Specifications Document:

Features of Cipher Main Implementation:

Message Error Checking:

Purpose:

The checkMessage function ensures that the user-entered message is valid, i.e. contains only letters and spaces, and then converts it to all uppercase letters for ease of encryption.

Assumptions:

A specific cipher has been chosen and the user has selected either ‘Encrypt’ or ‘Decrypt.’

Inputs:

There are no inputs for the function as all needed inputs are gathered within the function itself.

Outputs:

The user-chosen message is saved into the file ‘Message.txt ’ in all uppercase letters and the message is returned in all uppercase as a string.

State Changes:

The entered string ‘message’ is transformed into all uppercase letters and the file ‘Message.txt’ will change to contain the user’s input message.

Cases:

The message error-checking function occurs after the user has selected a cipher and have either chosen to ‘Encrypt’ or ‘Decrypt.’ The message is then checked to ensure it contains only letters and spaces, otherwise the user is re-prompted.

Expected Behavior:

When checkMessage is run, the user is prompted to enter a message. That message is then transformed to all uppercase and is checked to make sure it only contains letters and spaces. If the message contains anything other than letters and spaces, the user is given an error message and is re-prompted. This repeats until a valid message is given which is then stored in ‘Message.txt,’ and returned to the main function.

Features of One-Time Pad Implementation:

Encryption Mode:

Purpose:

Encryption mode allows the user to have their message encrypted and displayed without the message having ‘Decrypt’ run on it and displayed as well.

Assumptions:

Encrypt mode requires that the message and key contain only letters and spaces, and that they are the same length.

Inputs:

The Encrypt function takes in the user-submitted ‘message’ and ‘key’ as strings.

Outputs:

The encrypted message is returned as a string, output to the terminal, and saved in the Encrypt.txt file.

State Changes:

The text file ‘Encrypt.txt’ should change to hold the new encrypted message.

Cases:

When the main file is run, the user can choose Encrypt, Decrypt, or test. Based on the user selection, that method is then run. For Encrypt, the message is checked to make sure it is only letters and spaces, as well as the key. The length of the key is also checked to ensure it is the same length as the message.

Expected Behavior:

After ‘Encrypt’ has been run, the encrypted message should be listed in all uppercase letters below the original message and key in the terminal. The file ‘Encrypt.txt’ should contain the encrypted message in all uppercase letters and no changes should have been made to the ‘Decrypt.txt’ file.

Decryption Mode:

Purpose:

Decryption mode allows the user to have their message decrypted and displayed without the message having ‘Encrypt’ run on it and displayed as well.

Assumptions:

Decrypt mode requires that the message and key contain only letters and spaces, and that they are the same length.

Inputs:

The Decrypt function takes in the user-submitted ‘message’ and ‘key’ as strings.

Outputs:

The decrypted message is returned as a string, output to the terminal, and saved in the Decrypt.txt file.

State Changes:

The text file ‘Decrypt.txt’ should change to hold the new decrypted message.

Cases:

When the main file is run, the user can choose Encrypt, Decrypt, or Test. Based on the user selection, that method is then run. For Encrypt, the message is checked to make sure it is only letters and spaces, as well as the key. The length of the key is also checked to ensure it is the same length as the message.

Expected Behavior:

After ‘Decrypt’ has been run, the decrypted message should be listed in all uppercase letters below the original message and key in the terminal. The file ‘Decrypt.txt’ should contain the decrypted message in all uppercase letters and no changes should have been made to the ‘Encrypt.txt’ file.

Testing Encrypt/Decrypt:

Purpose:

The Encrypt and Decrypt testing function allows the user to run predefined messages and keys through both Encrypt and Decrypt to test encryption and decryption accuracy.

Assumptions:

The mode is set to “TEST” and the messages and keys are already set. For encryption testing, the message is set to “HELLO WORLD” and the key to “TESTENCRYPTION”. For decryption testing the message is set to “ICDRDYEO W” and the key to “HELLO WORLD”.

Inputs:

The testing function does not take any variables in.

Outputs:

The encrypted and decrypted message is returned as a string, output to the terminal, and saved in the Encrypt.txt file and Decrypt.txt file respectively. A “PASS” or “FAIL” message is then output to the terminal depending on if the encryption and decryption succeeded or not.

State Changes:

The text files ‘Encrypt.txt’ and ‘Decrypt.txt’ should change to hold the new encrypted and decrypted message respectively.

Cases:

When the main file is run, the user can choose Encrypt, Decrypt, or Test. Based on the user selection, that method is then run. For Test, the message and key are pre-set and do not change.

Expected Behavior:

After Test has been selected, function ‘test _main’ runs, subsequently running test Encrypt and Decrypt with preset values. The encrypted message should be printed to the terminal along with a “PASS” or “FAIL” message. Then below, the decrypted message should be printed to the terminal along with another “PASS” or “FAIL” message. The files ‘Encrypt.txt’ should contain the encrypted message in all uppercase letters and ‘Decrypt.txt’ should contain the decrypted message in all uppercase letters.