

1 Introduction

As it states on the syllabus, projects are to be done on your own and as with all assignments the sharing of files and code is strictly prohibited and constitutes an act of Academic Misconduct. Furthermore, the use of any electronic medium, such as code repositories, forums, blogs, message boards, email, etc. is strictly prohibited and constitutes an act of Academic Misconduct. You are not to discuss the project with anyone other than myself. You may use the resources that are on the COSC 117 course web page for this class in MyClasses.

As usual, you will submit all your work through the MyClasses page for this class, under Project #2. Make sure you do the formatting Shift+Ctrl+F before you submit your work. **All you need to submit are the Java code files for the program. Make sure you submit all of the java files in the project.**

2 Project

Cryptography is defined to be the process of creating ciphers such that when applied to a message it hides the meaning of the message. The basic idea is for person *A* to have a readable message, encrypt it into something that looks like gibberish, send it to person *B* who can then decrypt it into the original message. By sending the gibberish instead of the actual message protects the message if someone would intercept it as it is traveling from person *A* to person *B*. As long as the person intercepting it does not know the key to decrypt it.

One field cipher that was used by British forces in the Second Boer War and in World War I and by the British and Australians during World War II was created by the scientist and inventor Charles Wheatstone. A field cipher is one that was used literally in the field of battle by soldiers while they were fighting the enemy. These were, of course, encrypted and decrypted by hand. This cipher was good enough for WWI but was far too weak for WWII, the age of cipher machines like the Enigma, PURPLE, SIGABA, and the M-209 to name just a few.

Here is how it works. Person *A* and person *B* decide on a keyword before sending any messages. The keyword could also be several words or a phrase. The word is altered as follows, all spaces are removed, all J's are converted to I's, then any duplication of characters is removed. We also only use uppercase text for letters in both the keyword and the message. There is no advantage to using both uppercase and lowercase.

At this point the word is written in a 5×5 grid, starting in the upper left cell and moving across the first row, then the second and so on until the characters of the keyword are exhausted. The remaining cells in the grid are filled with the remaining letters of the alphabet (again, I and J are the same) in alphabetical order.

For example, the keyword COMMITTEE would be converted to COMITE and then C O M I T would be in the first row and E would be in the first cell in the second row. From there, we continue with A, B, D, F, and so on. When finished we get the following grid,

C	O	M	I	T
E	A	B	D	F
G	H	K	L	N
P	Q	R	S	U
V	W	X	Y	Z

The plaintext message is altered as well. The spaces and punctuation are removed, the message is converted to all uppercase, and the J's are converted to I's. In addition, the method we are using

to encrypt the message cannot have two of the same letters in a row. So for example, the word COMMITTEE would need to be altered. One method, that we will use here, is to put an X between any two letters that are the same. So COMMITTEE would be converted to COMXMITXTEXE. This can happen between words as well when we remove the spaces. For example, if the words THE ENTRY were in the plaintext then when the spaces are removed we have THEENTRY which would need to be changed to THEXENTRY. Also, the method we will be using encrypts two letters at a time, so the plaintext must have an even number of characters. If the plaintext has an odd number of characters we simply add an X to the end of it to make it even. This should be done after the spaces and punctuation are removed and the extra X's are inserted to separate repeated letters. Once the plaintext has been appropriately modified and the encryption/decryption grid has been created we are ready to encrypt or decrypt.

When encrypting the plaintext there are three different cases, depending on the digram (two letters) that is being encrypted.

1. If the letters of the plaintext digram are not in the same row or column. In this case we use the plaintext digram character positions as the corners of a rectangle and encrypt the digram with the other two corners, using the entry in the same row as the first plaintext character for the first ciphertext character and using the entry in the same column as the second plaintext character for the second ciphertext character.

For example, if we encrypt TH using this grid, we find the T and H in the grid, shown in red, and we take the other two corners, shown in blue. The O is on the same row as T so O is the first ciphertext character and N is on the same column as H so it becomes the second ciphertext character. Hence TH is encrypted as ON.

C	O	M	I	T
E	A	B	D	F
G	H	K	L	N
P	Q	R	S	U
V	W	X	Y	Z

Similarly, HT would encrypt as NO, PL would encrypt as SG, GI as LC and so on.

2. If the letters of the plaintext digram are in the same row. In this case we take the letters directly to the right of the plaintext characters, wrapping around to the far left character of the same row if the plaintext character is at the far right.

For example, to encrypt GL, which is on row 3, we shift the G to H and the L to N, so GL encrypts to HN.

C	O	M	I	T
E	A	B	D	F
G	H	K	L	N
P	Q	R	S	U
V	W	X	Y	Z

Similarly, AF would encrypt as BE, SU would encrypt as UP, CI as OT and so on.

