This challenge can be harder than it looks from first glance. You are given any hints. Just the encoded flag. This requires a lot of analysis to find how it was encoded.

```
VWRXWFWI{Vr3J8NJks4Tn58PjDv3IPNc9VueGFwlu}
STOUTCTF{10101010100101010101010101001010}
```

Below the given flag I have what the flag should look like. One thing that can be useful in deciphering is repeating letters. STOUTCTF has 3 Ts in it. In the encoded flag these are all translated into W. This could mean a few things. It could lead to a potential rotation/Caesar cipher or a random cypher where a letter corresponds to a random letter or number. This can be ruled out though since that is hard to repeat and almost impossible for a challenge at least to decode (Unless you exploit something with the randomness of the seed).

The next step is to determine the rotations and write a script to decipher based on rotations. This can be done by hand but its much easier to script it. Looking at the difference between T and W is 3. This is the rotations. Now its time to build the script.

```python
def rotation_cipher(text, shift):
    """
    Encrypt or decrypt text using a rotation cipher (Caesar cipher).

    Args:
        text (str): The input text to process.
        shift (int): The number of positions to shift each character.

    Returns:
        str: The processed text after applying the cipher.
    """
    result = []

    for char in text:
        if char.isalpha():  # Process only alphabetic characters
            # Determine whether the character is uppercase or lowercase
            start = ord('A') if char.isupper() else ord('a')
            # Rotate the character and keep it within the alphabet bounds
            rotated_char = chr((ord(char) - start + shift) % 26 + start)
            result.append(rotated_char)
        else:
            # Non-alphabetic characters remain unchanged
            result.append(char)

    return ''.join(result)
```

```
# Example usage
text = "VWRXWFWI{Vr3J8NJks4Tn58PjDv3IPNc9VueGFwlu}"
shift = 3
encrypted = rotation_cipher(text, shift)
decrypted = rotation_cipher(text, -shift)

print(f"Original: {text}")
print(f"Encrypted: {encrypted}")
print(f"Decrypted: {decrypted}")
```

Shift was set to 3 for the rotations. Upon running the script you get the deciphered flag.

```
Original: VWRXWFWI{Vr3J8NJks4Tn58PjDv3IPNc9VueGFwlu}
Encrypted: YZUAZIZL{Yu3M8QMnv4Wq58SmGy3LSQf9YxhJIzox}
Decrypted: STOUTCTF{So3G8KGhp4Qk58MgAs3FMKz9SrbDCtir}
```