Question-

a. Create an “encode_rot()” function to encode any given strings using ROT algorithm. The input should contain a key and a string of text. The key can be any integers both negative and positive (-12: turn left 12 positions, 36: turn right 36 positions). Only alphabet letters are encoded.

The following two lines of your code will generate an output of “Ocejkpg ECP ngctp 2 !!!“.

clear_text=” Machine CAN learn 2 !!!”

encode_rot(clear_text, 28)

```
## Encoding using rot13

def encode_rot(text,key):
  result = ''
  if key < -12 or key > 36:
    return 'Invalid Key'


  for eachchar in text: # Checking if the eachchar is alphabet
    if eachchar.isalpha():
      if eachchar.isupper():
        firstchar=ord('A')
        result += chr((ord(eachchar)-firstchar+key) %26 + firstchar) # Explanation is below

      elif eachchar.islower():
        firstchar=ord('a')
        result += chr((ord(eachchar)-firstchar+key) %26 + firstchar)

    else:
      result+=eachchar #If the eachchar is non-alphabet, then it will concat the same to original string

  return result
```

Explanation-

rot13 encoding is a simple encoding mechanism. Any character will be replaced by n number of char ahead position e.g. if key or n is 13 then 'A' will be replaced 'N', 'B' by 'O', 'N' by 'A', 'O' by 'B' etc. A-M are replaced by N-Z and viceversa.

I have created a function which will take clear_text and key as input.

Logic-

Considering key=13 for explanation.

a. It is important to find position of first letter 'A'. It is done by 'ord' function. It will return 65.

b. Any letter ord will be greater than letter A and so we need to substract ord of that number with letter A. Then we can find positional value of the letter w.r.t. letter A. Then I am adding key as 13. It will give me encoding character.

c. As there are 26 characters in english, it is crucial to consider 26. rot function implements mirror image technique. So let's say if we think of rot13, then mirror image of A-M is N-Z.

So let's say for letter A, step number 'b' will return 13. As there is possibility of step b return value more than 26, so I need to use %26 to find the number residual after z or a.

Step will return 13%26 = 13. I need to add chr(A) to find actual chr value of the encoded char. When I use char function, it will return N.

For Letter N, %26 step will return 26%26 = 65. When I use char function, it will return A.

e. At the end, I have used a logic to exclude non characters from the encryption logic with inclusion of if and else statements.

f. I have added a logic of invalid key if the input key < -12 or key > 36.

```
encode_rot('Welcome to Machine learning',28)
```

```
'Ygneqog vq Ocejkpg ngctpkpi'
```

```
encode_rot('Machine CAN learn 2 !!!',28)
```

```
'Ocejkpg ECP ngctp 2 !!!'
```

```
encode_rot('Machine CAN learn 2 !!!',16)
```

```
'Cqsxydu SQD buqhd 2 !!!'
```

```
encode_rot('Cqsxydu SQD buqhd 2 !!!',-16)
```

```
'Invalid Key'
```

```
encode_rot('Ocejkpg ECP ngctp 2 !!!',-28)
```

```
'Invalid Key'
```

```
encode_rot('Machine CAN learn 2 !!!',-20)
```

```
'Invalid Key'
```

```
encode_rot('Machine CAN learn 2 !!!',40)
```

```
'Invalid Key'
```

Question-

b. Create a decode_rot() to decode a ciphertext. The input only contains the ciphertext. The output contains the cleartext and the key that was used to encode text. The key will be between 0 and 25. (hint: Compare your decoded clear text with a dictionary text file and decide which one has the most dictionary words.)

The following two lines will generate an output of

The clear text is : “Data is like people, interrogate it hard enough and it will tell you whatever you want to hear.” The key is 16

cipher_text= “Tqjq yi byau fuefbu, ydjuhhewqju yj xqht udekwx qdt yj mybb jubb oek mxqjuluh oek mqdj je xuqh.”

decode_rot(cipher_text)

Proposed Logic-

a. When I will start to decode, there will be 26 different variables as for each 1 shift in key, the decode will return a new letter. So it is important to know which decoded word is an actual english word.

b. First step would be load dictionary into a variable and then match decoded word with dictionary. In the process, I am changing key value from 1 to 25 and wherever there is a match of decoded word with dictionary, it is the correct key for me.

