**CS REPORT**

**NOTABLE OBSTACLES**

1. **UNDERSTANDING THE TASK**

In the beginning, it was hard to start/work on the project because it was initially confusing to understand. The concept itself is pretty confusing to comprehend in the beginning.

1. **CLEANING THE CRIB STRING**

This part was difficult because there were different rules for strings with miscellaneous characters in front; as opposed to just having one way of deleting characters (getting them and replacing them all with one space), I had to use different commands to get rid of characters before and after the first letters.

1. **MAPPING FOR THE DECRYPTION**

It was difficult trying to create a key that would decrypt the cipher text. It took extra steps to make sure that the key would still be valid if a character was already there or if the key (in analyzing the cipher) ran into a character that was not a letter.

1. **SORTING THROUGH MANY LOOPS**

While my code worked, it was hard to understand my code sometimes because my variable-naming wasn’t the most explicit.

1. **SPLITTING UP THE TASK INTO MANAGEABLE FUNCTIONS**

Most of my confusion came from the fact that I kept mixing up functions and thinking that performed oppositely of how I wanted them to. It ultimately took clearer labeling to fix this problem.

**PSUEDOCODE**

Get the desired ciphertext and crib and plug it into the decrypt

Make modifiable duplicates of ciphertext and crib

Clean the strings of undesired characters for analysis

Clean for extra spaces, weird characters, make lower case

If the crib we use is empty, return false

Else – clean the crib of undesired characters

Repeatedly:

Look for the correct key

Start at the beginning of a word

Check from that pos that syntax is correct

If correct, use that start pos for a key

If not, look for the pos of the next word

If there is no pos where syntax is right, return -1;

Check that key values are not trying to replace each other

If so, stay in loop, wrong key

Find the correct key (Decryption)

Make a duplicate ciphertext for decryption

All letters in the key – toupper

Use the letters in the key to decrypt the message

Go to the pos in the key

Replace the letter in the cipher with key letter

Rollover changes to the ciphertext for edit.

Print cipher with decrypted characters.

**DATA TO USE**

**decrypt(cipher, phrase);**

**cipher: “”**

**crib: “abcdf”**

**Reason:** Should come back false (crib cannot be empty)

**cipher: “my\nsecret”**

**crib: “my secret”**

**Reason:** Should return false, syntax not exactly the same

**Cipher: “zr fewyl”**

**crib: “my love”**

**Reason:** Should return false; letters can’t follow the crib

**cipher: “szr fewy”**

**crib: “my love”**

**Reason:** Should return false; letters can’t be in front of crib

**cipher: “jk aj fd”**

**crib: “dog”**

**Reason:** Should return false; word is not found in the ciphertext

**cipher: “tfjitfjitfjitfji”**

**crib: “my”**

**Reason:** Should return false; word can’t follow crib.

**cipher: “tf ji tf ji tf ji tf ji”**

**crib: “my”**

**Reason:** Should return true; there are spaces.

**cipher: “TF JI TF JI TF JI TF JI”**

**crib: “my”**

**Reason:** Should also return true but with only decrypted values in caps.

**cipher: “TFQ$%^%$^ JI”**

**crib: “my”**

**Reason:** Can distinguish miscellaneous characters from letters.

**cipher: “xy ”**

**crib: “my”**

**Reason:** Returns true, no characters on the end.

**cipher: “#%#$%$”**

**crib: “my”**

**Reason:** Should return false, nothing can be decrypted.

**cipher: “abcde fgh”**

**crib: “ ”**

**Reason:** Doesn’t print. Nothing to decipher.