Labo 4

Computationeel Denken

1. Schrijf een functie
   gemeenschappelijke_letters (woord_1 , woord_2) die twee strings krijgt als parameters.
   De functie retourneert een nieuwe string bestaande uit de letters die beide strings gemeenschappelijk hebben.

   Je mag hoofletters beschouwen als verschillend van kleine letters, maar iedere letter mag slechts één keer gerapporteerd worden. (bijvoorbeeld, de strings "een" en "peer" hebben slechts één letter gemeenschappelijk, namelijk de letter "e").

   Uitvoer :
   Geef een eerste woord : peer
   Geef een tweede woord : perfect

   Beide woorden hebben 3 letters gemeenschappelijk, namelijk de letters : p e r

2. Schrijf een functie
   get_schone_string (zin) die een string als argument krijgt, en die dan een nieuwe string retourneert die hetzelfde is als het argument, maar waarbij ieder teken dat geen letter is, wordt vervangen door een spatie.

   Uitvoer :
   Geef een geheime zin in :

   Waarom5wordt(er%in*oktoberéal&kerstversiering§verkocht

   Waarom wordt er in oktober al kerstversiering verkocht

   Tip:
   Je kunt testen of een teken acceptabel is met eenvoudige vergelijkingen, bijvoorbeeld, alle kleine letters kun je herkennen omdat ze de test letter >= 'a' and letter <= 'z' doorstaan. Let op: Zorg dat hoofdletters ook herkend worden.

   Merk op:
   Strings zijn **onveranderbaar**, je kan m.a.w. niet zomaar een karakter vervangen door een spatie. De volgende code geeft een foutmelding.

```
woord = "paashaas"
woord[0] = "P"
print(woord)
```

3. Schrijf een functie **decoder (geheim , sleutel)**
die een string en een dictionary als argument krijgt en een gedecodeerde boodschap teruggeeft. Je gebruikt de dictionary om letters of tekens te vervangen.

```
sleutel = {
        '&' : 'n' ,
        '@' : 'a' ,
        '€' : 'e' ,
        '0' : 'o' ,
        '!' : 'i',
        'g' : 'd' ,
        'd' : 'g' ,
        'H' : 'f' ,
        'w' : 'b'
}
```

Kraak de volgende boodschap :

"k0HH!€ !s wl!jkw@@r h€t €cht€ vl0€!w@r€ d0ug"

Schrijf nu ook een functie **encoder (boodschap , sleutel)**
die een string en een dictionary als argument krijgt en een gecodeerde boodschap teruggeeft.

Deze functie draait eerst de sleutel om en gebruikt dan de functie **decoder ( )** om de correcte tekens te vervangen.

Uitvoer :

Geef een boodschap :
Dit is een idioot wachtwoord!

D!t !s €€& !g!00t w@chtw00rg!

4. Gaap, lijger en grolar bear,... Er bestaan verschillende bijzondere kruisingen tussen dieren.

   Schrijf twee functies die helpen namen te verzinnen voor het nageslacht van de kruising tussen twee verschillende diersoorten.

   De functie **splits_woord (dier)**
   moet een ingevoerde string splitsen in een prefix en een suffix.

   De prefix bestaat uit alle medeklinkers aan het begin van het woord, de suffix start vanaf de eerste klinker.

   Sla beide delen van het woord op in een dictionary en geef deze dictionary mee als output van de functie.

   De dictionary die teruggegeven wordt bij de invoer van het woord "schaap" zou dus de volgende moeten zijn :

   { 'prefix' : 'sch' , 'suffix' : 'aap' }

   Schrijf nu een functie **nieuwe_kruising (dier_1 , dier_2)** die 2 namen van dieren als argument krijgt.

   Deze functie voegt de prefix van het eerste dier samen met de suffix van het tweede dier door gebruik te maken van de functie **splits_woord ( )** binnenin de definitie van deze nieuwe functie.

   De output is een string met de naam van het nieuwe nageslacht.

   Uitvoer :

   Geef een eerste dier : leeuw
   Geef een tweede dier : tijger

   lijger

5. Maak de onderstaande dictionaries aan:

```
puntenlijst = [
  {
    "naam" : "Jan" ,
    "familienaam" : "Janssen" ,
    "score" : 15 ,
  },
  {
    "naam" : "William" ,
    "familienaam" : "Williamson ",
    "score" : 6 ,
  },
  {
    "naam" : "Bill" ,
    "familienaam" : "Beaux" ,
    "score" : 11 ,
  },
  {
    "naam" : "Amy" ,
    "familienaam" : "Stake" ,
    "score" : 19 ,
  },

  {
    "naam" : "Miss" ,
    "familienaam" : "Chin" ,
    "score" : None ,
  }
]

boodschap = {
        0 :     ["Edde gij gestudeerd? " , "Je bent niet geslaagd." ],
        10 :    ["Geslaagd, " , "Met de hakken over de sloot." ],
        14 :    ["Proficiat, " , "Doe zo verder!" ],
        18 :    ["Uitmuntend, " , "Heel goed gedaan!" ],
        None :  ["Beste " , "Maak een afspraak om de test in te halen." ]
}
```

Schrijf een hoofdprogramma die gebruik maakt van een lus om door alle studenten te lopen en een boodschap schrijft voor elke student afhankelijk van zijn of haar score.

Uitvoer :

Proficiat, Jan Janssen. Doe zo verder! Jouw score is 15 op 20.
Edde gij gestudeerd? William Williamson. Je bent niet geslaagd. Jouw score is 6 op 20.
Geslaagd, Bill Beaux. Met de hakken over de sloot. Jouw score is 11 op 20.
Uitmuntend, Amy Stake. Heel goed gedaan! Jouw score is 19 op 20.
Beste Miss Chin. Maak een afspraak om de test in te halen.

6. Schrijf een functie **decoder (tekening , teken)**
die twee keer een string als argument krijgt: een gecodeerde tekening en een karakter waarmee de tekening kan ontcijferd worden.
De functie geeft de originele tekening terug.

Elke lijn in de gecodeerde tekening is als volgt gecodeerd:

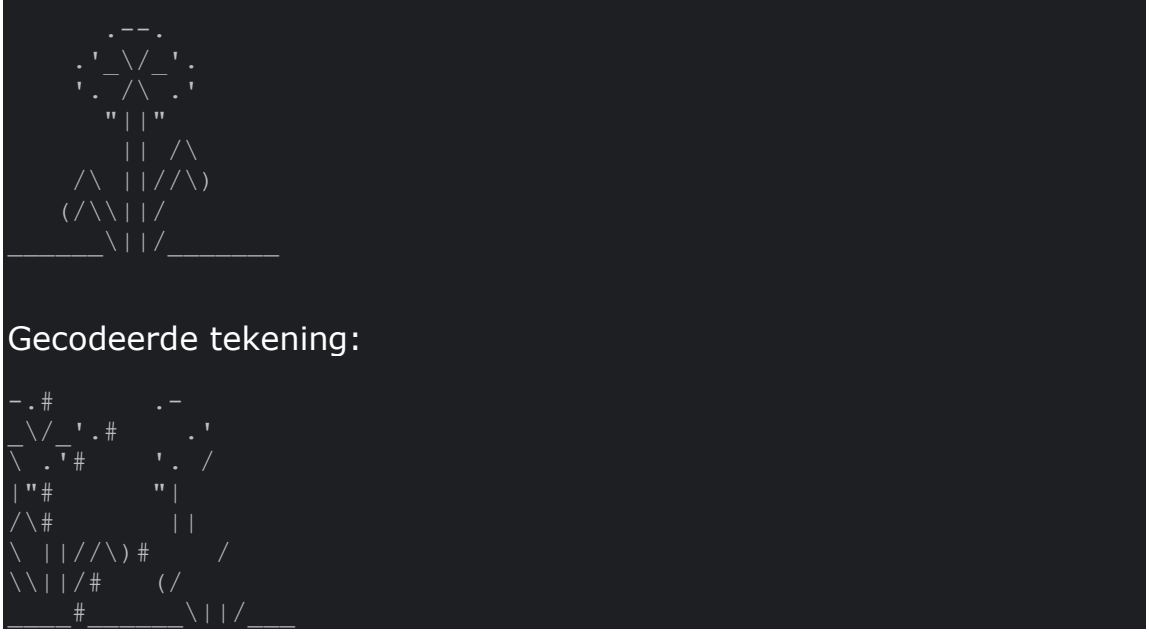
Gecodeerde lijn : 456789#123

Je decodeert elke lijn in de tekening door de regel te splitsen aan het teken (#) en het eerste deel te wisselen met het tweede deel.

De originele lijn was dus : 123456789

Voorbeeld van een gecodeerde afbeelding:

Originele tekening:



Gecodeerde tekening:

Schrijf een hoofdprogramma die een gecodeerde tekening inleest uit een tekstbestand en de gedecodeerde tekening uitprint.

Het hoofdprogramma kan er zo uitzien :

```
file =  open(r'C:\Users\r-nummer\PycharmProjects\Labo4\

        tekening1.txt' , 'r' )

tekening = file.read()

print("De originele tekening : ")

print( decoder( tekening , '#' ) )
```

De te gebruikten tekstbestanden kan je op Toledo terugvinden.