CST8234 – C Programming
Encryption Algorithm

> Cryptography or cryptology; from Greek κρυπτός kryptós, "hidden, secret"; and γράφειν graphein, "writing", or -λογία-logia, "study", respectively is the practice and study of techniques for secure communication in the presence of third parties.
>
> Wikipedia

Basic Encryption Algonquin

In this encryption algorithm, each byte of *plaintext* (unencrypted) data has been adjusted arithmetically by adding an "encryption key" value to the byte. An 8-bit byte of data can have values that range from 0-255 (expressing the 256 8-bit patterns from 00000000 to 11111111).

To produce the *ciphertext*, the encryption key value is added to each *plaintext* byte. Where the resulting sum exceeds the maximum value allowed in a byte (the sum is greater than 255), the newly calculated number will be "wrapped around" to the start of the range of available numbers.

The following chart shows a sample of several *plaintext* data byte values going through the conversion to *ciphertext* using an example encryption key of 5. For example, the *plaintext* letter 'A' (byte value 65) has the encryption key 5 added to it to become the *ciphertext* letter 'F' (byte value 70).

Note that even if the unencrypted *plaintext* input data is all printable characters, the resulting *ciphertext* will likely contain many data byte values that are not printable characters. For example, a *plaintext* letter 'A' encrypted using a key of 207 would generate a *ciphertext* byte of 65+207-256=16 and 16 does not correspond to any standard printable character. (Some operating systems assign private non-standard printable glyphs to every character value; so, you may or may not see anything print on your screen for such *ciphertext* bytes.)

***Plaintext* Character**	***Plaintext* Byte Value**	**Key**	***Ciphertext* Byte Value**	***Ciphertext* Character**
A	65	5	70	F
B	66	5	71	G
C	67	5	72	H
a	97	5	102	f
b	98	5	103	g
c	99	5	104	h
z	122	5	127	⌂
space	32	5	37	%
1	49	5	54	6
2	50	5	55	7
3	51	5	56	8
þ	254	5	3	♥
ÿ	255	5	4	♦

You are to write a small C program `cipher`, to encrypt / decrypt a file. The program should have the following functionality:

```
cipher [ OPTIONS ] SOURCE DESTINATION
```

OPTIONS:

`d` KEY

decrypt the file SOURCE using KEY and writes back into DESTINATION

`e` KEY

encrypt the file SOURCE using KEY and writes back into DESTINATION

In case that the user does not give all the required command arguments, you program should print a usage message and exit with EXIT_FAILURE.

Now that you have finished your this encryption algorithm, use your **`cipher`** to decrypt the file **`mystery_key`** using the file's size as the key.