MINIA UNIVERSITY
FACULTY SCIENCE
Department of Computer Science
Data Structures Using Python

Exercises #4
Enumeration & Exceptions

4.1 By using enumerate(), Write a Python program that displays each index and the corresponding element in the list [5, 15, 35, 8, 98].

4.2 By using list comprehension and enumerate(), write a program to print the list after removing the 0th,4th,5th numbers in [12,24,35,70,88,120,155].

4.3 Consider the following lists of books titles, authors, and years of publication:

```
titles = ["Everyday Italian", "XQuery Kick Start", "Learning XML"]
authors = ["James McGovern", "Erik T. Ray", "Giada De Laurentiis"]
years = [2003, 2007, 2005]
```

Write a Python code that:

- uses ***zip()*** to form a list of tuples, named ***books***, from the given lists, where each tuple contains the book title, author, and year of publication, then
- uses ***enumerate()*** to display the books information in the following form:
  1. Everyday Italian (2005), by Giada De Laurentiis
  2. XQuery Kick Start (2003), by James McGovern
  3. Learning XML (2007), by Erik T. Ray

4.4 Write a Python program that creates a list of integers, then asks the user to enter an index of an element and a value, then attempts to write the value to the list based on the given index. If that index is out of bounds, the program should catch the exception that results, and print the following error message: "You are attempting to access an out of bounds element". Otherwise, the program should write the value to the list based on the given index and display the updated list.

4.5 Define a Python function, ***factorial(n)***, which raises a ***ValueError*** exception, with the message 'Factorial expects non-negative integers', if n < 0. Then, write a main function that calls factorial(n) with different values for n, and handles the exception should it occur.

4.6 Define a function ***capitalize_last_name()*** that accepts as argument a string with a first and a last name, and returns a string in which only the first letter of the first name is uppercase, whereas all letters of the last name are uppercase; e.g., 'mary tomas' becomes 'Mary TOMAS'.

If something other than a ***str*** object is passed as an argument, the function should raise a ***TypeError*** exception. If the ***str*** does not consist of exactly two words, the function should raise a ***ValueError*** exception.

4.7 Write an interactive calculator, where user input is assumed to be a formula that consist of a number, an operator (+, -, *, or /), and another number, separated by white space (e.g. 1 + 1). Split user input, and check whether the resulting list is valid:

- If the input does not consist of 3 elements, raise a ***FormulaError***, which is a custom Exception.

- Try to convert the first and third input to a float (like so: float_value = float(str_value)). Catch any ***ValueError*** that occurs, and raise a ***FormulaError***
- If the second input is not a valid operator, again raise a ***FormulaError***

If the input is valid, perform the calculation and print out the result. The user is then prompted to provide new input, and so on, until the user enters 'quit'.

4.8 Define an exception class called ***OverflowException***, which represents exceptions that occur when the value of a numeric variable exceeds a specified value.

Then, define a class ***Counter***, which has two instance variables: ***count*** that represents the value of the counter, and ***maxcnt*** that represents the maximum value of the counter. It has a ***constructor*** that creates a counter with a given maximum count and an initial value 0, and it has three methods: ***increment()***, which raises an OverflowException that displays the message "An overflow occurred!", if the current count equals the maximum count, otherwise it increases the current count by 1; ***reset()***, which sets the counter to zero; and ***get_count()***, which returns the current count.

Finally, using class ***Counter***, write a program that reads a sentence, and counts the frequency of each vowel (a, e, i, o, u) in it, then displays these frequencies. The program should use a separate counter for each vowel, and it should display an error message, if any of the counters overflows.