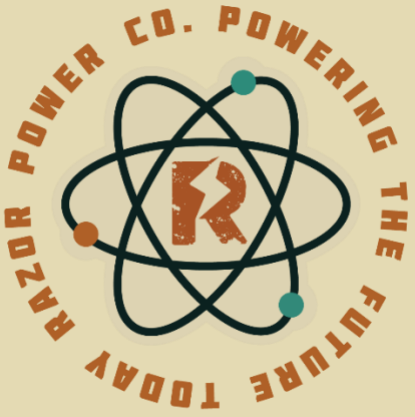
RazorHack 2024 Writeup

“brainfrick 😵 🛸”

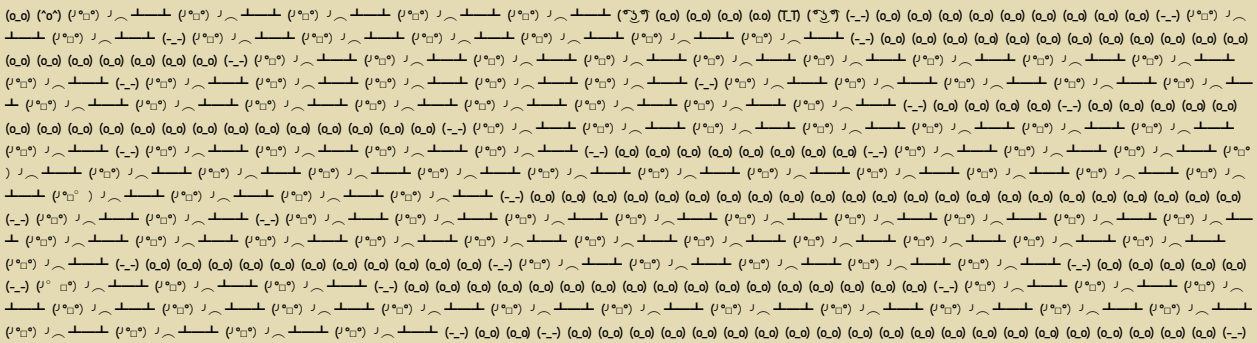


One of RazorPower's System Analyst, Ben, has developed a new programming language he calls BrainFrick, and it’s unlike anything I've seen before. I’m stumped trying to understand it. Can you help me crack the code and reveal the hidden message?

The last step is decoding the ciphertext!

Author: Kate

In this challenge you are given a file called emojiverse.txt that contains:

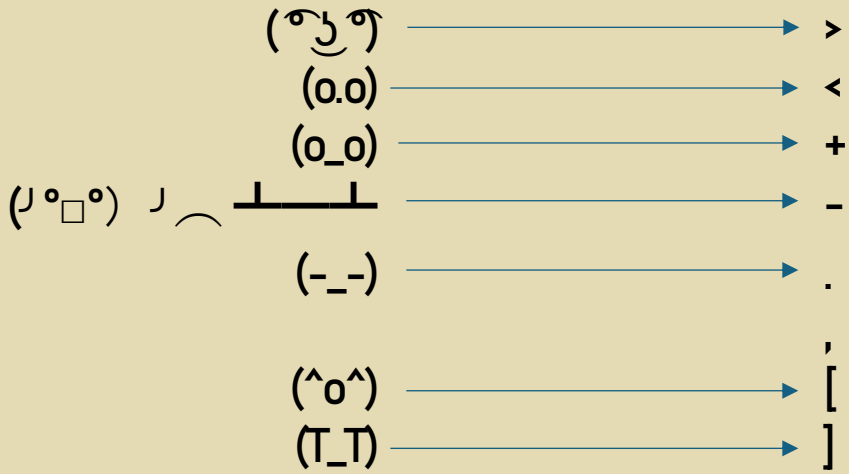


BrainFrick is inspired by the programming language BrainF*ck, known for its minimalistic and unique design. BrainF*ck operates with only eight distinct commands. In BrainFrick, we've taken this concept to the next level by using emojis in place of BrainF*ck's standard symbols.

The eight language commands each consist of a single character:

Character	Meaning
`>`	Increment the data pointer by one (to point to the next cell to the right).
`<`	Decrement the data pointer by one (to point to the next cell to the left).
`+`	Increment the byte at the data pointer by one.
`-`	Decrement the byte at the data pointer by one.
`.`	Output the byte at the data pointer.
`,`	Accept one byte of input, storing its value in the byte at the data pointer.
`[`	If the byte at the data pointer is zero, then instead of moving the instruction pointer forward to the next command, jump it *forward* to the command after the *matching* `]` command.
`]`	If the byte at the data pointer is nonzero, then instead of moving the instruction pointer forward to the next command, jump it *back* to the command after the *matching* `[` command.[a]

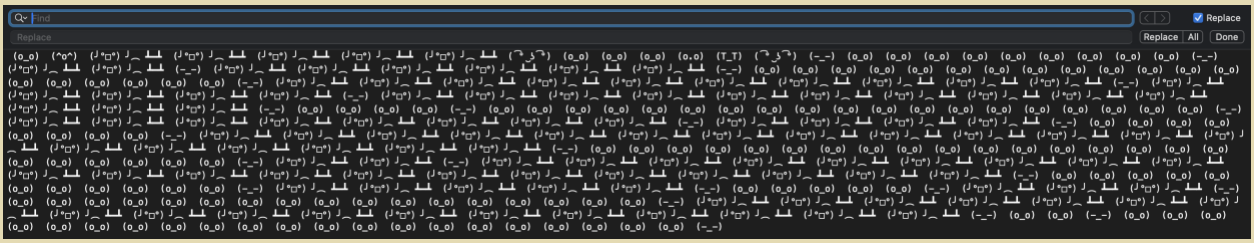
I've used a set of 7 unique emojis in BrainFrick, each mapped one-to-one to a specific BrainF*ck command. This injective mapping ensures that each emoji corresponds directly to a single BrainF*ck character, making the translation easy!



