```python
#Dictionaries Challenge 25:  Code Breakers App

from collections import Counter

print("Welcome to the Code Breakers App")

#List of elements to remove from all text for analysis
non_letters = ['1','2','3','4','5','6','7','8','9','0',' ','.','?','!',',','"',"'",':',';','(',')','%','$','&','#','\n','\t']

#Comment out user input for key phrase 1
#Information for the first key key_phrase_1
#key_phrase_1 = input("Enter a word or phrase to count the occurrence of each letter: ").lower().strip()

#Hard code a pre-determined key_phrase_1 for communication purposes
key_phrase_1 = """
To Sherlock Holmes she is always the woman. I have seldom heard him mention her under any other name.
In his eyes she eclipses and predominates the whole of her sex. It was not that he felt any emotion akin to love for Irene Adler.
All emotions, and that one particularly, were abhorrent to his cold, precise but admirably balanced mind.
He was, I take it, the most perfect reasoning and observing machine that the world has seen,
but as a lover he would have placed himself in a false position.
He never spoke of the softer passions, save with a gibe and a sneer.
They were admirable things for the observer excellent for drawing the veil from men's motives and actions.
But for the trained reasoner to admit such intrusions into his own delicate and finely adjusted temperament was to introduce
a distracting factor which might throw a doubt upon all his mental results.
Grit in a sensitive instrument, or a crack in one of his own highpower lenses,
would not be more disturbing than a strong emotion in a nature such as his.
And yet there was but one woman to him, and that woman was the late Irene Adler, of dubious and questionable memory.
I had seen little of Holmes lately. My marriage had drifted us away from each other.
My own complete happiness, and the homecentred interests which rise up around the man who first finds himself master of his own establishment,
were sufficient to absorb all my attention, while Holmes, who loathed every form of society with his whole Bohemian soul,
remained in our lodgings in Baker Street, buried among his old books, and alternating from week to week between cocaine and ambition,
the drowsiness of the drug, and the fierce energy of his own keen nature.
He was still, as ever, deeply attracted by the study of crime,
and occupied his immense faculties and extraordinary powers of observation in following out those clues,
and clearing up those mysteries which had been abandoned as hopeless by the official police.
From time to time I heard some vague account of his doings: of his summons to Odessa in the case of the Trepoff murder,
of his clearing up of the singular tragedy of the Atkinson brothers at Trincomalee,
and finally of the mission which he had accomplished so delicately and successfully for the reigning family of Holland.
Beyond these signs of his activity, however, which I merely shared with all the readers of the daily press,
I knew little of my former friend and companion.
"""
key_phrase_1 = key_phrase_1.lower()

#Removing all non letters from key_phrase_1
for non_letter in non_letters:
    key_phrase_1 = key_phrase_1.replace(non_letter, '')

total_occurrences = len(key_phrase_1)

#Create a counter object to tally the number of each letter
letter_count = Counter(key_phrase_1)

#Determine the frequency analysis for the message
print("\nHere is the frequency analysis from key phrase 1: ")
print("\n\tLetter\t\tOccurrence\tPercentage")
for key, value in sorted(letter_count.items()):
    percentage = 100*value/total_occurrences
    percentage = round(percentage, 2)
    print("\t" + key + "\t\t" + str(value) + "\t\t" + str(percentage) + "%")

#Make a list of letters from highest occurrence to lowest
ordered_letter_count = letter_count.most_common()
key_phrase_1_ordered_letters = []
for pair in ordered_letter_count:
    key_phrase_1_ordered_letters.append(pair[0])

#Print the list
print("\nLetters ordered from highest occurrence to lowest: ")
for letter in key_phrase_1_ordered_letters:
    print(letter, end='')

#Comment out user input for key_phrase_2
#Information for the second key key_phrase_2
#key_phrase_2 = input("\n\nEnter a word or phrase to count the occurrence of each letter: ").lower().strip()

#Hard code a pre-determined key_phrase_2 for communication purposes.
key_phrase_2 = """
Quite so! You have not observed. And yet you have seen.
That is just my point. Now, I know that there are seventeen steps, because I have both seen and observed.
By the way, since you are interested in these little problems,
and since you are good enough to chronicle one or two of my trifling experiences, you may be interested in this.
He threw over a sheet of thick, pink tinted notepaper which had been lying open upon the table.
It came by the last post, said he. Read it aloud.
The note was undated, and without either signature or address.
There will call upon you tonight, at a quarter to eight o'clock,
it said, "a gentleman who desires to consult you upon a matter of the very deepest moment.
Your recent services to one of the royal houses of Europe have shown that you are one who may safely be trusted
with matters which are of an importance which can hardly be exaggerated.
This account of you we have from all quarters received.
Be in your chamber then at that hour, and do not take it amiss if your visitor wear a mask.
This is indeed a mystery, I remarked. What do you imagine that it means?
I have no data yet. It is a capital mistake to theorise before one has data.
Insensibly one begins to twist facts to suit theories, instead of theories to suit facts.
But the note itself. What do you deduce from it?
I carefully examined the writing, and the paper upon which it was written.
The man who wrote it was presumably well to do, I remarked, endeavouring to imitate my companion's processes.
Such paper could not be bought under half a crown a packet.
It is peculiarly strong and stiff.
"""
key_phrase_2 = key_phrase_2.lower()
```

```python

#Removing all non letters from key_phrase_2
for non_letter in non_letters:
    key_phrase_2 = key_phrase_2.replace(non_letter, '')

total_occurrences = len(key_phrase_2)

#Create a counter object to tally the number of each letter
letter_count = Counter(key_phrase_2)

#Determine the frequency analysis for the message
print("\n\nHere is the frequency analysis from key phrase 2: ")
print("\n\tLetter\t\tOccurrence\tPercentage")
for key, value in sorted(letter_count.items()):
    percentage = 100*value/total_occurrences
    percentage = round(percentage, 2)
    print("\t" + key + "\t\t" + str(value) + "\t\t" + str(percentage) + "%")

#Make a list of letters from highest occurrence to lowest
ordered_letter_count = letter_count.most_common()
key_phrase_2_ordered_letters = []
for pair in ordered_letter_count:
    key_phrase_2_ordered_letters.append(pair[0])

#Print the list
print("\nLetters ordered from highest occurrence to lowest: ")
for letter in key_phrase_2_ordered_letters:
    print(letter, end='')

#Encode/Decode a given message using key_phrase_1 and key_phrase_2
choice = input("\n\nWould you like to encode or decode a message: ").lower()
phrase = input("What is the phrase: ").lower()

#Removing all non letters from the users phrase
for non_letter in non_letters:
    phrase = phrase.replace(non_letter, '')

#User wants to encode a message
if choice == 'encode':
    encoded_phrase = []
    for letter in phrase:
        index = key_phrase_1_ordered_letters.index(letter)
        letter = key_phrase_2_ordered_letters[index]
        encoded_phrase.append(letter)

    print("\nThe encoded message is: ")
    for letter in encoded_phrase:
        print(letter, end='')

#User wants to decode a message
elif choice == 'decode':
    decoded_phrase = []
    for letter in phrase:
        index = key_phrase_2_ordered_letters.index(letter)
        letter = key_phrase_1_ordered_letters[index]
        decoded_phrase.append(letter)

    print("\nThe decoded message is: ")
    for letter in decoded_phrase:
        print(letter, end='')

#User entered an invalid option
else:
    print("Please type 'encode' or 'decode'.  Try again.")
```