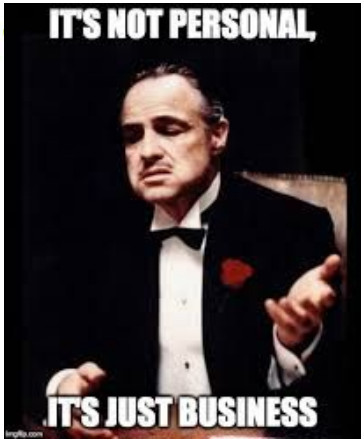
Separate data from business logic.

```java
public static String pravidla(String dvojica){
    switch (dvojica){
        case "AA": return "B"
        case "AB": return "A"
        case "AC": return "A"
        case "BA": return "C";
        case "BB": return "A";
        case "BC": return "C";
        case "CA": return "AA";
        case "CB": return "A";
        case "CC": return "BB";
        default: return "";
    }
```

```java
String res = "";
for (int i = 0; i < s.length()-1; i++){
    if(s.substring(i,i+2).matches("AA")
        res += "B";
    }
    if(s.substring(i,i+2).matches("AB")
        res += "A";
    }
    if(s.substring(i,i+2).matches("AC")){
        res += "A";
    }
    if(s.substring(i,i+2).matches("BA")){
        res += "C";
    }
    if(s.substring(i,i+2).matches("BB")){
        res += "A";
    }
    if(s.substring(i,i+2).matches("BC")){
        res += "C";
    }
    if(s.substring(i,i+2).matches(
        res += "AA";
    }
```

```java
}

ɜramatika {
atic  String[] INPUTS = {"AA", "AB", "AC", "BA", "BB", "BC", "CA", "CB", "CC"}
atic  String[] OUTPUTS = {"B", "A", "A", "C", "A", "C", "AA", "A", "BB"};
```



Short functions are the key.

```
public static int alternate(String str) {
    String znaky = unique(str);
    int maxDlzka = 0;

    for (int i = 0; i < znaky.length(); i++) {
        for (int j = i + 1; j < znaky.length(); j++) {
            char a = znaky.charAt(i);
            char b = znaky.charAt(j);
            maxDlzka = Math.max(maxDlzka, filterAndValidate(str, a, b));
        }
    }

    return maxDlzka;
}
```

Checking documentation is important.

```
static boolean contains(StringBuilder s, char c) {
    for(int i = 0; i < s.length(); ++i){
        if(s.charAt(i) == c){
            return true;
        }
    }
    return false;
}
```

s.IndexOf(c)>=0

Thinking outside the box.
Because too many ifs are ify.

```java
public class Pole3D {
    public static boolean equalsIgnoreCase(String[][][] a, String[][][] b) {
        return normalize(a).equals(normalize(b));
    }

    public static String normalize(String[][][] arr) {
        if (arr == null) return "null";
        StringBuilder sb = new StringBuilder();
        for (String[][] matica : arr) {
            if (matica == null) {
                sb.append("null;");
                continue;
            }
            for (String[] r : matica) {
                if (r == null) {
                    sb.append("null;");
                    continue;
                }
                for (String elem : r) {
                    sb.append(elem == null ? "null" : elem.toLowerCase()).append(";");
                }
            }
        }
        return sb.toString();
```

Call once use everywhere.

StringBuffer is for highly mutable seqeuences not “XY”.

```
for (int i = 0; i < s.length() - 1; i++) {

    StringBuffer sb = new StringBuffer();
    sb.append(s.charAt(i));
    sb.append(s.charAt(i + 1));
    if (sb.toString().equals("AA")) {
        result.append("B");
    } else if (sb.toString().equals("AB")
            || sb.toString().equals("AC")
            || sb.toString().equals("BB")
            || sb.toString().equals("CB")) {
        result.append("A");
```

This is highly mutable StringBuilder is preferable.

```java
public static String change(String s){
    String toRet = "";
    String buff = "";

    for (int i=0; i<s.length(); i++) {
        buff += s.charAt(i);

        if (i != 0) {
            if (buff.equals("AA")) toRet += "B";
            else if (buff.equals("AB")) toRet += "A";
            else if (buff.equals("AC")) toRet += "A";
            else if (buff.equals("BA")) toRet += "C";
            else if (buff.equals("BB")) toRet += "A";
            else if (buff.equals("BC")) toRet += "C";
            else if (buff.equals("CA")) toRet += "AA";
            else if (buff.equals("CB")) toRet += "A";
            else if (buff.equals("CC")) toRet += "BB";
            buff = buff.charAt(1) + "";
        }
    }
```

Ctrl+Alt+L = Format Code

Please use extensively.

1 project per cvičenie -> 😊

Výber súboru v riešení:

Alternating.java

Alternating.java
Gauss.java
Gramatika.java
J17.java
Podobne.java
Pole3D.java
Sedemciferne.java
Zaujimave.java