Cipher Solver by Nathan Mack

1. Introduction:

- The Cipher Solver program is designed to decrypt texts encrypted using various classical ciphers, including Caesar, Affine, Atbash, Rail-fence, and Vigenère ciphers. It is designed in a fashion that all you input is the ciphertext, and the program brute forces several common cryptographic ciphers with a defined number of keys/variables depending on the cipher. Each output is scored by how much the cleartext resembles English.

2. Functions:

2.1. caesar(ciphertext, shift):

- **Description:**
  - Decrypts a Caesar cipher text.
- **Parameters:**
  - **ciphertext** (str): The text encrypted with Caesar cipher.
  - **shift** (int): The number of positions each letter in the ciphertext is shifted backward in the alphabet.
- **Returns:**
  - str: The decrypted text.

2.2. egcd(a, b):

- **Description:**
  - Calculates the Euclidean Greatest Common Divisor.
- **Parameters:**
  - **a** (int): First integer.
  - **b** (int): Second integer.
- **Returns:**
  - tuple: A tuple containing three integers (g, x, y) where g is the greatest common divisor of a and b, and x, y are integers such that g = ax + by.

2.3. modinv(a, m):

- **Description:**
  - Calculates the Modular Multiplicative Inverse used in the Affine cipher.
- **Parameters:**
  - **a** (int): The first key value used in the Affine cipher.
  - **m** (int): The modulo value.
- **Returns:**
  - int: The modular multiplicative inverse.

2.4. affine(ciphertext, a, b):

- **Description:**
  - Decrypts an Affine cipher text.
- **Parameters:**
  - **ciphertext** (str): The text encrypted with Affine cipher.
  - **a** (int): The first key value used in the Affine cipher.
  - **b** (int): The second key value used in the Affine cipher.
- **Returns:**

    - str: The decrypted text.

2.5. atbash(ciphertext):

- **Description:**
    - Decrypts an Atbash cipher text.
- **Parameters:**
    - **ciphertext** (str): The text encrypted with Atbash cipher.
- **Returns:**
    - str: The decrypted text.

2.6. railfence(ciphertext, key):

- **Description:**
    - Decrypts a Rail-fence cipher text.
- **Parameters:**
    - **ciphertext** (str): The text encrypted with Rail-fence cipher.
    - **key** (int): The number of rails used in the Rail-fence cipher.
- **Returns:**
    - str: The decrypted text.

2.7. vigenere(ciphertext, key):

- **Description:**
    - Decrypts a Vigenère cipher text.
- **Parameters:**
    - **ciphertext** (str): The text encrypted with Vigenère cipher.
    - **key** (str): The keyword used in the Vigenère cipher.
- **Returns:**
    - str: The decrypted text.

2.8. score(input, wordlist):

- **Description:**
    - Grades output by comparing each word with words in a wordlist.
- **Parameters:**
    - **input** (str): The decrypted text.
    - **wordlist** (list): A list of words to compare against.
- **Returns:**
    - int: The score based on the percentage of words found in the wordlist.

2.9. printTop(arr, amt):

- **Description:**
    - Prints the top results based on the score.
- **Parameters:**
    - **arr** (list): List containing decrypted text, algorithm used, key, and score.
    - **amt** (int): The number of top results to print.

3. Variables:

3.1. ctext:

- **Type:** str
- **Description:**
    - The encrypted text entered by the user.

3.2. wordlist:

- **Type:** list
- **Description:**
    - A list of words used to score the decryption results.

3.3. solutions:

- **Type:** list
- **Description:**
    - A list containing decrypted text, algorithm used, key, and score.

4. Main Program:

- **User Input:**
    - **ctext** (str): The encrypted text entered by the user.
- **File Input:**
    - **words.txt**: A text file containing a list of words used to score the decryption results.
- **Output:**
    - Prints the top 5 decrypted texts along with the decryption algorithm used and the score.