**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: 6/6

Подготовила студентка

группы БВТ1901: Нкурикийе Х

Проверил: Мосева М.С.

Москва 2021

import java.util.\*;

public class task6 {

static int bell(int a) { // 6.1

int[][] bell = new int[a+1][a+1];

bell[0][0] = 1;

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

bell[i][0] = bell[i-1][i-1];

for (int j=1; j<=i; j++) {

bell[i][j] = bell[i - 1][j - 1] + bell[i][j - 1];

}

}

return bell[a][0];

}

public static String translateWord(String s){ // 6.2

String res = "";

boolean ifLetterUp = false;

if (s.equals(" ") || s.equals("")) {

return res;

}

char l = s.charAt(0);

if (l >= 'A' && l <= 'Z')

ifLetterUp = true;

if ((l == 'a') || (l == 'e') || (l == 'i') || (l == 'o') || (l == 'u') || (l == 'y') ||

(l == 'A') || (l == 'E') || (l == 'I') || (l == 'O') || (l == 'U') || (l == 'Y')) {

res = s + "yay";

}

else {

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

char r = s.charAt(i);

if ((r=='a') || (r=='e') || (r=='i') || (r=='o') || (r=='u') || (r=='y')) {

String end = s.substring(0, i);

res = s.substring(i) + end + "ay";

break;

}

}

}

if (ifLetterUp)

res = Character.toUpperCase(res.charAt(0)) + res.toLowerCase().substring(1, res.length());

return res;

}

public static String translateSentence(String s) {

StringBuilder res = new StringBuilder();

new StringBuilder();

StringBuilder letters;

new StringBuilder();

StringBuilder symbols;

if (s.equals(" ")) {

return res + " ";

}

String[] words = s.split(" ");

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

letters = new StringBuilder();

symbols = new StringBuilder();

for (char l : words[i].toCharArray()) {

if (l >= 'A' && l <= 'z') {

letters.append(l);

}

else {

symbols.append(l);

}

}

if (i == words.length - 1)

res.append(translateWord(letters.toString())).append(symbols);

else

res.append(translateWord(letters.toString())).append(symbols).append(" ");

}

return res.toString();

}

public static boolean validColor(String a) { // 6.3

a = a.toLowerCase();

if (a.contains("rgba")) {

String[] arrOfRGBA = a.substring(5).split("[\\D&&[^.]]");

if (arrOfRGBA.length == 4) {

try {

for (int i = 0; i < 3; i++){

if (!(Integer.parseInt(arrOfRGBA[i]) >= 0 && Integer.parseInt(arrOfRGBA[i]) <= 255))

return false;

}

return Float.parseFloat(arrOfRGBA[3]) >= 0 && Float.parseFloat(arrOfRGBA[3]) <= 1;

} catch (Exception e) {

return false;

}

}

else return false;

}

else if (a.contains("rgb")) {

String[] arrOfRGBA = a.substring(4).split("[\\D&&[^.]]");

if (arrOfRGBA.length == 3) {

try{

for (int i = 0; i < 3; i++) {

if (!(Integer.parseInt(arrOfRGBA[i]) >= 0 && Integer.parseInt(arrOfRGBA[i]) <= 255))

return false;

}

return true;

}

catch (Exception e){

return false;

}

}

else return false;

}

else return false;

}

public static String stripUrlParams(String url, String ... argsToDell) { // 6.4

String[] args = url.substring(url.indexOf("?") + 1).split("&");

StringBuilder finalArgs = new StringBuilder();

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

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

if (args[i].charAt(0) == args[j].charAt(0)) {

args[i] = " ";

break;

}

}

for (String s : argsToDell) {

if (args[i].charAt(0) == s.charAt(0)) {

args[i] = " ";

break;

}

}

}

Arrays.sort(args);

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

if (!args[i].equals(" "))

if (i != args.length - 1)

finalArgs.append(args[i]).append("&");

else

finalArgs.append(args[i]);

}

return url.substring(0, url.indexOf("?")+1) + finalArgs;

}

public static String[] getHashTags(String a) { // 6.5

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

int highLength = wordsArr[0].length();

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

if (wordsArr[i].length() > highLength)

highLength = wordsArr[i].length();

}

int tagsSize = 3;

if (wordsArr.length < 3)

tagsSize = wordsArr.length;

String[] tagsArr = new String[tagsSize];

int tagsLeft = tagsSize;

int tagsPosCounter = 0;

for (int i = 0; (tagsLeft > 0)&&(highLength > 0); i++) {

if (wordsArr[i].length() == highLength) {

tagsArr[tagsPosCounter] = "#" + wordsArr[i];

tagsLeft--;

tagsPosCounter++;

}

if (i == wordsArr.length-1){

i = -1;

highLength--;

}

}

return tagsArr;

}

public static int ulam(int a) { // 6.6

int[] ulamPeriod = new int[a];

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

switch (i) {

case 0 -> ulamPeriod[i] = 1;

case 1 -> ulamPeriod[i] = 2;

default -> {

int waysOfSolve = 0;

int rightNumber = ulamPeriod[i - 1] + 1;

while (waysOfSolve != 2) {

waysOfSolve = 0;

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

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

if ((ulamPeriod[j] != ulamPeriod[k]) && (ulamPeriod[j] + ulamPeriod[k] == rightNumber))

waysOfSolve++;

}

}

if (waysOfSolve != 2)

rightNumber++;

else

ulamPeriod[i] = rightNumber;

}

}

}

}

return ulamPeriod[a-1];

}

public static String longestNonRepeatingSubstring(String str) { // 6.7

Map<Character, Integer> visitedChars = new HashMap<>();

String output = "";

for (int start = 0, end = 0; end < str.length(); end++) {

char currChar = str.charAt(end);

if (visitedChars.containsKey(currChar))

start = Math.max(visitedChars.get(currChar)+1, start);

if (output.length() < end - start + 1)

output = str.substring(start, end + 1);

visitedChars.put(currChar, end);

