Rot13

ROT13 is a simple letter substitution cipher that replaces a letter with the letter 13 letters after it in the alphabet. ROT13 is an example of the Caesar cipher.

Create a function that takes a string and returns the string ciphered with Rot13. If there are numbers or special characters included in the string, they should be returned as they are. Only letters from the latin/english alphabet should be shifted, like in the original Rot13 "implementation".

Please note that using `encode` is considered cheating.

```
def rot13(text):
    def shift_char(c):
        if 'a' <= c <= 'z':
            # Shift within lowercase letters 'a'-'z'
            return chr((ord(c) - ord('a') + 13) % 26 + ord('a'))
        elif 'A' <= c <= 'Z':
            # Shift within uppercase letters 'A'-'Z'
            return chr((ord(c) - ord('A') + 13) % 26 + ord('A'))
        else:
            # Non-alphabetic characters remain unchanged
            return c

    # Apply the shift to each character in the string
    return ''.join(shift_char(c) for c in text)
```

the if body statement suggests

- that if the current letter is either lowercase or uppercase
- it calculates the next position using unicode characters

- ord(c) - which gives the unicode code
- basically numerical value of ‘c’ - numerical value of ‘a’

we check for position relativity as unicode is more than just letters and alphabets, which is why we add ord(’a’) again at the very end

line 5 gives relative position of current character relative to a and effectively + 13 to shift it, to ensure it doesnt exceed 26, using a modulo of 26 ensures that the remainders are carried on over after a

```
import string
from codecs import encode as _dont_use_this_

def rot13(message):
    alpha = "ABCDEFGHIJKLMNOPQRSTUVWXYZ"
    outputMessage = ""
    for letter in message:
        if letter in alpha.lower():
            outputMessage += alpha[(alpha.lower().index(letter)
        elif letter in alpha:
            outputMessage += alpha[(alpha.index(letter) +13) % 2
        else:
            outputMessage += letter
    return outputMessage
```

better practice