Luminor Homework Task

International bank account number (IBAN) checker

Done by:
Valdemaras Danys

Vilnius
2021

Contents

1 System Objective

Goal of the project - to develop a system, which checks input International Bank Account Number.

2 Scripts and used algorithms

Project Scripts folder contains three files:

1. iban_algorithms.py
   This scripts contains all the needed functions to check if the input IBAN is valid or not. Each function important and represents different step in a validation process. All checking functions are made according to the prescribed standards and rules. These operating principles was taken from Wikipedia and official IBAN website. All functions in this project is made with Python. Uses only default packages and do not need any additional installations.

   - Data cleansing
     Function data_cleansing(IBAN) takes IBAN as an argument and returns IBAN without unnecessary symbols and spaces. This facilitates further inspection, secures from little mistakes that user may cause by mistyping or prevent user's confusion in case his input IBAN had formatting spaces between characters.
   - Iban check digit
     iban_check_digit(IBAN) This function takes an IBAN as an argument and returns result: True in case the IBAN passed validation and False in case the IBAN is not valid. Checking is made according to the integrity check on the IBAN using ISO 7064 (MOD 97-10) algorithm.
   - Account number check
     TODO (Validation based on the country specific validation algorithms.)
   - IBAN length
     Function iban_length(IBAN) verifies if the total length of the specified IBAN meets the length according to the country standard and returns True/False value according to the checking result.
   - Format (Structure)
     TODO (The IBAN's internal components are checked for correct format and respective positions.)

2. single_iban_check.py
   Function asks for and input - IBAN. After that, IBAN is validated. If it do not pass any of the checking functions, system is stopped and user sees message with functions name, so he would know, what is wrong with the IBAN and what test it did not pass. If the number passes all the tests, user is informed with a message.

3. iban_list_check.py
   Function asks for and input - IBANs list name with format. I would recommend using simple text files (such as .txt), because function uses python's default reading from file function. After that, IBANs are validated. If the number do not pass any of the checking functions, system is not stopped (like in single_iban_check.py function), instead user sees message with

functions name, so he would know, what is wrong with the IBAN and what test it did not pass. Also message is saved to the file "invalid_ibans.txt". Then function continues with a next number in the list. If the number passes all the tests, user is informed with a message and that number is saved to the "valid_ibans.txt" file. When all IBANs are validated, function closes files and system stops.

3 Tests

Folder with tests contains four files. Three of them are made for each of the scrips (in Scripts folder) and one file (ibans_for_testing.txt) is testing data, that is used to test iban_list_check.py script. Testing scripts uses python unittest package, which contains useful functions for simple unit tests.

4 Using instructions

Using is very simple. Firstly, according to the count of Your IBANs You need to choose script. For one (or any other small amount) You should use single_iban.py script. This function requires IBAN as input and returns answer, if it is valid or not. If you have a lot of IBANs, You probably need a iban_list_check.py script. Just save IBANs in the text file (each number in a new line) and save it in the Scripts folder. Then, after runing the script, input the name of the list. The results will be saved to "valid_ibans.txt" and "invalid_ibans,txt" files.

5 Future work

- Implement missing validations (Account number checking and Format (Structure) validation).
- Expand number of tests, so they would cover all of the possible cases (marginal, rare)
- Create graphical user interface (GUI) for the scripts, so it would be easier and more comfortable to use it for simple users.