}

return output;

}

public static String convertToRoman(int a) { // 6.8

StringBuilder ans = new StringBuilder();

if (a / 1000 != 0) {

for (int i = a; i / 1000 != 0; i -= 1000) {

ans.append("M");

}

a %= 1000;

}

if (a / 100 != 0) {

int hundred = a / 100;

if (hundred <= 3)

ans.append("C".repeat(Math.max(0, hundred)));

else if (hundred == 4)

ans.append("CD");

else if (hundred <= 8) {

ans.append("D");

ans.append("C".repeat(hundred - 5));

} else if (hundred == 9)

ans.append("CM");

a %= 100;

}

if (a / 10 != 0) {

int ten = a / 10;

if (ten <= 3)

ans.append("X".repeat(Math.max(0, ten)));

else if (ten == 4)

ans.append("XL");

else if (ten <= 8) {

ans.append("L");

ans.append("X".repeat(ten - 5));

} else if (ten == 9)

ans.append("XC");

a %= 10;

}

if (a % 10 != 0) {

int num = a % 10;

if (num <= 3)

ans.append("I".repeat(Math.max(0, num)));

else if (num == 4)

ans.append("IV");

else if (num <= 8) {

ans.append("V");

ans.append("I".repeat(num - 5));

} else ans.append("IX");

}

return ans.toString();

}

public static boolean formula(String a) { // 6.9

boolean res = false;

int equalsPos = a.indexOf("=");

if ((equalsPos > -1) && (a.lastIndexOf("=") == equalsPos)) {

int mathAns = Integer.parseInt(a.substring(equalsPos+1).trim());

String mathExpress = a.substring(0, equalsPos);

if ((a.contains("+")) && (a.indexOf("+") < equalsPos)) {

String[] mathVars = mathExpress.trim().split(" \\+ ");

if (Integer.parseInt(mathVars[0]) + Integer.parseInt(mathVars[1]) == mathAns)

res = true;

}

else if ((a.contains("\*")) && (a.indexOf("\*") < equalsPos)) {

String[] mathVars = mathExpress.trim().split(" \\\* ");

if (Integer.parseInt(mathVars[0]) \* Integer.parseInt(mathVars[1]) == mathAns)

res = true;

}

else if ((a.contains("/")) && (a.indexOf("/") < equalsPos)) {

String[] mathVars = mathExpress.trim().split(" \\/ ");

if (Integer.parseInt(mathVars[0]) / Integer.parseInt(mathVars[1]) == mathAns)

res = true;

}

else if ((a.contains("-")) && (a.indexOf("-") < equalsPos)) {

String[] mathVars = mathExpress.trim().split(" \\- ");

if (Integer.parseInt(mathVars[0]) - Integer.parseInt(mathVars[1]) == mathAns)

res = true;

}

}

return res;

}

public static boolean palindromeDescendant(int a) { // 6.10

boolean res = false;

int aLength = Integer.toString(a).length();

String[] arrayOfNumbers = Integer.toString(a).split("");

StringBuilder workNum = new StringBuilder(Integer.toString(a));

while ((aLength > 1) && (Integer.parseInt(workNum.toString()) != numReverse(Integer.parseInt(workNum.toString())))) {

workNum = new StringBuilder();

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

if (i % 2 != 0)

workNum.append(Integer.parseInt(arrayOfNumbers[i - 1]) + Integer.parseInt(arrayOfNumbers[i]));

}

aLength = workNum.length();

arrayOfNumbers = workNum.toString().split("");

}

if ((Integer.parseInt(workNum.toString()) == numReverse(Integer.parseInt(workNum.toString()))) && (aLength > 1))

res = true;

return res;

}

public static int numReverse(int a) {

int res = 0;

while (a != 0) {

int digit = a % 10;

res = res \* 10 + digit;

a /= 10;

}

return res;

}

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 51 до 60):");

int n = in.nextInt();

switch (n) {

case 51 -> {

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

System.out.println("Введите число:");

int a1 = in.nextInt();

System.out.println("Результат: " + bell(a1));

}

case 52 -> {

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

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

int n2 = in.nextInt();

if (n2 == 1) {

System.out.println("Введите слово:");

String a2 = sc.nextLine();

System.out.println("Результат: " + translateWord(a2));

}

else {

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

String b2 = sc.nextLine();

System.out.println("Результат: " + translateSentence(b2));

}

}

case 53 -> {

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

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

String a3 = sc.nextLine();

System.out.println("Результат: " + validColor(a3));

}

case 54 -> {

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

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

String a4 = sc.nextLine();

System.out.println("Нужен ли второй аргумент? 1 - если нужен");

int b4 = in.nextInt();

if (b4 == 1) {

System.out.println("Введите второй аргумент:");

String c4 = sc.nextLine();

System.out.println("Результат: " + stripUrlParams(a4, c4));

}

else {

System.out.println("Результат: " + stripUrlParams(a4));

}

}

case 55 -> {

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

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

String a5 = sc.nextLine();

String[] res5 = getHashTags(a5);

System.out.println("Результат: ");

for (String i: res5) {

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

}

}

case 56 -> {

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

System.out.println("Введите число:");

int a6 = in.nextInt();

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

}

case 57 -> {

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

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

String a7 = sc.nextLine();

System.out.println("Результат: " + longestNonRepeatingSubstring(a7));

}

case 58 -> {

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

System.out.println("Введите число (от 0 до 3999):");

int a8 = in.nextInt();

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

}

case 59 -> {

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

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

String a9 = sc.nextLine();

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

}

case 60 -> {

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

System.out.println("Введите число:");

int a10 = in.nextInt();