3. If the letters of the plaintext digram are in the same column. In this case we take the letters directly below the plaintext characters, wrapping around to the top character of the same column if the plaintext character is at the bottom of the column.

For example, to encrypt BR, which is on column 3, we shift the B to K and the R to X, so

BR encrypts to KX.

C	O	M	I	T
E	A	B	D	F
G	H	K	L	N
P	Q	R	S	U
V	W	X	Y	Z

Similarly, AW would encrypt as HO, TU would encrypt as FZ, LS as SY and so on.

The decryption process is simply a reversal of the encryption process. The process breaks into the same three different cases depending on the positions of the ciphertext characters.

1. If the letters of the ciphertext diagram are not in the same row or column. In this case we use the ciphertext diagram character positions as the corners of a rectangle and encrypt the digram with the other two corners, using the entry in the same row as the first ciphertext character for the first plaintext character and using the entry in the same row as the second ciphertext character for the second plaintext character. Note that this is exactly like the encryption process.
2. If the letters of the ciphertext diagram are in the same row. In this case we take the letters directly to the left of the ciphertext characters, wrapping around to the far right character of the same row if the ciphertext character is at the far left.
3. If the letters of the ciphertext diagram are in the same column. In this case we take the letters directly above the ciphertext characters, wrapping around to the bottom character of the same column if the ciphertext character is at the top of the column. Finally, we would concatenate all of the plaintext digrams together.

For this project you are to use the following main program with no alterations and you are to produce the needed methods.

```

1 public static void main(String[] args) {
2     Scanner keyboard = new Scanner(System.in);
3
4     System.out.print("Input the Keyword: ");
5     String keyword = keyboard.nextLine();
6     keyword = ModifyKeyword(keyword);
7
8     System.out.print("Input the Plaintext: ");
9     String plaintext = keyboard.nextLine();
10    plaintext = ModifyPlaintext(plaintext);
11
12    System.out.println("Modified Keyword: " + keyword);
13    System.out.println("Modified Plaintext: " + plaintext);
14
15    char grid[][] = BuildGrid(keyword);
16
17    System.out.println("Cipher Grid");
18    Print2DArray(grid);
19
20    // Encrypt
21    String ciphertext = Encrypt(plaintext, grid);
22    System.out.println("Ciphertext: " + ciphertext);
23
24    // Decrypt
25    String newplaintext = Decrypt(ciphertext, grid);
26    System.out.println("Decrypted ciphertext: " + newplaintext);
27 }

```

- **public static void** Print2DArray(**char** A[][])

Print2DArray is to print out a two-dimensional array of characters to the screen,

- **public static** String ModifyKeyword(String keyword)

ModifyKeyword is to take the keyword or phrase the user types in and alter it as described above so that it is ready to place in the encryption/decryption grid.

- **public static** String ModifyPlaintext(String plaintext)

ModifyPlaintext is to take the plaintext the user types in and alter it as described above so that it is ready to be encrypted.

- **public static char**[][] BuildGrid(String keyword)

BuildGrid is to take the altered keyword from ModifyKeyword and load it into the 5×5 grid as described above. This method should also fill in the rest of the grid so that it is ready to be used in the encryption/decryption process.

- **public static** String Encrypt(String plaintext, **char**[][] grid)

Encrypt is to take the altered plaintext from ModifyPlaintext and the grid from BuildGrid and encrypt the plaintext as described above.

- **public static** String Decrypt(String ciphertext, **char**[][] grid)

Decrypt is to take the ciphertext from the Encrypt process and the grid from BuildGrid and decrypt the ciphertext as described above.

Two sample runs of the program are below.

```
Input the Keyword: committee
Input the Plaintext: Attack at dawn
Modified Keyword: COMITE
Modified Plaintext: ATXTACKATDAWNX
Cipher Grid
C O M I T
E A B D F
G H K L N
P Q R S U
V W X Y Z
Ciphertext: FOZMEOHBFHOKZ
Decrypted ciphertext: ATXTACKATDAWNX
```

```
Input the Keyword: Jack MacGiever
Input the Plaintext: The gold is buried under the oak tree twenty five paces after the end of the stone wall.
Modified Keyword: IACKMGEVR
Modified Plaintext: THEGOLDISBURIEDUNDERTHEOAKTREXETWENTYFIVEPACESAFTERTHEXENDOFTHESTONEWALXLX
Cipher Grid
I A C K M
G E V R B
D F H L N
O P Q S T
U W X Y Z
Ciphertext: QNVESDOGTRYGAGOIDFVBQNGPCMSBVWBPAFTZWLCGFWCKRPEPPBBSFVWVDFPDQNRPOPFBAEHYHY
Decrypted ciphertext: THEGOLDISBURIEDUNDERTHEOAKTREXETWENTYFIVEPACESAFTERTHEXENDOFTHESTONEWALXLX
```