To translate the emojis into BrainF*ck characters, you have a couple of options:

Open the file in a text editor like Notepad and replace each emoji with its corresponding BrainF*ck symbol.

- Windows: You can use CTRL + H to find and replace
- Mac: You can use CTRL + F and then click the replace checkbox



Alternatively, you can automate this with a simple Python script. The maketrans method in Python allows you to create a translation table with key-value pairs, where each emoji is mapped to its BrainF*ck equivalent for easy substitution.

```
text = "+[----->+++<]>.+++++++++.--.------.++++++++++++++++++.----------.-----.-------------.++++.++++++++++++++++.--------.----.++++++++.
------------------.++++++++++++++++++++++.--.--------------------.++++++++++.-----.+++++.---.++++++++++++++++.------------------.++.
++++++++++++++++++++."

translation_table = str.maketrans({'+': ' (o_o) ', '-': ' (ﾉ°□°) ﾉ ┻━┻ ', '.': ' (-_-) ', '[': ' (^o^) ', ']': ' (T_T) ', '<': ' (o.o) ',
'>': ' (ʘ‿ʘ) '})
new_text = text.translate(translation_table)

f = open("brainfrickoutput.txt", "w")
f.write(new_text)
f.close()

```

After translating the emojis to the BrainF*ck charcters, you get:

```
+[----->+++<]>.+++++++++.--.------.++++++++++++++++++.----------.-----.-----------
-.++++.++++++++++++++++.--------.----.++++++++.------------------
.++++++++++++++++++++++.--.--------------------.++++++++++.-----.+++++.---
.++++++++++++++++.-----------------.++.++++++++++++++++++++.
```

This can be thrown into an online BrainF*ck compiler.

The output from the BrainF*ck compiler reveals an encoded message. We can tell it's a type of substitution cipher since it follows the typical flag format: flag{}.

We know what the first four letters are! The first four letters are flag. So now we know:

g -> f this is a character shift of -1

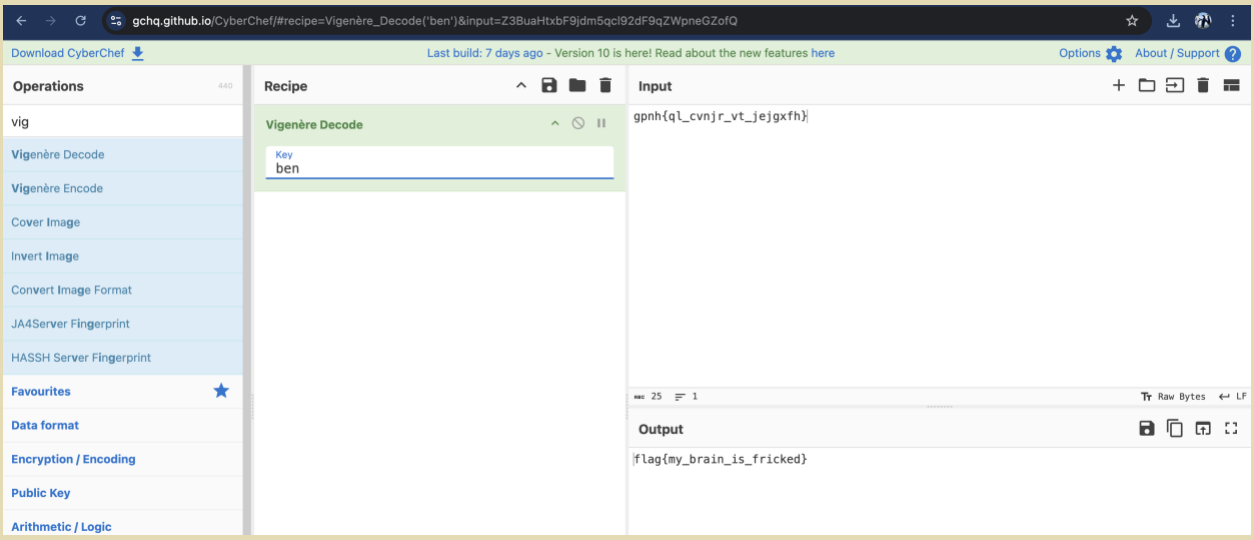
p -> l this is a character shift of -4

n -> a this is a character shift of -6

q -> g this is a character shift of -1

We're starting to see a pattern, with the number one repeating, which strongly suggests this is a Vigenère cipher! This type of cipher uses a repeating key to apply variable shifts to each character in the text, making it a bit more challenging to decode than a normal Caesar Cipher.

Let's apply this key pattern to the rest of the ciphertext. You can do this manually or use a free tool like CyberChef. Remember, counting begins at 0—so 'a' is 0, 'B' is 1, a shift of 4 results in 'E,' and a shift of 6 gives us 'N.' With this information, we notice that our decryption key is ben.



gpnh{ql_cvnjr_vt_jejgxfh} -> flag{my_brain_is_fricked}