UTF-8 Validation

A character in UTF8 can be from 1 to 4 bytes long, subjected to the following rules:

1. For 1-byte character, the first bit is a 0, followed by its unicode code.
2. For n-bytes character, the first n-bits are all one's, the n+1 bit is 0, followed by n-1 bytes with most significant 2 bits being 10.

This is how the UTF-8 encoding would work:

```
   Char. number range  |        UTF-8 octet sequence
      (hexadecimal)    |              (binary)
   --------------------+---------------------------------------------
   0000 0000-0000 007F | 0xxxxxxx
   0000 0080-0000 07FF | 110xxxxx 10xxxxxx
   0000 0800-0000 FFFF | 1110xxxx 10xxxxxx 10xxxxxx
   0001 0000-0010 FFFF | 11110xxx 10xxxxxx 10xxxxxx 10xxxxxx
```

Given an array of integers representing the data, return whether it is a valid utf-8 encoding.

Note:
The input is an array of integers. Only the **least significant 8 bits** of each integer is used to store the data. This means each integer represents only 1 byte of data.

Example 1:

```
data = [197, 130, 1], which represents the octet sequence: 11000101 10000010 000000
01.

Return true.
It is a valid utf-8 encoding for a 2-bytes character followed by a 1-byte character
.
```

Example 2:

```
data = [235, 140, 4], which represented the octet sequence: 11101011 10001100 00000
100.

Return false.
The first 3 bits are all one's and the 4th bit is 0 means it is a 3-bytes character
.
The next byte is a continuation byte which starts with 10 and that's correct.
But the second continuation byte does not start with 10, so it is invalid.
```

Subscribe to see which companies asked this question

Solution 1

```
class Solution {
public:
    bool validUtf8(vector<int>& data) {
        int count = 0;
        for (auto c : data) {
            if (count == 0) {
                if ((c >> 5) == 0b110) count = 1;
                else if ((c >> 4) == 0b1110) count = 2;
                else if ((c >> 3) == 0b11110) count = 3;
                else if ((c >> 7)) return false;
            } else {
                if ((c >> 6) != 0b10) return false;
                count--;
            }
        }
        return count == 0;
    }
};
```

written by fight.for.dream original link here

Solution 2

public class Solution {

```
public bool ValidUtf8(int[] data) {
    int bitCount = 0;

    foreach(int n in data){

        if(n >= 192){
            if(bitCount != 0)
                return false;
            else if(n >= 240)
                bitCount = 3;
            else if(n >= 224)
                bitCount = 2;
            else
                bitCount = 1;
        }else if(n >= 128){
            bitCount--;
            if(bitCount < 0)
                return false;
        }else if(bitCount > 0){
            return false;
        }
    }

    return bitCount == 0;
}
```

}

written by lalayangguang original link here

Solution 3

```
class Solution(object):
    def validUtf8(self, data):
        """
        :type data: List[int]
        :rtype: bool
        """
        if len(data) == 0:
            return True
        i = 0
        while i < len(data):
            if data[i] < 128:
                i += 1
            elif data[i] >= 192 and data[i] < 224 and len(data)-i>=2:
                if data[i+1] >= 128 and data[i+1] < 192:
                    i += 2
                else:
                    return False
            elif data[i] >= 224 and data[i] < 240 and len(data)-i>=3:
                if data[i+1] >= 128 and data[i+1] < 192 and data[i+2] >= 128 and
data[i+2] < 192:
                    i += 3
                else:
                    return False
            elif data[i] >= 240 and data[i] < 248 and len(data)-i>=4:
                if data[i+1] >= 128 and data[i+1] < 192 and data[i+2] >= 128 and
data[i+2] < 192 and data[i+3] >= 128 and data[i+3] < 192:
                    i += 4
                else:
                    return False
            else:
                return False
        return True
```

written by AlgoGuruZ original link here