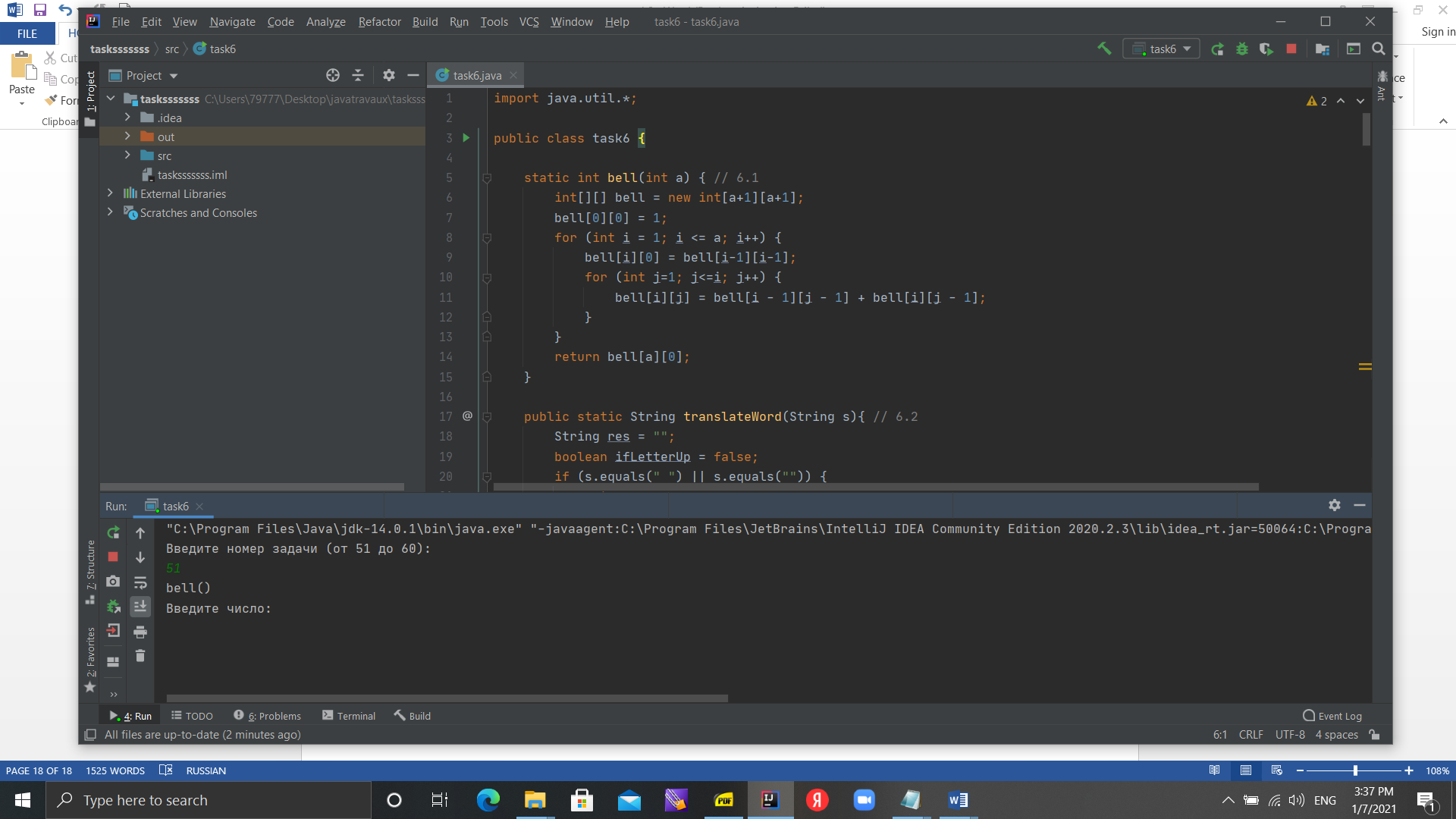
System.out.println("Результат: " + palindromeDescendant(a10));

}

}

}

}



52//6.2

case 52 -> {

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

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

int n2 = in.nextInt();

