Activities: Exception Handling

1. Create situation where an error situation is generated in order to find out the type of error for:
   (a) accessing an element in a list by index when the index is off, e.g. `lista = ["one", "two"]; print(lista[2])`
   (b) accessing a dictionary value with a key that is not in the dictionary
   (c) opening a file that is not there.

2. Write two versions of a function that calculates the harmonic mean

$$\frac{n}{1/a_0 + 1/a_2 + \ldots + 1/a_{n-1}}$$

   of a list of integers [a_0, a_1, … , a_{n-1}]. The first version uses an if-statement to make sure that the list element is not zero. List elements with zero value are ignored. The second version uses exceptions to do the same thing. Then use the timing module to determine which version is faster. To create the list, us

```
list1 = [random.randint(0,100000) for _ in range(100000)]
```

3. Write a function using exceptions that asks the user to enter a set of integers. The integers are summed up and the function returns the mean of the integers. The user indicates that she is done by typing "Done". If the user enters a string that is not an integer, the system merely complains and keeps going. Here is the "defensive" version of the program:

```
def mean():
    accu = 0
    count = 0
    while True:
        string = input("Please enter a number or \"Done\": " )
        if string == "Done":
            if count == 0:
                print("You did not enter any numbers, you silly person.")
                return 0
            else:
                return accu/count
        elif not string.isdigit():
            print("This is not a number, you silly person.")
        else:
            accu += int(string)
            count += 1
```

4. Write two versions of a function that calculates the average of the values of a dictionary over the keys 1, …, 100. Skip over integers that are not in the dictionary. Do so using if-statements and using exceptions.