Nick is sending a message to Alex using the Elgamal encryption algorithm.

The plaintext message is nicholas. Converting to numbers using a = 00, b = 01, ..., z = 25; the platintext message becomes 1308 0207 1411 0018.

Alex's public key is:

The prime modulus is p = 2539.
The primitive root is a = 2.
The private key is k = 51. The public key is b = 2^51 mod 2539 = 403.

Nick uses k = 15 to calculate the ephemeral key.

```
PowerMod[403, 15, 2539]
```

```
1794
```

The Elgamal message is (2300, 516) (2300, 664) (2300, 2490) (2300, 1824).
The same ephemeral key 1794 was used for each encryption.
Now assume that one of the plaintext messages is known. Say, that we know that the first message decrypts to plaintext 1308.

516/1308 mod 2539 = the masking key

First, we construct the inverse of 1308 mod 2539.

```
ExtendedGCD[1308, 2539]
```

```
{1, {33, -17}}
```

The inverse of 1308 mod 2539 is 33.

Now we calculate the masking key.

```
Mod[516*33, 2539]
```

```
1794
```

The masking key for all four encryptions is 1794.

Next determine the inverse of the masking key.

```
ExtendedGCD[1794, 2539]
```

```
{1, {593, -419}}
```

The inverse of the masking key is 593.

Now decrypt the remaining ciphertext messages.

```
Mod[664 * 593, 2539]
```

```
207
```

```
Mod[2490 * 593, 2539]
```

```
1411
```

```
Mod[1824 * 593, 2539]
```

```
18
```

We have recovered the plaintext message: 1308 0207 1411 0018.

So, the ephemeral key should change from message to message.
Knowing now that the ephemeral key should change, Nick now chooses the following sequence of exponents to use for constructing the ephemeral keys: 15 23 11 4.

Here is the encryption of 1308.

```
PowerMod[2, 15, 2539]
```

```
2300
```

2300 is the first ephemeral key.
PowerMod[403, 15, 2539] is the first masking key.

```
Mod[1308 * PowerMod[403, 15, 2539], 2539]
```

```
516
```

The first block of encryption is (2300, 516).

Here is the encryption of 0207.

```
PowerMod[2, 23, 2539]
```

```
2291
```

2291 is the second ephemeral key.
PowerMod[403, 23, 2539] is the second masking key.

```
Mod[207 * PowerMod[403, 23, 2539], 2539]
```

```
1378
```

The second block of ciphertext is (2291, 1378).

Here is the encryption of 1411.

```
PowerMod[2, 11, 2539]
```

```
2048
```

2048 is the third ephemeral key.
PowerMod[403, 11, 2539] is the third masking key.

```
Mod[1411 * PowerMod[403, 11, 2539], 2539]
```

```
1535
```

The third block of ciphertext is (2048, 1535).

Here is the encryption of 0018.

```
PowerMod[2, 4, 2539]
```

```
16
```

16 is the fourth ephemeral key.
PowerMod[403, 4, 2539] is the fourth masking key.

```
Mod[18 * PowerMod[403, 4, 2539], 2539]
```

```
1675
```

The fourth block of ciphertext is (16, 1675).

The ciphertext message is (2300, 516) (2291, 1378) (2048, 1535) (16, 1675).

Now we decrypt.

We use the notation (y, z) for the ciphertext.

For each block, we first calculate the inverse of the ephemeral key y mod 2539 and then z*(y^(-1))^n, which should be the plaintext.

```
ExtendedGCD[2300, 2539]
```

```
{1, {818, -741}}
```

The inverse of y is 818.

```
Mod[516 * PowerMod[818, 51, 2539], 2539]
```

```
1308
```

The first plaintext block is 1308.

```
ExtendedGCD[2291, 2539]
```

```
{1, {-215, 194}}
```

```
Mod[-215, 2539]
```

```
2324
```

The inverse of y is 2324.

```
Mod[1378 * PowerMod[2324, 51, 2539], 2539]
```

```
207
```

The second plaintext block is 0207.

```
ExtendedGCD[2048, 2539]
```

```
{1, {393, -317}}
```

The inverse of y is 393.

```
Mod[1535 * PowerMod[393, 51, 2539], 2539]
```

```
1411
```

The third plaintext block is 1411.

```
ExtendedGCD[16, 2539]
```

```
{1, {-476, 3}}
```

```
Mod[-476, 2539]
```

```
2063
```

The inverse of y is 2063,

```
Mod[1675 * PowerMod[2063, 51, 2539], 2539]
```

```
18
```

The fourth plaintext block is 0018.