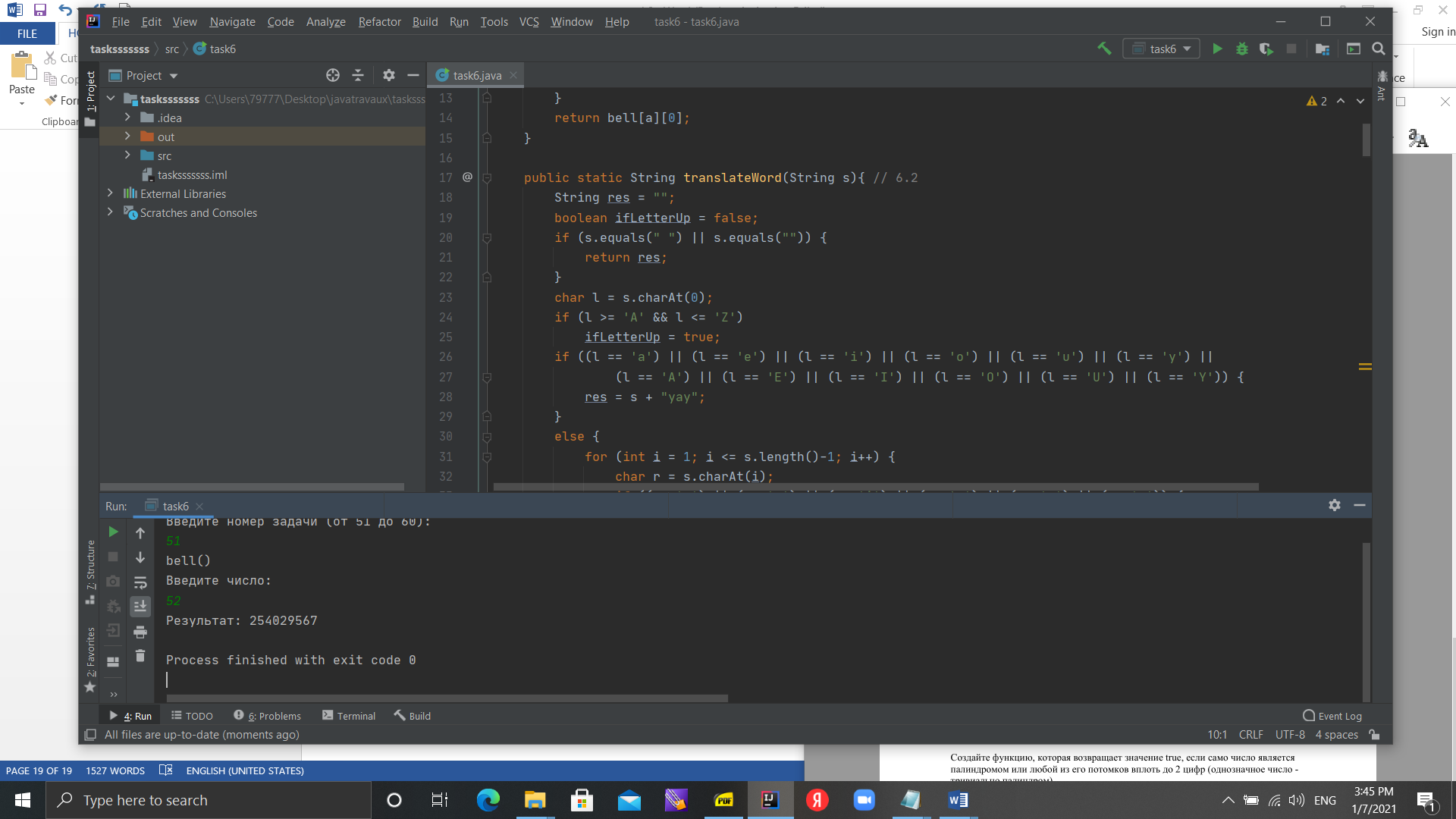
if (n2 == 1) {

System.out.println("Введите слово:");

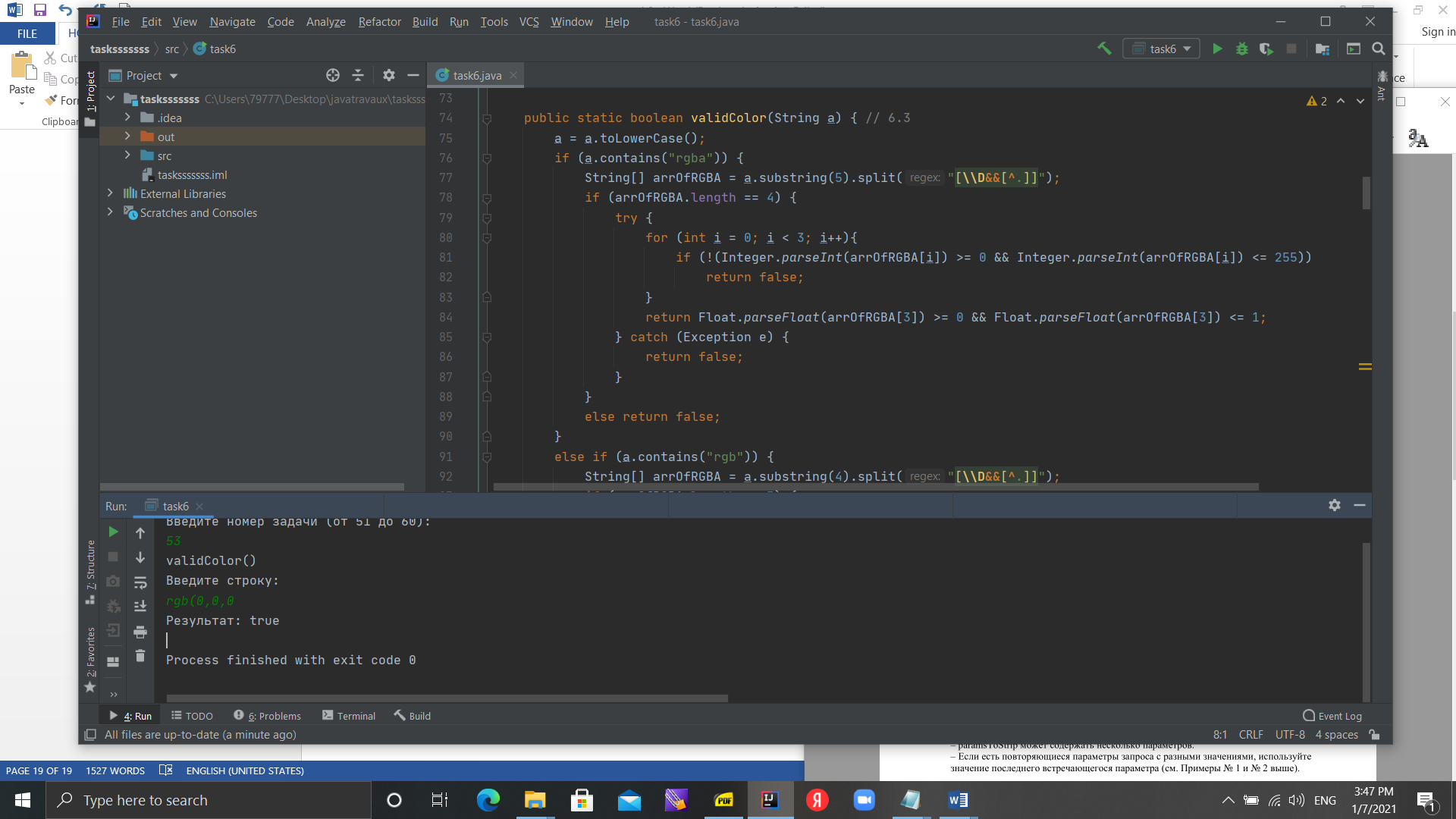
String a2 = sc.nextLine();

System.out.println("Результат: " + translateWord(a2));

