C Programming Project #1 - Encryption

Background

One of the simplest forms of encryption is the **Caesar cipher** which shifts each character in a message by a predetermined amount. The message is decrypted by shifting each character back by the same amount. However, this type of encryption is trivial to break since there are only a finite number of shift values and they can each be checked by brute force even without a computer.

```
  Msg: [ T ][ H ][ I ][ S ][ I ][ S ][ A ][ T ][ E ][ S ][ T ]
ASCII: [ 84][ 72][ 73][ 83][ 73][ 83][ 65][ 84][ 69][ 83][ 84]
  Key: [ K ][ K ][ K ][ K ][ K ][ K ][ K ][ K ][ K ][ K ][ K ]
ASCII: [ 75][ 75][ 75][ 75][ 75][ 75][ 75][ 75][ 75][ 75][ 75]
Result: [159][147][148][158][148][158][140][159][144][158][159]
```

A more advanced method of encryption is the **Vigenère Cipher** where instead of a single shift value a key word is used. Each character in the message is encrypted by shifting the character by an amount determined by a character in the key word. When the key word is exhausted it is recycled and we start at the beginning.

```
  Msg: [ T ][ H ][ I ][ S ][ I ][ S ][ A ][ T ][ E ][ S ][ T ]
ASCII: [ 84][ 72][ 73][ 83][ 73][ 83][ 65][ 84][ 69][ 83][ 84]
  Key: [ K ][ E ][ Y ][ K ][ E ][ Y ][ K ][ E ][ Y ][ K ][ E ]
ASCII: [ 75][ 69][ 89][ 75][ 69][ 89][ 75][ 69][ 89][ 75][ 69]
Result: [159][141][162][158][142][172][140][153][158][158][153]
```

Even this approach has flaws due to the repeating nature of the key. An **autokey cipher** incorporates the message itself as the key. In this case, like a Vigenère Cipher we use a key word (which could be just a single letter). When we exhaust the key we begin using the characters of the message itself as key values.

```
  Msg: [ T ][ H ][ I ][ S ][ I ][ S ][ A ][ T ][ E ][ S ][ T ]
ASCII: [ 84][ 72][ 73][ 83][ 73][ 83][ 65][ 84][ 69][ 83][ 84]
  Key: [ K ][ E ][ Y ][ T ][ H ][ I ][ S ][ I ][ S ][ A ][ T ]
ASCII: [ 75][ 69][ 89][ 84][ 72][ 73][ 83][ 73][ 83][ 65][ 84]
Result: [159][141][162][167][145][156][148][157][152][148][168]
```

Requirements

1. Write a C program that reads a string from stdin and encrypts it using the **autokey cipher** described above. Call the executable `encrypt`.
2. Write a second C program that reads a string from stdin and decrypts it using the same method. Call the executable `decrypt`.
3. OPTIONAL: Use preprocessor directives to allow either an encrypt/decrypt program to be built using the same source file. A makefile is provided that will automatically build this way if you call your C file proj1.c.
4. Your program should reflect good design principles and should be resistant to malicious input (meaning that it should be able to handle any input gracefully).

Submission

- Create a *neatly formatted* document (.doc, .txt, or .pdf) with the following:
  1. A brief overview of your programs structure
  2. Your encrypt & decrypt functions
  3. Sample output (please be clear & concise)
- Zip the directory that contains your project and the makefile. Your project will be tested automatically so be sure that you include **all code** required to build.
- Submit the document and your code **separately** to Canvas.

Grading Rubric:

1. Encryption/decryption functions – 35 points
2. Valid output & error handling – 25 points
3. Submission (including document with all 3 items) - 15 points.