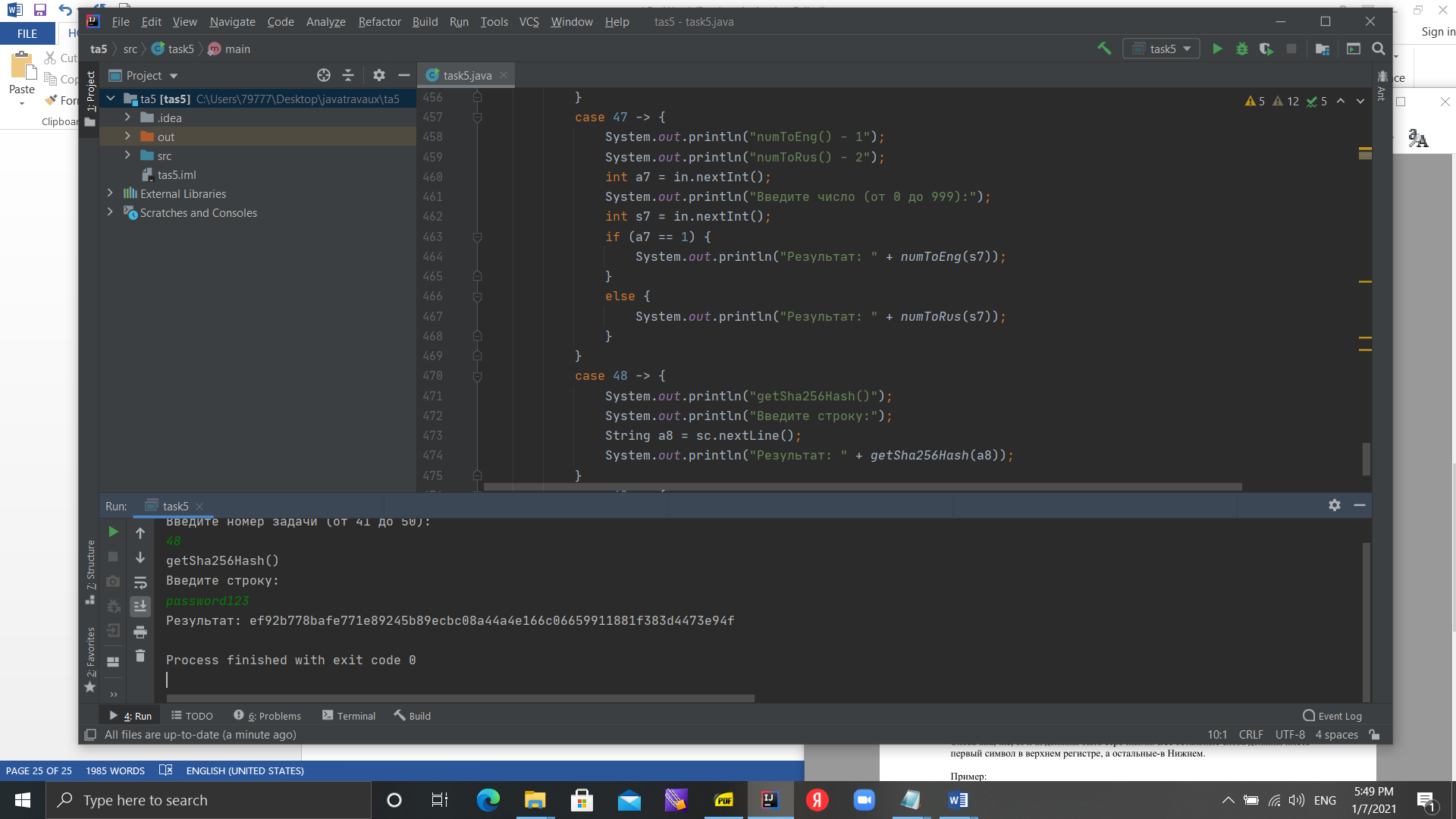
46//5.6