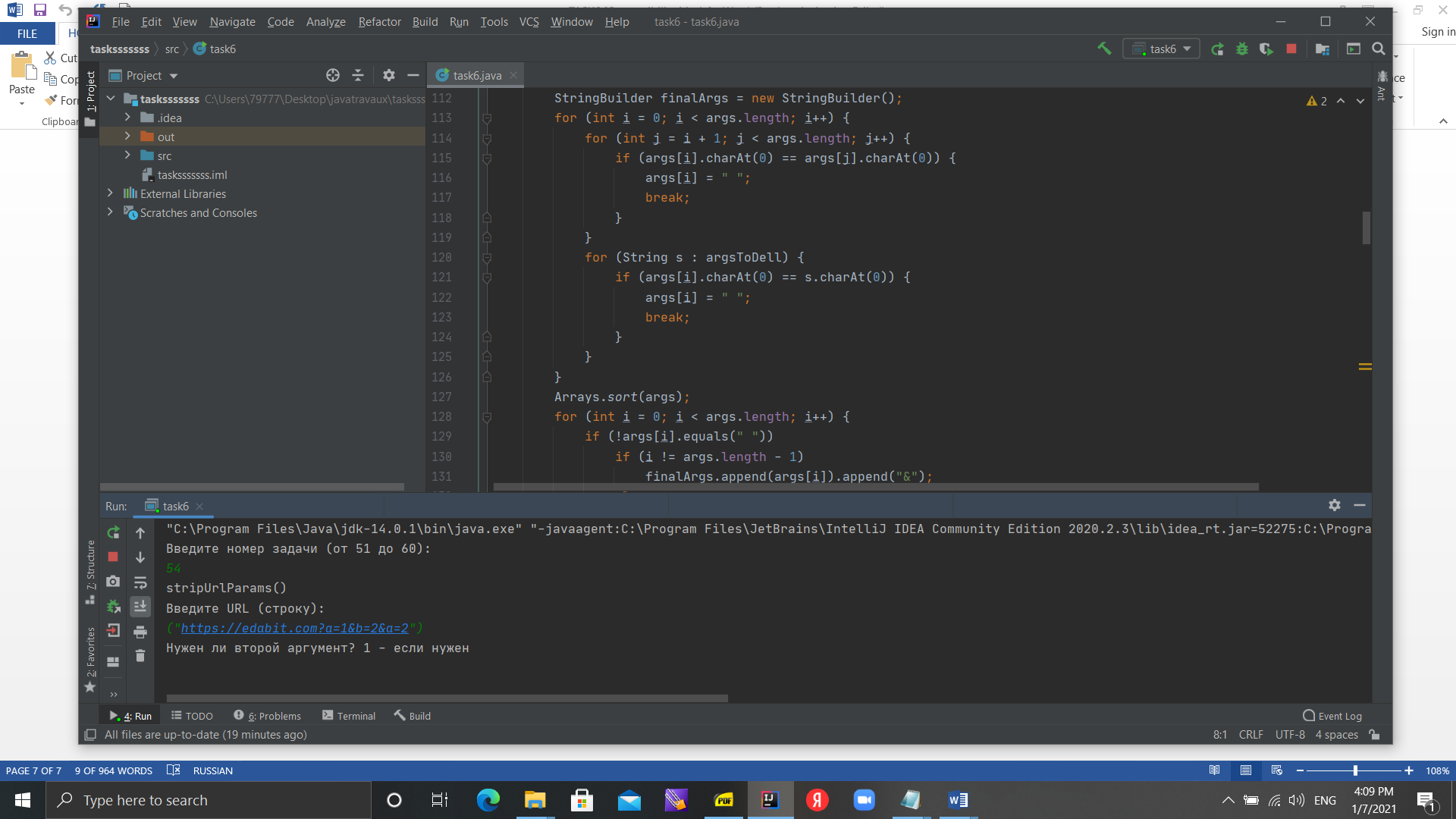
}



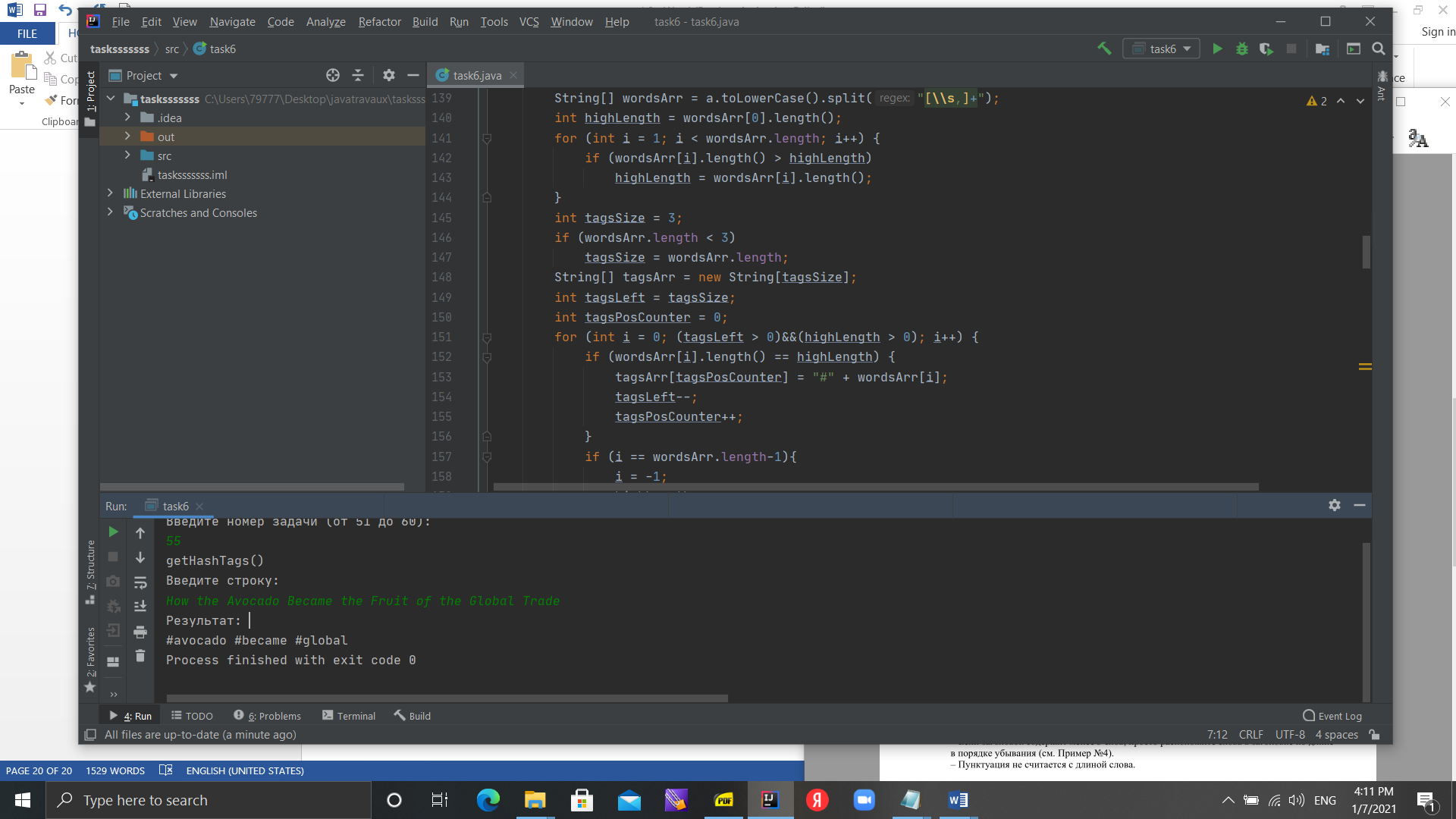
