Problems

7.16 The first input argument of function `index()` in Problem 6.27 is supposed to be the name of a text file. If the file cannot be found by the interpreter or if it cannot be read as a text file, an exception will be raised. Reimplement function `index()` so that the message shown here is printed instead:

```python
>>> index('rven.txt', ['raven', 'mortal', 'dying', 'ghost'])
File 'rven.txt' not found.
```

7.17 In Problem 6.34, you were asked to develop an application that asks users to solve addition problems. Users were required to enter their answers using digits 0 through 9. Reimplement the function `game()` so it handles nondigit user input by printing a friendly message like “Please write your answer using digits 0 though 9. Try again!” and then giving the user another opportunity to enter an answer.

```python
>>> game(3)
8 + 2 =
Enter answer: ten
Please write your answer using digits 0 though 9. Try again!
Enter answer: 10
Correct.
```

7.18 The blackjack application developed in Section 6.5 includes the function `dealCard()` that pops the top card from the deck and passes it to a game participant. The deck is implemented as a list of cards, and popping the top card from the deck corresponds to popping the list. If the function is called on an empty deck, an attempt to pop an empty list is made, and an `IndexError` exception is raised.

Modify the blackjack application by handling the exception raised when trying to deal a card from an empty deck. Your handler should create a new shuffled deck and deal a card from the top of this new deck.

7.19 Implement function `inValues()` that asks the user to input a set of nonzero floating-point values. When the user enters a value that is not a number, give the user a second chance to enter the value. After two mistakes in a row, quit the program. Add all correctly specified values when the user enters 0. Use exception handling to detect improper inputs.

```python
>>> inValues()
Please enter a number: 4.75
Please enter a number: 2,25
Error. Please re-enter the value.
Please enter a number: 2.25
Please enter a number: 0
7.0
```

```python
>>> inValues()
Please enter a number: 3.4
Please enter a number: 3,4
Error. Please re-enter the value.
Please enter a number: 3,4
Two errors in a row. Quitting ...
```
7.20 In Problem 7.19, the program quits only when the user makes two mistakes in a row. Implement the alternative version of the program that quits when the user makes the second mistake, even if it follows a correct entry by the user.

7.21 If you type \[Ctrl\] + \[C\] while the shell is executing the `input()` function, a `KeyboardInterrupt` exception will be raised. For example:

```python
>>> x = input()  # Typing Ctrl-C
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
KeyboardInterrupt
```

Create a wrapper function `safe_input()` which works just like function `input()` except that it returns `None` when an exception is raised.

```python
>>> x = safe_input()  # Typing Ctrl-C
>>> x
# x is None
>>> x = safe_input()  # Typing 34
34
>>> x
# x is 34
'34'
```