c. To find the decoded word, I am using encode_rot function with negative of key. As rot function is based on mirror image logic, negative is used.

```
##Function to load dictionary.txt.
## Downloaded dictionary.txt from popular website

from google.colab import drive
drive.mount('/content/drive')

def load_dictionary():
  words = []
  f = open("/content/drive/MyDrive/Colab Notebooks/dictionary.txt", "r")
  words = f.read().splitlines()
  return words
```

```
Mounted at /content/drive
```

```
dict_words=load_dictionary()
print(dict_words)
```

```
['AARHUS', 'AARON', 'ABABA', 'ABACK', 'ABAFT', 'ABANDON', 'ABANDONED', 'ABANDONING', 'ABANDONMENT', 'ABANDONS', 'ABASE', 'ABASED', 'ABAS
```

```
## Function to find any word in english dictionary and it will return true or false based on match

def find_dictionary(word):
  dict_words=load_dictionary()
  for eachword in dict_words:
    if eachword==word.upper():
      return True
  return False
```

```
find_dictionary('Machine')
```

```
True
```

```
find_dictionary('can')
```

```
True
```

```
find_dictionary('learn')
```

```
True
```

```
find_dictionary('to')
```

```
True
```

```
## Function to check if any word is alphabatic or not

def is_word_char(word):
  for eachchar in word:
    if not eachchar.isalpha():
      return False
  return True
```

```
is_word_char('!!!')
```

```
False
```

```
is_word_char('Machine')
```

```
True
```

```
##encoding function without key limit as I need to use key 0 to 25 for the decoding function

def encode_rot2(text,key):
  result = ''


  for eachchar in text: # Checking if the eachchar is alphabet
    if eachchar.isalpha():
      if eachchar.isupper():
        firstchar=ord('A')
        result += chr((ord(eachchar)-firstchar+key) %26 + firstchar) # Explanation is below

      elif eachchar.islower():
        firstchar=ord('a')
        result += chr((ord(eachchar)-firstchar+key) %26 + firstchar)

    else:
      result+=eachchar #If the eachchar is non-alphabet, then it will concat the same to original string

  return result
```

```
import re

def split_special_char(word):
  word_list=re.split(r'(\W)',word)
  return word_list



def if_alpha_special_char(word):

  count=0

  for eachchar in word:
    if not eachchar.isalpha():
      count+=1
      continue
  if count==len(word):
    return 'AllSpecial'
  elif count==0:
    return 'AllAlpha'
  else:
    return 'AlphaSpecial'
```

```
print(if_alpha_special_char('Machine'))
```

```
AllAlpha
```

```
print(if_alpha_special_char('Machine2'))
```

```
AlphaSpecial
```

```
print(if_alpha_special_char('!!!'))
```

```
AllSpecial
```

```
print(if_alpha_special_char('Machine,'))
```

```
AlphaSpecial
```

```
print(if_alpha_special_char('Mach,ine,'))
```

```
AlphaSpecial
```

```
print(if_alpha_special_char('Mach,ine'))
```

```
AlphaSpecial
```

```
## Function to decode any cypher_text

def decode_rot(cyper_text):
  result = ''

  cypher_words=cyper_text.split() ##it will return list of each word in the cypher text
  cypher_word_lists=[]

  for eachword in cypher_words:
    if if_alpha_special_char(eachword) == 'AlphaSpecial': #checking if any word is alphabatic or not
      cypher_word_list=split_special_char(eachword) #it will return list of each word after split with special character along with special cl
      cypher_word_lists=cypher_word_lists+cypher_word_list
    else :
      cypher_word_lists=cypher_word_lists+[eachword]

  for eachword in cypher_word_lists:
    for i in range(0,25): #ranging between 1 to 26
      decoded_word = encode_rot2(eachword,-i) # negative of code will show the decoded word of any encoded word

      if if_alpha_special_char(decoded_word) == 'AllAlpha': #checking if any decoded word is a real alphabatic word or not
        if find_dictionary(decoded_word) :  # checking decoded word against dictionary
          code=i
          result+=decoded_word
          result+=' '
          break # if there is a match break from loop
      elif if_alpha_special_char(eachword) == 'AllSpecial': #add non alphabatic word to final result
        result=result[:-1] #it is needed to remove space after special character separated word. e.g. "machine," will give result like "machi
        result+=eachword
        result+=' '
        break
      else:
        continue

  result=result.strip() # if there is any additional space either to start or end of result, it is important to remove it.
  return result,code
```

```
encode_rot('Data is like people, interrogate it hard enough and it will tell you whatever you want to hear.',16)
```

```
'Tqjq yi byau fuefbu, ydjuhhewqju yj xqht udekwx qdt yj mybb jubb oek mxqjuluh oek mqdj je xuqh.'
```

```
decode_rot('Tqjq yi byau fuefbu, ydjuhhewqju yj xqht udekwx qdt yj mybb jubb oek mxqjuluh oek mqdj je xuqh.')
```

```
('Data is like people, interrogate hard enough and will tell you whatever you want to hear.',
 16)
```

Annotation-

The above approach will vary if there is anonymous words like special characters in middle of word. In that case, there is need of data cleaning.
I am only considering a standard line with appropriate position of special characters in the line.