6.3//53



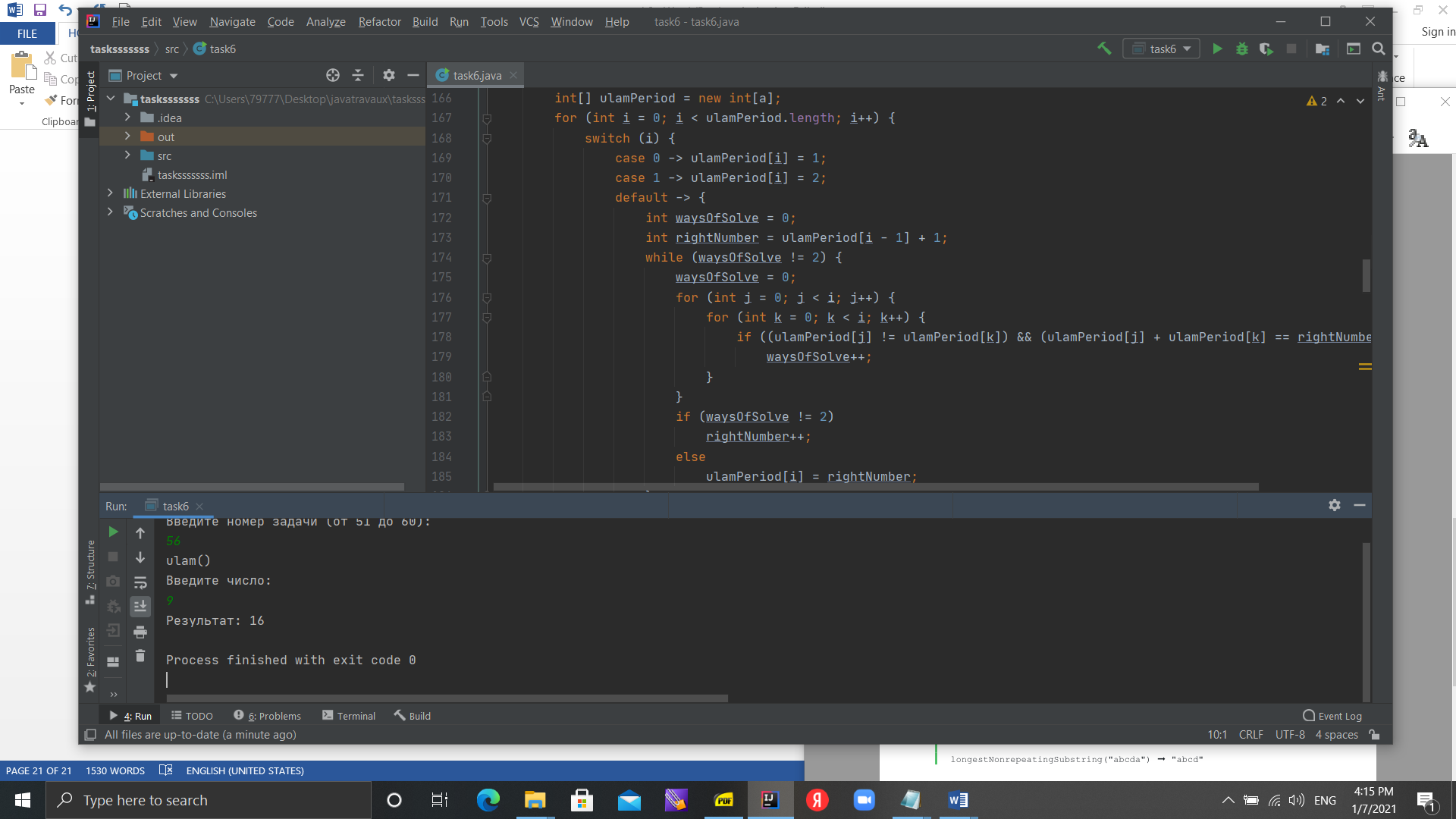
6.4//54



