Jacob Faulk

So I can’t submit my Sourcecode using Canvas (due to it not being an acceptable filetype), so I have put it on Sage under the filename “EnigmaEmulator.sagews”. I have also put it here in its entirety.

```
# Jacob Faulk

def main():
    # plaintext = input("Welcome to the Enigma Machine emulator. Enter plaintext to be 
encrypted.\n")

    # Enter plaintext here
    plaintext = "Lorem ipsum dolor sit amet, consectetur adipiscing elit."
    plaintext = plaintext.upper()
    print("Plaintext: " + plaintext)

    print("The first step in encoding a message in the Enigma involves a plugboard.")
    print("This would be set every day and sent out to the German soldiers in WWII.")
    print("For this example, the plugboard will be set as follows:")

    alphabet = list("ABCDEFGHIJKLMNOPQRSTUVWXYZ")
    plugboard_locations = list("ETWAYDSVFPRCQJXOGUHZNLIMBK")
    plugboard = dict(zip(alphabet, plugboard_locations))

    print_rotor(plugboard)

    print("In this example, " + alphabet[0] + " corresponds to " + plugboard_locations[0] +
          ", " + alphabet[1] + " corresponds to " + plugboard_locations[1] + ", and so on.\n")

    print("The Enigma would convert your first letter, " +
          plaintext[0] + ", to " + plugboard[plaintext[0]] + ".\n\n")

    # Note that the print statements will be expanded on in the future, for this rough draft, I 
will just work on the internals of the Enigma
    rotor_1 = dict(zip(list("HLKEGUYWRDCNTBFVQIZPMXSAJO"),
                       list("GWHDISFZYJATEPCLVUNXMQRKOB")))
    rotor_2 = dict(zip(list("XRUPTZEFDSHCMNOJQWLYVGIABK"),
                       list("OPTNYFGUERBQSZWAHJMVLKDXIC")))
    rotor_3 = dict(zip(list("THXIJYKMZDAOWVSEQFBPUNRGCL"),
                       list("VYCKSURPTLNQBMJHDEFXOWGZAI")))

    rotor_1_backup = rotor_1.copy()
    rotor_2_backup = rotor_2.copy()
    rotor_3_backup = rotor_3.copy()

    reflector = dict(zip(list("ZEFUHBDMNIJGACVTQRWYXOSLPK"),
                         list("GBURKVPSFHJTMXAIEZLNWODQYC")))

    final_result = ""

    for character in plaintext:
        if character in alphabet:
            current = character
            print("Current character: " + current)
            current = plugboard[current]
            print("Turned into " + current + " by the plugboard.")
            current = rotor_1[current]
            print("Turned into " + current + " by the first rotor.")
            rotor_1 = rotate_rotor(rotor_1)
            print("First rotor rotated.")

            # this rotates the other rotors when the first one makes a complete rotation
            if rotor_1 == rotor_1_backup:
                rotor_2 = rotate_rotor(rotor_2)
                print("Second rotor rotated.")
                if rotor_2 == rotor_2_backup:
                    rotor_3 = rotate_rotor(rotor_3)
                    print("Third rotor rotated.")

            current = rotor_2[current]
            print("Turned into " + current + " by the second rotor.")
            current = rotor_3[current]
            print("Turned into " + current + " by the third rotor.")

            # now the letter is reflected back through the same rotors
            current = reflector[current]
            print("Turned into " + current + " by the reflector.")

            # TODO make reverse versions of the rotors
            current = rotor_3[current]
            print("Turned into " + current + " by the third rotor.")
            current = rotor_2[current]
            print("Turned into " + current + " by the second rotor.")
            current = rotor_1[current]
            print("Turned into " + current + " by the first rotor.")
            rotor_1 = rotate_rotor(rotor_1)
            print("First rotor rotated.") # TODO see if this is accurate
            if rotor_1 == rotor_1_backup:
                rotor_2 = rotate_rotor(rotor_2)
                print("Second rotor rotated.")
                if rotor_2 == rotor_2_backup:
                    rotor_3 = rotate_rotor(rotor_3)
                    print("Third rotor rotated.")

            # TODO make reverse version of the plugboard
            current = plugboard[current]
            print("Turned into " + current + " by the plugboard.")
            print("Final character encryption: " + current + "\n\n")
            final_result += current
        else:
            final_result += character


    print("The final result is: " + final_result)
    # TODO decryption
    # TODO final presentation of the ciphertext

def rotate_rotor(rotor):
    keys = list()
    values = list()
    for key, value in rotor.items():
        keys.append(key)
        values.append(value)

    # rotating the rotor
    first_char = values[0]
    for i in range(len(values) - 1):
        values[i] = values[i + 1]
    values[len(values) - 1] = first_char

    return dict(zip(keys, values))

def print_rotor(rotor):
    for key in rotor.keys():
        print(key + ' ', end='')
    print()
    for value in rotor.values():
        print(value + ' ', end='')
    print()

if __name__ == "__main__":
    main()
```

"Enigma Machine." 2019. In Wikipedia. https://en.wikipedia.org/w/index.php?title=Enigma_machine&oldid=891536713.

Hern, Alex. 2014. "How Did the Enigma Machine Work?" The Guardian, November 14, 2014, sec. Technology. https://www.theguardian.com/technology/2014/nov/14/how-did-enigma-machine-work-imitation-game.

"How Enigma Machines Work." n.d. Accessed April 10, 2019. http://enigma.louisedade.co.uk/howitworks.html.

"How the Enigma Works." n.d. Accessed March 17, 2019. https://www.pbs.org/wgbh/nova/article/how-enigma-works/.

NOVA. n.d. Decoding Nazi Secrets. Accessed March 17, 2019. https://www.youtube.com/watch?v=DyfBDKtqU6Y.