Florent Dondjeu Tschoufack

COSC 472-001 11/12

**Cryptography Algorithms Implementation**

**Report:**

This project took about 12 hours to complete using python3. The code is separated into two files, "cryptography.py" contains all of the cryptography algorithms. Caesar, Monoalphabetic and polyalphabetic were implemented with OOP and TTH is a function. "program.py" calls the algorithms and handle errors.

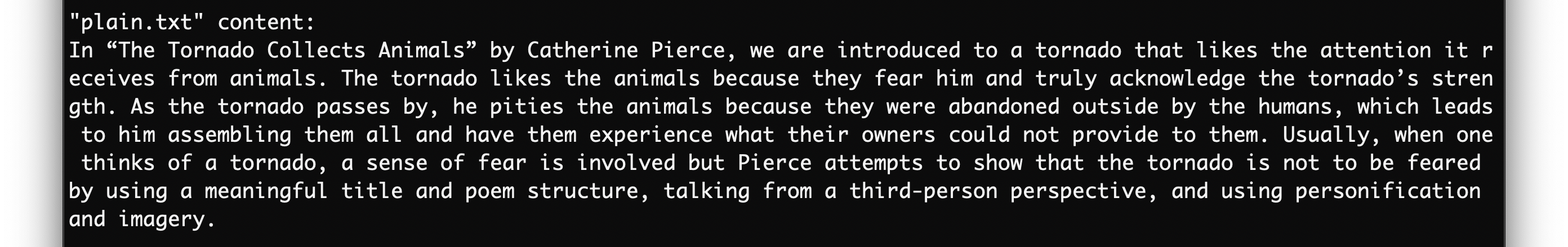
**Run:**

To run the program simply type "python program.py" in a terminal.

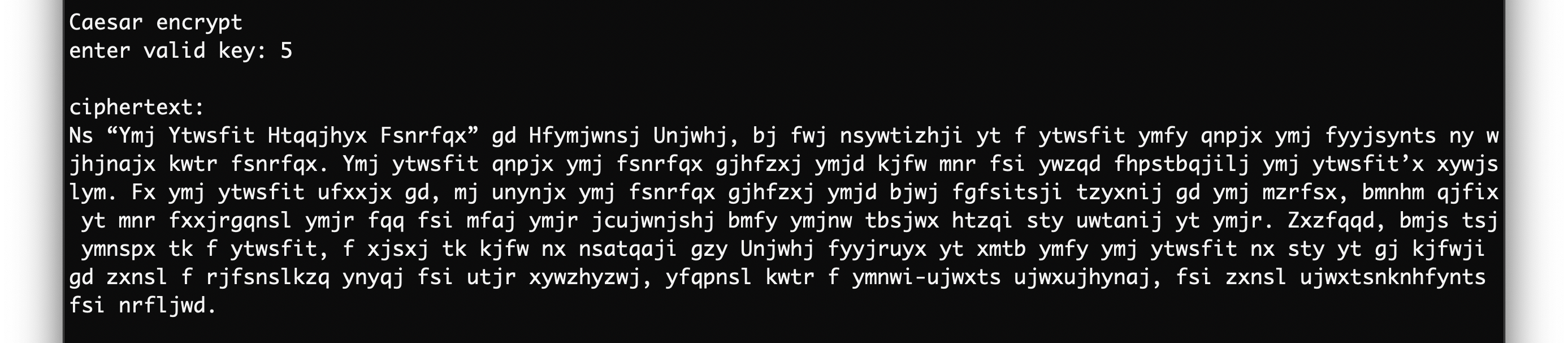
When prompted to enter a file name, ensure that the file or directory containing the file is in the same directory as "program.py".

**Elaboration:**

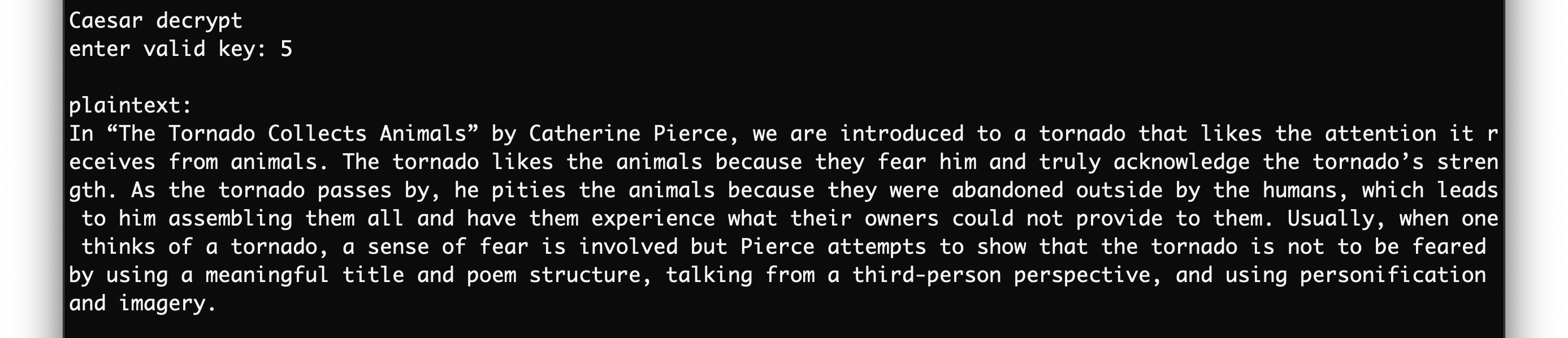
Plaintext used for all algorithms:



