interactive shell and assign them as strings to variable names arabic, japanese, and serbian. Finally, for each string, print the Unicode code point of each character in the string using an iteration loop pattern.

**Problems**

6.20 Write function reverse() that takes as input a phone book, that is, a dictionary mapping names (the keys) to phone numbers (the values). The function should return another dictionary representing the reverse phone book mapping phone numbers (the keys) to the names (the values).

```python
>>> phonebook = {'Smith, Jane': '123-45-67',
               'Doe, John': '987-65-43',
               'Baker, David': '567-89-01'}
>>> reverse(phonebook)
```

6.21 Write function ticker() that takes a string (the name of a file) as input. The file will contain company names and stock (ticker) symbols. In this file, a company name will occupy a line and its stock symbol will be in the next line. Following this line will be a line with another company name, and so on. Your program will read the file and store the name and stock symbol in a dictionary. Then it will provide an interface to the user so the user can obtain the stock symbol for a given company. Test your code on the NASDAQ 100 list of stock given in file nasdaq.txt.

File: nasdaq.txt

```python
>>> ticker('nasdaq.txt')
Enter Company name: YAHOO
Ticker symbol: YHOO
Enter Company name: GOOGLE INC
Ticker symbol: GOOG
...
```

6.22 The mirror image of string vow is string wov, and the mirror image wood is string boow. The mirror image of string bed cannot be represented as a string, however, because the mirror image of e is not a valid character.

Develop function mirror() that takes a string and returns its mirror image but only if the mirror image can be represented using letters in the alphabet.

```python
>>> mirror('vow')
'wov'
>>> mirror('wood')
'boow'
>>> mirror('bed')
'INVALID'
```

6.23 You would like to produce a unique scary dictionary but have a hard time finding the thousands of words that should go into such a dictionary. Your brilliant idea is to write a function scaryDict() that reads in an electronic version of a scary book, say Frankenstein by Mary Wollstonecraft Shelley, picks up all the words in it, and writes them in alphabetical
order in a new file called dictionary.txt. You can eliminate one- and two-letter words because none of them are scary.
You will notice that punctuation in the text makes this exercise a bit more complicated. You can handle it by replacing punctuation with blanks or empty strings.

```python
>>> scaryDict('frankenstein.txt')
abandon
abandoned
abbey
abhorr
abhorred
abhorrence
abhorrent
...
```

6.24 Implement function `names()` that takes no input and repeatedly asks the user to enter the first name of a student in a class. When the user enters the empty string, the function should print for every name the number of students with that name.

```python
>>> names()
Enter next name: Valerie
Enter next name: Bob
Enter next name: Valerie
Enter next name: Amelia
Enter next name: Bob
Enter next name: 
There is 1 student named Amelia
There are 2 students named Bob
There are 2 students named Valerie
```

6.25 Write function `different()` that takes a two-dimensional table as input and returns the number of distinct entries in the table.

```python
>>> t = [[1,0,1],[0,1,0]]
>>> different(t)
2
>>> t = [[32,12,52,63],[32,64,67,52],[64,64,17,34],[34,17,76,98]]
>>> different(t)
10
```

6.26 Write function `week()` that takes no arguments. It will repeatedly ask the user to enter an abbreviation for a day of the week (Mo, Tu, We, Th, Fr, Sa, or Su) and then print the corresponding day.

```python
>>> week()
Enter day abbreviation: Tu
Tuesday
Enter day abbreviation: Su
Sunday
Enter day abbreviation: Sa
Saturday
Enter day abbreviation: