

### **PROBLEMS**

# Keymaster

A1: Easy

10 pt

A2: Medium

20 pt

50 pt

A3: Hard

Dots and Dashes

B1: Easy

10 pt

**B2: Medium** 

15 pt

B3: Hard

40 pt

## **CI** Pipelines

C1: Easy

5 pt

C2: Medium

15 pt

C3: Hard

50 pt

### **Data Centers**

D1: Easy

10 pt

D2: Medium

20 pt

D3: Hard

65 pt

# **Commercial Operations**

E1: Easy

20 pt

E2: Medium

40 pt

E3: Hard

50 pt

### Auto-correct

F1: Easy

5 pt

# **Problem B2: Dots and Dashes** - Medium

15 points

Accepted

**Problem** 

My Submissions

Dots and Dashes is a text encoding, similar to (but not the same as!) other formats used in electric telegraphy. Like those other formats, it uses sequences of dots "." and dashes "-" to encode characters, but unlike those other formats, it uses the following translation table:

+		
A .   B   C	D	
E -   F	J	-
K   L     M   N   O     Q   R	P	
	V	
Y	,	
'   "   ;     (   )	:	
[   ]   {     0   1	}	
2   3   4     6   7	5	-
8   9   +	-	
%		
+		

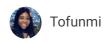
As an optional aid, a copy of the translation table can be found here, with each translation on its own line. i.e.

Α.

В ...

# FACEBOOK Coding Competitions

# FB Hack > 2021 > EMEA Coding Challenge 2021



Keymaster Keymaster		
A1: Easy	10 pt	
A2: Medium		
AZ: Medium	20 pt	
A3: Hard	50 pt	
Dots and Dashes		
B1: Easy	10 pt	
B2: Medium	15 pt	
B3: Hard	40 pt	
CI Pipelines		
C1: Easy	5 pt	
C2: Medium	15 pt	
C3: Hard	50 pt	
Data Centers		
D1: Easy	10 pt	
D2: Medium	20 pt	
D3: Hard	65 pt	
Commercial Operations		
E1: Easy	20 pt	
E2: Medium	40 pt	
E3: Hard	50 pt	
Auto-correct		
F1: Easy	5 pt	

describing open source projects using the *dots* and dashes protocol. Your task is to **decode these descriptions**.

Unfortunately, your state of the art telegraph machine broke, and to your surprise, nobody makes replacement parts anymore, so you'll have to make do with what you've got (everyone else switched to telephones a hundred years ago, who knew).

The part that broke translated low/high signals into dots, dashes and pauses, so you will have to compensate for this. The machine now populates the ticker tape at every time step with a "-" if the signal is low, and a "+" if the signal is high. Dots and dashes are represented by consecutive time steps containing high signals, pauses are represented by consecutive time steps contains low signals. Your input contains N messages in this new encoding.

This introduces a complication – each telegraph operator broadcasts at a different rate, according to the following constraints:

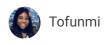
- Dots take the shortest period of time.
- Pauses between dots and dashes have the same duration as dots.
- Dashes are at least three times as long as dots.
- Pauses between characters have the same duration as dashes.
- Pauses between words are at least twice as long as pauses between characters.
- Durations are always a whole multiple of time steps.
- The timings of dots, dashes and the various pauses are consistent within a message (i.e. two dots in the same message will take exactly the same time, etc).

You will need to figure out each operator's timings from their message.

### **Constraints**

# FACEBOOK Coding Competitions

# FB Hack > 2021 > EMEA Coding Challenge 2021



Keymaster	
A1: Easy	10 pt
A2: Medium	20 pt
A3: Hard	50 pt
Dots and Dashes	
B1: Easy	10 pt
B2: Medium	15 pt
B3: Hard	40 pt
CI Pipelines	
C1: Easy	5 pt
C2: Medium	15 pt
C3: Hard	50 pt
Data Centers	
D1: Easy	10 pt
D2: Medium	20 pt
D3: Hard	65 pt
Commercial Operations	
E1: Easy	20 pt
E2: Medium	40 pt
E3: Hard	50 pt
Auto-correct	
F1: Easy	5 pt

lines, each containing the description of an open source project **sent by a different telegraph operator**, this time encoded using its *low/high signal*. Dot, dash, and pause timings are guaranteed to be unambiguous for each message.

# **Output**

Your output should be a file containing N lines, with the ith line containing the decoded version of the ith encoded message in the input, **in all-caps**.

# **Explanation of Sample**

The first message is encoded by a very efficient operator taking 1, 3 and 6 time steps for dot, character and word pauses.

The second message was sent by a slower operator taking 2, 7 and 14 time steps respectively, but sending the same text.

The last message is a longer piece of text transmitted once again by the efficient operator.

# Sample Input



# Sample Output

FB HACK! FB HACK! LINUX KERNEL CONTROL GRO