public class MyClass {

public static void main(String args[]) {

int[] show = problems.hard("A");

for (int r:show){

System.out.printf("%d",r);

System.out.println("");

}

}

}

class problems{

public static boolean contains(char[] arr, char item) { //Check if an item is existed in an array

for (char n : arr) {

if (item == n) {

return true;

}

}

return false;

}

public static int max(int[] t) {

int maximum = t[0]; // Find a maximum number in an array

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

if (t[i] > maximum) {

maximum = t[i];

}

}

return maximum;

}//https://perso.ensta-paris.fr/~diam/java/online/notes-java/data/arrays/arrays-ex-max.html

public static int[] hard(String s){

char[] caps = {'A','B','C','D','E','F','G','H','I','J','K','L','M','N','O','P','Q','R','S','T','U','V','W','X','Y','Z'};

int upper = 0;

int lower = 0;

int[] countU = new int[s.length()];//use easy Method to count no of Uppercase letters

int[] countL = new int[s.length()];//of Lowercase letters instead of using s.length()

int idxU = -1;

int idxL = -1;

int[] result = new int[2];

if (s.length()==1){ // In case the string only contains 1 element.

char c = s.charAt(0);

if (contains(caps,c)){

countU[0] = 1;

}

if (!contains(caps,c)){

countL[0] = 1;

}

}

for (int i=0; i<s.length()-1; i++ ){ // Main algorithm for counting no of Uppercase letters in seperate sequences

char c = s.charAt(i);

char nextc = s.charAt(i+1);

if (contains(caps,c)){

upper = upper + 1;

if (!contains(caps,nextc)){

idxU = idxU + 1;

countU[idxU] = upper;

upper = 0;

}

if (contains(caps,nextc) && i == s.length()-2){ //Ex \_\_\_\_AA

idxU = idxU + 1;

upper = upper + 1;

countU[idxU] = upper;

}

}

if (!contains(caps,c) && contains(caps,nextc) && i == s.length()-2){ //Ex \_\_\_\_aA

idxU = idxU + 1;

upper = upper + 1;

countU[idxU] = upper;

}

}

for (int i=0; i<s.length()-1; i++ ){ // Counting no of Lowercase letters in seperate sequences similarly

char c = s.charAt(i);

char nextc = s.charAt(i+1);

if (!contains(caps,c)){

lower = lower + 1;

if (contains(caps,nextc)){

idxL = idxL + 1;

countL[idxL] = lower;

lower = 0;

}

if (!contains(caps,nextc) && i == s.length()-2){

idxL = idxL + 1;

lower = lower + 1;

countL[idxL] = lower;

}

}

if (contains(caps,c) && !contains(caps,nextc) && i == s.length()-2){

idxL = idxL + 1;

lower = lower + 1;

countL[idxL] = lower;

}

}

//Final Result

result[0] = max(countL);

result[1] = max(countU);

return result;

}

}