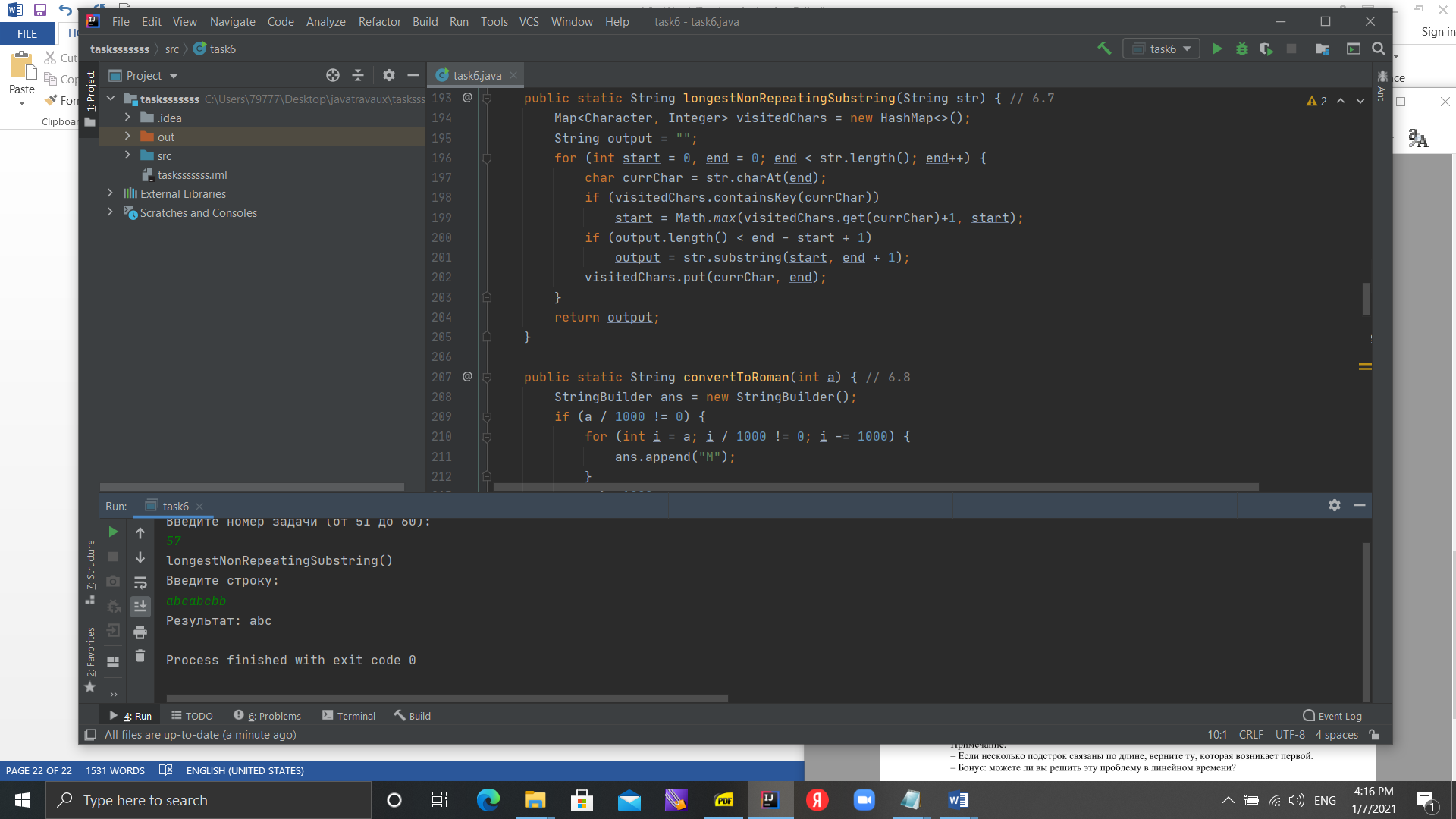
6.5//55



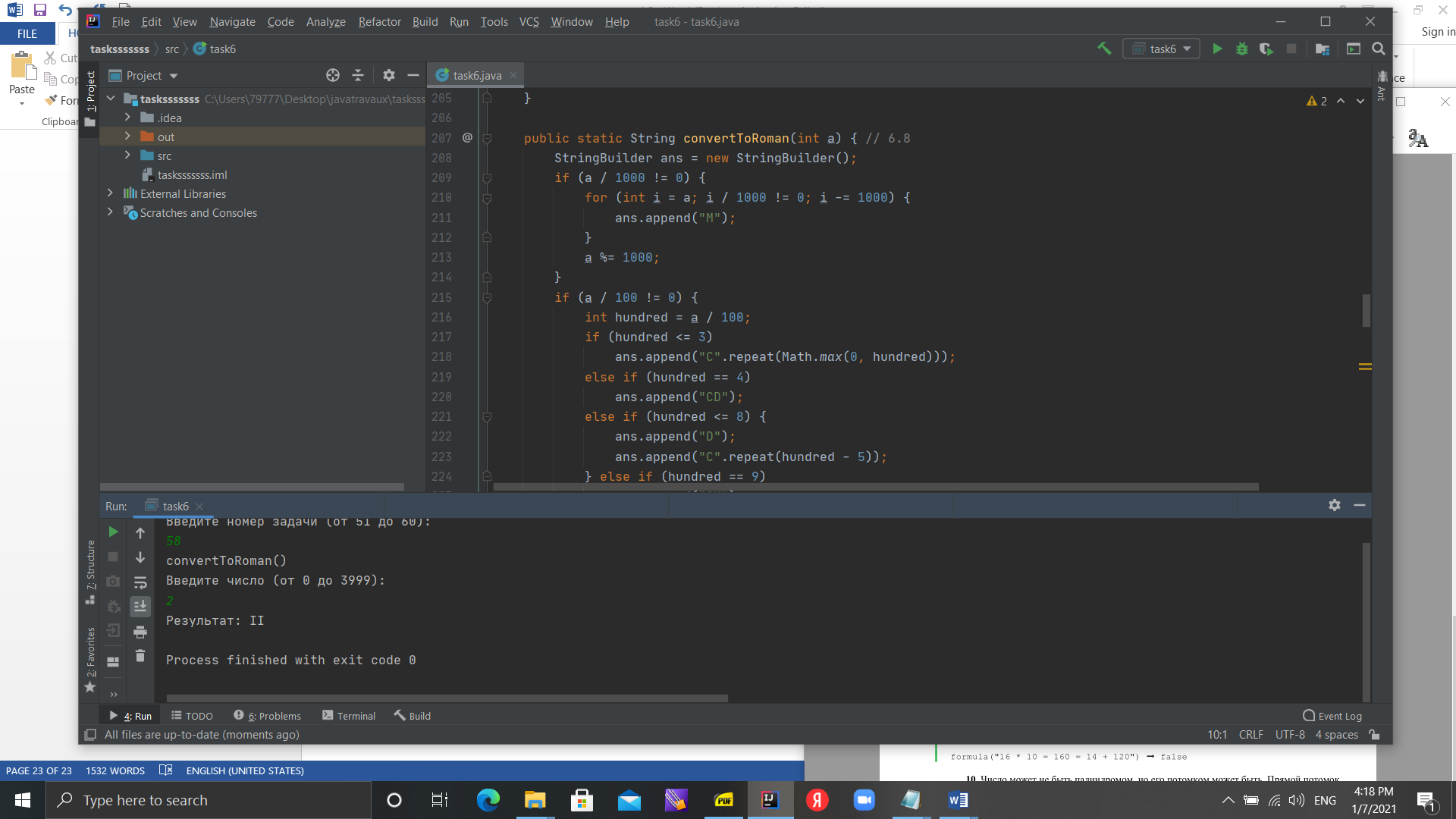
6.6//56



