2. **[5 marks] Cryptography.**

Consider the following symmetric key cryptosystem:

- The **secret key** is a tuple of two integers (n, a) where $a > 0$, $n > 0$, and $\gcd(a, n) = 1$.
- The **plaintext** and **ciphertext** messages are strings, where every character has an ord value $< n$.
- To **encrypt** a plaintext message with secret key (n, a):
  - For each character c in the message, compute ord(c), multiply by a, and take the remainder modulo n. Then convert the integer into a character using chr.
- **Decryption** reverses the encryption process.

(a) **[4 marks]** In the space below, implement the encryption function for this cryptosystem.
Hint: This is very similar to the character-based encryption/decryption algorithms from lecture.

```
def encrypt(secret_key: tuple[int, int], message: str) -> str:
    """Encrypt the message using the given secret key.

    Preconditions:
    - secret_key is in the form (n, a) described above
    - all(ord(c) < secret_key[0] for c in message)
    """
    enc = []
    for c in message:
        encrypted = (ord(c) * a) % n
        enc.append(chr(encrypted))
    return ''.join(enc)
```

(b) **[1 mark]** In the encryption, why did we require that all characters in the plaintext message be $< n$?

If ord(c) ≥ n, then the encrypted characters would repeat for certain characters, making it impossible to decrypt.