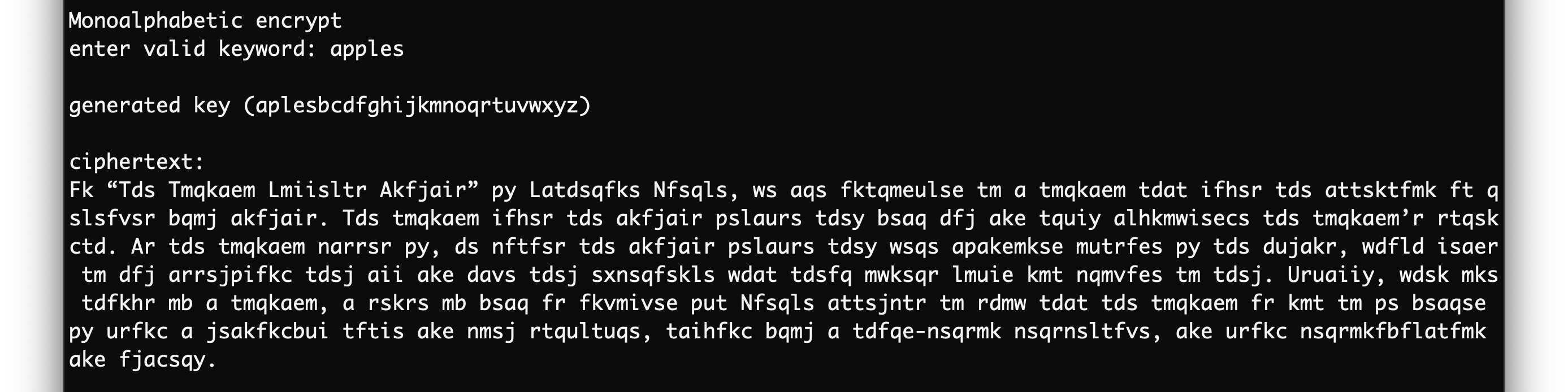
**Caesar:** In order to encrypt and decrypt, the key needs to be of type int. For encrypting, my program reads the plaintext and when it encounters an alphabetic character, it adds the key to the current character which results another alphabetic character that will be part of the ciphertext.



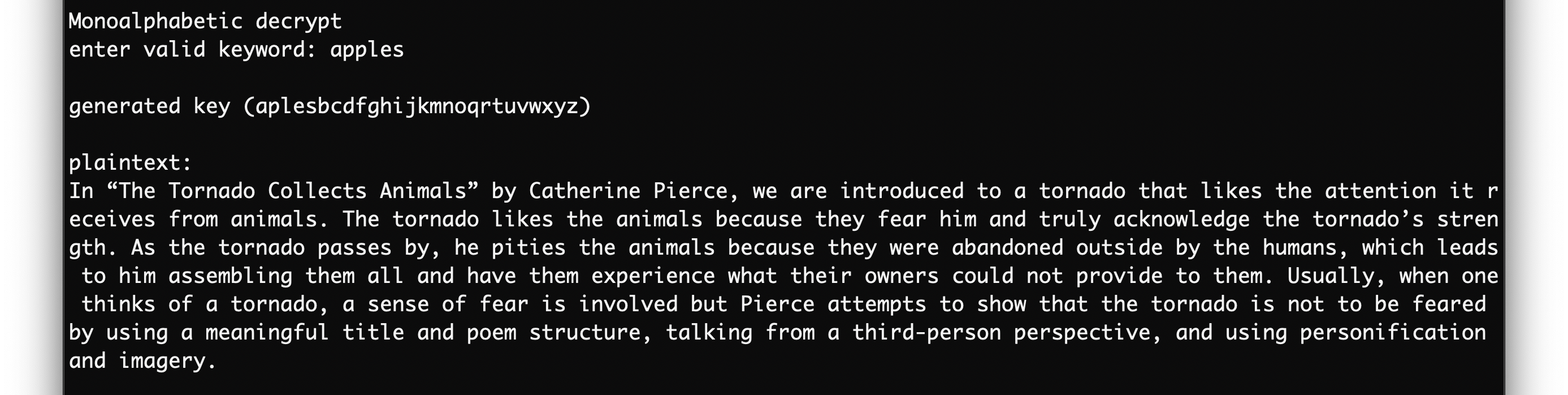
For decrypting, my program reads the ciphertext and when it encounters an alphabetic character, it subtracts the key to the current character which results another alphabetic character that will be part of the original plaintext assuming the correct key was given. Using the ciphertext above:



**Monoalphabetic:** In order to encrypt and decrypt, they key needs to be of type char or string. After the user enters the key, any duplicate alpha characters will be removed and the alpha characters missing will be added. The key used was “apples”. To encrypt, each alpha character is mapped to the corresponding letter in the key according to their position.