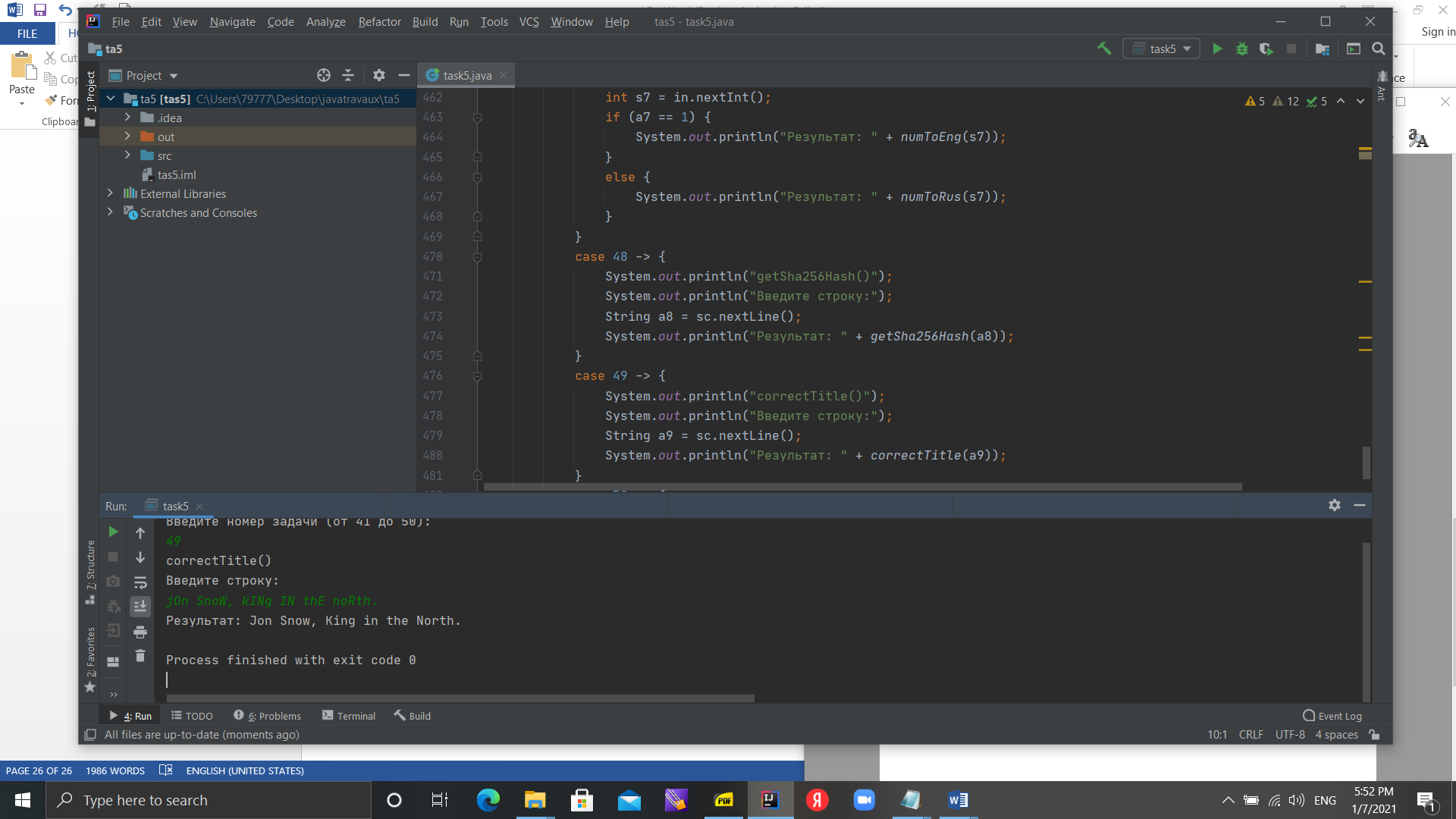
47//5.7



48//5.8



49//5.9



50//5.10

