Jim Vargas
CS162 Programming Assignment 2
Algorithm and Flowchart

Algorithm

The program is intended to function as a simple encryption procedure. Hereon, let all instances of 'message' refer to the message which is to be transmitted, let 'block' refer to the paragraph of text in which the message is encrypted into, and let 'seed' refer to the process by which the message is encrypted with the block. These topics will be further defined below.

Basic Structure:

1. Greet user and state program purpose
2. Prompt for the first input, get first input
3. Prompt for second input, get first input
4. Seed the block with the message
5. Display seeded block
6. Repeat

Prompts:

1. First prompt:
   - Greet user (as above).
   - First, prompt for the message. This is to be a single sentence.
   - Second, get the message. Check to see if the input is valid, i.e., not blank, not too long, etc.
2. Second prompt:
   - Prompt for the block. This is to be a paragraph, i.e., multiple sentences.
   - Get the block. Check to see if the input is valid, as above.
3. Third prompt:
   - Display encrypted message. Prompt the user to see if they wish to complete another cycle of the process.

Seeding:

The first letter in each word in the block is replaced by a letter in the message. This replacement occurs sequentially; the first word in the block gets the first letter in the message, the second the second, etc. Additionally, the first letter of evey word in the message following the first (thus, not including the first), will be capitalized when replacing the corresponding letter in the block. The block's length, when returned having been seeded by the message, will be shortened to include only the necessary number of words to sufficienty seed the block.

Here is an example:
message: "where does poop go"
unseeded block: "You may think this message is odd, but it is likely more original than this silly block."
seeded block: "wou hay ehink rhis eessage Ds odd, eut st Ps oikely oore Priginal Ghen ohis"

Seeding Algorithm:

1. Loop through all characters in the block.
2. If the current character in the block is one which the preceding character was a 'space,' the current character is viable to be replaced.
3. For each character in the message, convert each viable character in the block with a message character, sequentially.
4. If the current character in the message is one which the preceding character was a 'space,' the current character will be capitalized in the block. That is, when a character in the block is replaced by a character in the message which followed a 'space,' it will be capitalized.

Here is a pseudo-code example of this algorithm.

Algorithm 1 Seeding Algorithm

```
procedure SEED
    given: message
    given: block
    new data (List of Integers): viablePositions
    for each character in block do
        if current character position − 1 is a 'space' or current position is zero then
            add current position to viablePositions.
    for iterate length of viablePositions do
        if current character position − 1 in message is a 'space' then
            block @ viablePosition @ current iteration ← uppercase: message @ current iteration
        else
            block @ viablePosition @ current iteration ← message @ current iteration
```

In the second **for** loop, the algorithm uses layered indexing, using the '@' symbol. Notice, if there aren't an equal or greater number of words in the block as there are characters in the message, then the length of *viablePositions* defined in the algorithm shall not be of sufficient length to fully seed the block with every word.