Rivets Arm His Lama Den

Write a Python program that implements a version of the RSA algorithm described below. Input will come from `stdin` and will be in the following format:

```
n
C1, C2, C3, ...
```

where n refers to the modulus in RSA, and the comma-separated values represent individual ciphertexts C, each of which corresponds to an encrypted character.

In general, you will need to implement the following algorithm:

Algorithm 1 Rivets Arm His Lama Den

$n \leftarrow$ the first line of input (from stdin: an integer)
$C \leftarrow$ the second line of input (from stdin: a comma separated list of integers)
$p, q \leftarrow$ factor n as the product of two primes
$z \leftarrow \text{lcm}(p-1, q-1) = ((p-1) * (q-1)) / \gcd(p-1, q-1)$ ▷ Charmichael
$e\_values \leftarrow$ generate all values of e where $1 < e < z$ and that are of the form $2^n + 1$

for all e **in** e_values **do**
    $d \leftarrow$ calculate the modular inverse of e such that $d = e^{-1} (\text{mod } z)$ ▷ naively
    $K_{priv} \leftarrow (d, n)$
    $M \leftarrow$ an empty string initializing the plaintext message
    for all c **in** C **do**
        $m \leftarrow$ decrypt c with K_{priv} and convert to its ASCII character representation
        break if any issues occur with this process ▷ this is the wrong value of e
        $M \leftarrow M + m$
    print M

Structure your output so that it matches mine as illustrated by the following sample run of my program:

```
jgourd@latech:~$ python rivets-arm-his-lama-den.py < ciphertext-1.txt
p=389, q=683
n=265687
z=132308
--
Trying e=3
d=44103
Public key: (3, 265687)
Private key: (44103, 265687)
ERROR: invalid plaintext.
--
```

```
Trying e=5
d=79385
Public key: (5, 265687)
Private key: (79385, 265687)
ERROR: invalid plaintext.
--
Trying e=17
d=46697
Public key: (17, 265687)
Private key: (46697, 265687)
ERROR: invalid plaintext.
--
Trying e=257
d=90093
Public key: (257, 265687)
Private key: (90093, 265687)
ERROR: invalid plaintext.
--
Trying e=65537
d=69585
Public key: (65537, 265687)
Private key: (69585, 265687)
The lady said, "Oh my! You have nice eyes. Are they yours?"
I laughed.
```

Here's another sample run of my program:

```
jgourd@latech:~$ python rivets-arm-his-lama-den.py < ciphertext-2.txt
p=743, q=863
n=641209
z=319802
--
Trying e=3
d=106601
Public key: (3, 641209)
Private key: (106601, 641209)
ERROR: invalid plaintext.
--
Trying e=5
d=127921
Public key: (5, 641209)
Private key: (127921, 641209)
ERROR: invalid plaintext.
--
Trying e=17
d=169307
Public key: (17, 641209)
Private key: (169307, 641209)
ERROR: invalid plaintext.
--
```

```
Trying e=257
d=51019
Public key: (257, 641209)
Private key: (51019, 641209)
ERROR: invalid plaintext.
--
Trying e=65537
d=192295
Public key: (65537, 641209)
Private key: (192295, 641209)
Never attribute to malice that which is adequately explained by stupidity.
(Hanlon's razor)
```

Note that there is no need to filter out invalid plaintexts. Simply generate all candidate plaintexts and output them to `stdout` as shown above.

Notes and Requirements:

- Submit your source code only. I will provide my own ciphertext to test with;
- Read the input from `stdin`;
- Write the output (as illustrated above) to `stdout`;
- Comment your source code appropriately; and
- Appropriately layout your program; e.g., write separate functions to:
  - Factor a number into the product of two primes;
  - Determine if a number is prime;
  - Calculate the greatest common divisor of two numbers;
  - Generate candidate values of e;
  - Naively calculate d, the modular inverse of e; and
  - Decrypt ciphertext C into plaintext M using the private key K_{priv}.

Please, no GUIs. Make this a command line application without frills that I can execute at the command line.