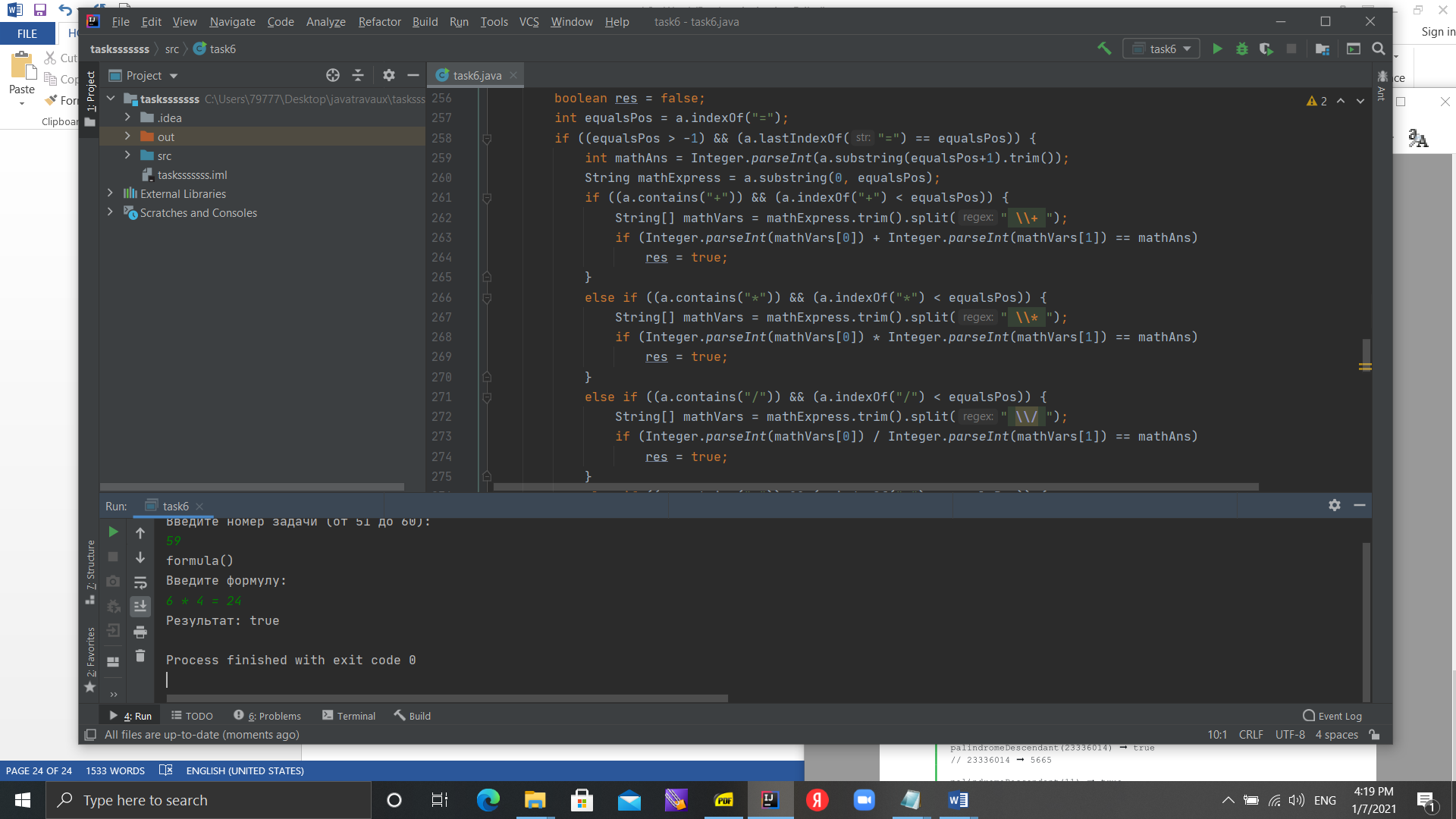
6.7//57



6.8//58



6.9//59



6.10//60