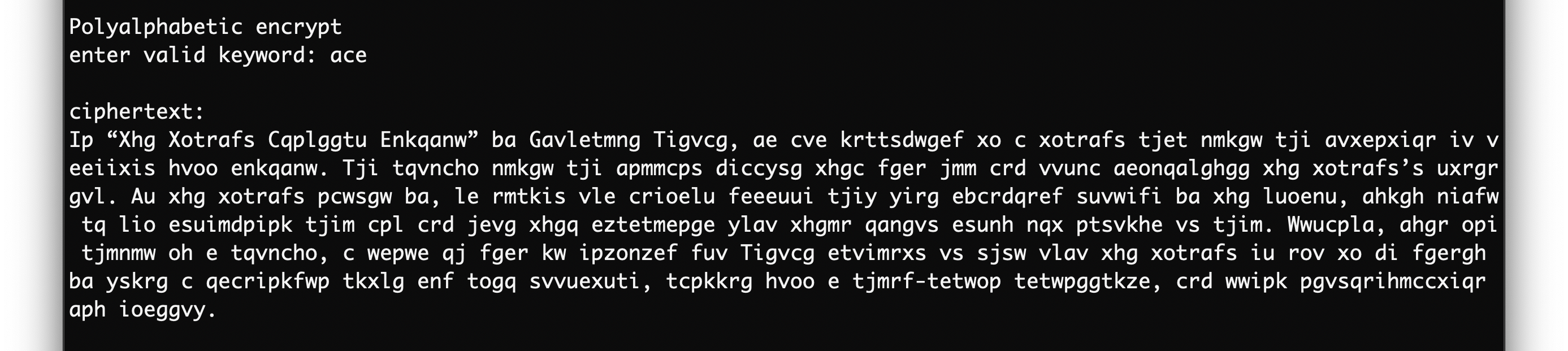


Same procedure for decrypt. Using ciphertext above:

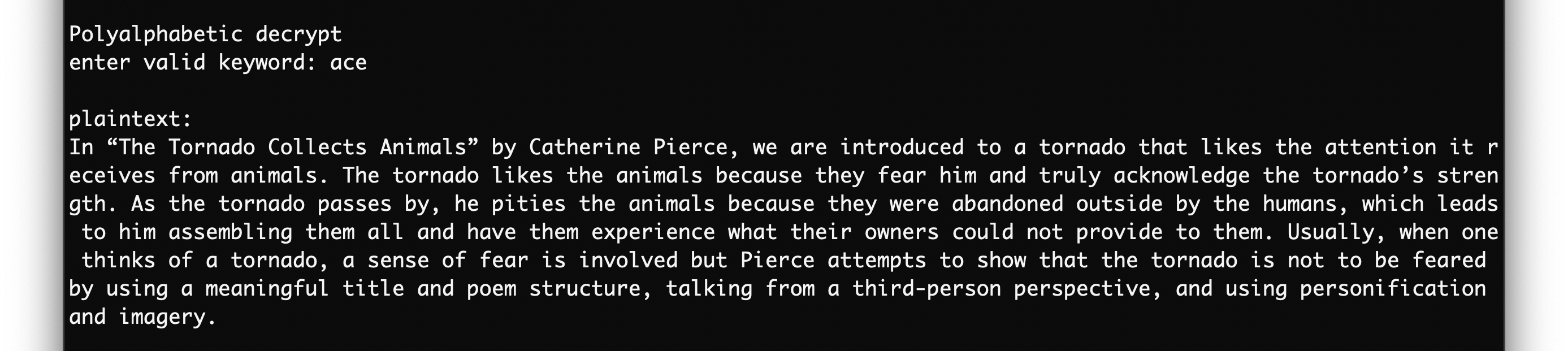


**Polyalphabetic:** In order to encrypt and decrypt, they key needs to be of type char or string.

To encrypt, my program will increase the length of the key to be equal to the length of the plaintext. For every alpha character read with its according alpha character from the key, it will perform a Caesar.



Same procedure for decrypt. Using the ciphertext above:



**TTH:** In order to hash a string, the program initially removes any non-alphabetic characters. Then it divides the new string into blocks of 16 characters and pads the last block with zeros if necessary. Then each block is feed into the compression algorithm. After the last iteration, the running four total is converted to char, resulting the hash value of the given string.

