**Start les** (5 min)

* Leerdoel: Een programma schrijven

**Uitleg Caesar code** (10 min)

* Uitleg hoe Caesar code werkt
* Uitleg Rot13
  + rot13() functie doornemen

**Opdracht 1 + 2**

* Studenten maken opdracht 1 en 2.
* Bespreek oplossingen met de klas

**LoL**  (5 min)

* Herinnering dat lijsten ook lijsten mogen bevatten, zogenaamde LoLs

**Opdracht 3**

* Probeer, plan, programmeer.
  + Studenten eerst met ideeen laten komen. Hoe zouden ze het op papier kunnen oplossen?
* Laat studenten een oplossing programmeren

**Afronding.**

**Oplossing.**

Deze oplossing gebruikt het idee dat de letter ‘e’ de meest voorkomende letter is in de nederlandse taal. Het telt welke letter in de code het meeste voorkomt, gaat er dan vanuit dat dit de ‘e’ is en berekent de gebruikt rotatie.

def rot(c, n):

    hoofdletters = "ABCDEFGHIJKLMNOPQRSTUVWXYZ"

    letters = "abcdefghijklmnopqrstuvwxyz"

    if c in hoofdletters:

        c = chr(ord(c) + n)

        if c not in hoofdletters:

            c = chr(ord(c) - 26)

    elif c in letters:

        c = chr(ord(c) + n)

        if c not in letters:

            c = chr(ord(c) - 26)

    return c

def encipher(s, n):

    if len(s) == 0:

        return s

    else:

        return rot(s[0], n) + encipher(s[1:], n)

def count\_letter(s, c):

    if len(s) == 0:

        return 0

    elif s[0] == c :

        return 1 + count\_letter(s[1:], c)

    else:

        return count\_letter(s[1:], c)

def count\_letters(alphabet, s):

    if len(alphabet)== 0:

        return[]

    return [zcount\_letter(s, alphabet[0])] + count\_letters(alphabet[1:], s)

def max(L):

    if len(L) == 1:

        return L[0]

    M = max(L[1:])  # De max van de REST van L

    if L[0] > M:

        return L[0]

    else:

        return M

def decipher(s):

    letters = "abcdefghijklmnopqrstuvwxyz"

    lc = count\_letters(letters, s)

    freq\_letter\_index = lc.index(max(lc))

    if 5 - freq\_letter\_index >= 0:

        rotation = (4 - freq\_letter\_index)

    else:

        rotation = 26 + (4 - freq\_letter\_index)

    sdecoded = encipher(s, rotation)

    return sdecoded

print(decipher("Bzdrzqbhiedq? Hj gda khdudq Bzdrzqrzkzcd."))