# **0x04. UTF-8 Validation**

**AlgorithmPython**

* Weight: 1
* Project over - took place from Oct 28, 2024 6:00 AM to Nov 1, 2024 6:00 AM
* An auto review will be launched at the deadline

#### **In a nutshell…**

* **Auto QA review:** 7.0/14 mandatory
* **Altogether:**  **50.0%**
  + Mandatory: 50.0%
  + Optional: no optional tasks

For the “0x04. UTF-8 Validation” project, you will need to apply your knowledge in bitwise operations, understanding of the UTF-8 encoding scheme, and Python programming skills to validate whether a given dataset represents a valid UTF-8 encoding. Here’s a list of concepts and resources that will be helpful:

### **Concepts Needed:**

1. **Bitwise Operations in Python**:
   * Understanding how to manipulate bits in Python, including operations like AND (&), OR (|), XOR (^), NOT (~), shifts (<<, >>).
   * [Python Bitwise Operators](https://intranet.alxswe.com/rltoken/BslyYNZlXdyxW3_b0WNOcg)
2. **UTF-8 Encoding Scheme**:
   * Familiarity with the UTF-8 encoding rules, including how characters are encoded into one or more bytes.
   * Understanding the patterns that represent a valid UTF-8 encoded character.
   * [UTF-8 Wikipedia](https://intranet.alxswe.com/rltoken/oqFi6P1hNvp9aSuNv---IQ)
   * [Characters, Symbols, and the Unicode Miracle](https://intranet.alxswe.com/rltoken/d--jVK8sBSlhkosu7pFzdw)
   * [The Absolute Minimum Every Software Developer Absolutely, Positively Must Know About Unicode and Character Sets](https://intranet.alxswe.com/rltoken/9EwaXVds22dSK3IvF5nNCA)
3. **Data Representation**:
   * How to represent and work with data at the byte level.
   * Handling the least significant bits (LSB) of integers to simulate byte data.
4. **List Manipulation in Python**:
   * Iterating through lists, accessing list elements, and understanding list comprehensions.
   * [Python Lists](https://intranet.alxswe.com/rltoken/TaN91MgmOL80GeOGvmldIw)
5. **Boolean Logic**:
   * Applying logical operations to make decisions within the program.

By studying these concepts and utilizing the resources provided, you will be equipped to tackle the UTF-8 validation project, effectively applying bitwise operations and logical reasoning to determine the validity of UTF-8 encoded data.

## **Additional Resource**

* [Mock Technical Interview](https://intranet.alxswe.com/rltoken/X1lZqipeyegt8pbQ9aXSFQ)

## **Requirements**

### **General**

* Allowed editors: vi, vim, emacs
* All your files will be interpreted/compiled on Ubuntu 20.04 LTS using python3 (version 3.4.3)
* All your files should end with a new line
* The first line of all your files should be exactly #!/usr/bin/python3
* A README.md file, at the root of the folder of the project, is mandatory
* Your code should use the PEP 8 style (version 1.7.x)
* All your files must be executable

## **Tasks**

### **0. UTF-8 Validation**

**mandatory**

Score: 50.0% (*Checks completed: 100.0%*)

Write a method that determines if a given data set represents a valid UTF-8 encoding.

* Prototype: def validUTF8(data)
* Return: True if data is a valid UTF-8 encoding, else return False
* A character in UTF-8 can be 1 to 4 bytes long
* The data set can contain multiple characters
* The data will be represented by a list of integers
* Each integer represents 1 byte of data, therefore you only need to handle the 8 least significant bits of each integer

carrie@ubuntu:~/0x04-utf8\_validation$ cat 0-main.py

#!/usr/bin/python3

"""

Main file for testing

"""

validUTF8 = \_\_import\_\_('0-validate\_utf8').validUTF8

data = [65]

print(validUTF8(data))

data = [80, 121, 116, 104, 111, 110, 32, 105, 115, 32, 99, 111, 111, 108, 33]

print(validUTF8(data))

data = [229, 65, 127, 256]

print(validUTF8(data))

carrie@ubuntu:~/0x04-utf8\_validation$

carrie@ubuntu:~/0x04-utf8\_validation$ ./0-main.py

True

True

False

carrie@ubuntu:~/0x04-utf8\_validation$

**Repo:**

* GitHub repository: alx-interview
* Directory: 0x04-utf8\_validation
* File: 0-validate\_utf8.py

Check submission View results

Copyright © 2024 ALX, All rights reserved.