Degree in Software Engineering – Computing Basics

Unit 2.7 Exercises: Structured Data Types (II)

This document includes a collection of exercises from the second part of Unit 2.7: Structured Data Types. It is recommended that you try doing the exercises without looking at the solutions first, and then you check your answers.

Exercise 1

Write a function that takes a string as parameter and returns a new string in which the first character of each word has been capitalized, if it was not originally. To that aim, you can use the `.upper()` function, which makes any string upper-case.

Write a main program that asks the user to type any sentence and uses the above function to capitalize the first character of each word.

Example:

```
Input a sequence: there are more than 50 employees in Google
There Are More Than 50 Employees In Google
```

Exercise 2

Write a function that takes a string as parameter and replaces all vowels in it by their corresponding numeric values in *leet* language as follows: a by `4`, e by `3`, `i` by `1`, and o by `0`. Try considering upper-case vowels too.

Write a main program that asks the user to type any sentence and uses the above function to replace the vowels in it.

Example:

```
Input a sequence: back to the eighties
b4ck t0 th3 31ght13s
```

Exercise 3

Write a function that takes a non-negative integer number as parameter and returns a string that represents such a number in binary. Of course, the use of functions defined in Python such as `bin()` or `.format()` to perform the conversion is forbidden.

Write a main program that asks the user to type a non-negative integer number and then shows its binary representation on screen by using the above function.

Example:

```
Input a non-negative integer value: 87
The binary representation of 87 is 1010111
```

Exercise 4

WARNING: this exercise is not particularly easy, but you can take it as a challenge to assess your Python skills. Give it a try!

In the past, each country had different representation systems to identify bank accounts. In Spain, bank account codes (CCC, in Spanish) were composed of 20 numerical digits. In other countries, however, bank accounts could contain both numerical digits and characters, apart from the fact their length could be different. For this reason, the IBAN system was stablished to ease the management and control of bank accounts and their operations between different countries. To this aim, the IBAN system adds two identifying characters for the issuing country, and two control digits to avoid errors. Some examples are shown below:

- The Spanish account `21000813610123456789` would transform to:
  `ES79 2100 0813 6101 2345 6789`
- The German account `100000000123456789` would transform to:
  `DE91 1000 0000 0123 4567 89`
- The British account `MIDL07009312345678` would transform to:
  `GB98 MIDL 0700 9312 3456 78`

You are asked to develop a program that, given a bank account of any place in the world and the corresponding identifying two-character country code, the program generates its IBAN showing each character/digit in groups of four, as shown in the examples. The process to calculate the two-digit control code after the country code is as follows:

1. The bank account, the country code, and the value `00` are appended in a single string.
   Example: `21000813610123456789ES00`
2. Each non-numeric character is transformed into its corresponding numeric value so that A corresponds to 10 and Z corresponds to 35. Therefore, the string may increase its size after the conversion.
   Example: `21000813610123456789142800`
3. The two-digit control code is calculated as the difference of the remainder of the previously calculated number divided by 97 from 98. Note: if the resulting number is lower than 10, you will need to add a trailing 0 to have 2 digits.
   Example: `98 - 21000813610123456789142800 % 97` → `79`
4. Once the two-digit control code has been obtained, the IBAN can be obained.
   Example: `ES7921000813610123456789`

Example:

```
Input country code: GB
Input bank account: MIDL07009312345678
GB98 MIDL 0700 9312 3456 78
```

Exercise 5

Write a function that takes a matrix of size M×N as parameter and returns the corresponding transposed matrix of size N×M. Remember that such a matrix is the one whose values are swapped with respect to the main diagonal.

Write a main program that shows the correct behavior of the above function.

Example for matrix `[[0, 1, 2, 3], [4, 5, 6, 7], [8, 9, 10, 11]]`:

```
The transposed matrix is [[0, 4, 8], [1, 5, 9], [2, 6, 10], [3, 7, 11]]
The original matrix is [[0, 1, 2, 3], [4, 5, 6, 7], [8, 9, 10, 11]]
```

Exercise 6

As in the above exercise, write a function that takes a matrix of size M×N as parameter, but this time transposes such a matrix. Put another way, this function must not return any value. In this way, when this function is called for a variable that contains a matrix, its content is replaced by the corresponding transposed matrix of size N×M.

Write a main program that shows the correct behavior of the developed function, for which you may need to store a matrix in a variable and then call the above function. By printing the variable before and after calling the function, you should see if it works.

Example for matrix `[[0, 1, 2, 3], [4, 5, 6, 7], [8, 9, 10, 11]]`:

```
Before the function: [[0, 1, 2, 3], [4, 5, 6, 7], [8, 9, 10, 11]]
After the function: [[0, 4, 8], [1, 5, 9], [2, 6, 10], [3, 7, 11]]
```

Exercise 7

Write a function that takes a natural number M as parameter and returns the corresponding identity matrix of size M×M.

Write a main program that asks the user to input the size of the identity matrix to be generated, and shows it in square format on screen.

Example:

```
Input the size of the identity matrix: 3
1 0 0
0 1 0
0 0 1
```

Exercise 8

Write a function that takes a matrix as parameter and determines whether such a matrix is symmetrical with respect its main diagonal or not.

Write a main program that shows the correct behavior of the above function.

Example for matrix `[[0, 1, 2], [1, 3, 4], [2, 4, 5]]`:

```
Matrix [[0, 1, 2], [1, 3, 4], [2, 4, 5]] is symmetrical
```

Example for matrix `[[0, 1, 2], [0, 1, 2], [0, 1, 2]]`:

```
Matrix [[0, 1, 2], [0, 1, 2], [0, 1, 2]] is NOT symmetrical
```