Enigma Class Usage

Hopefully this class set up should work with the calling of a minimal number of functions. Though all functions are available to be called, I wouldn't recommend it unless you've made a copy of the program. As of now, there are no exceptions handled for out of bounds inputs. Anything out of bounds will either cause garbage output or crash the program.

You need to create a Class object first:

machine = Enigma()

All class function call descriptions will be made in relation to 'machine'. If you want to call the object something else just replace all instances of 'machine' with whatever name you want.

Functions needed to encrypt/decrypt:

machine.setKey(*some string variable* **)**

This is the most important string. To create an acceptable string that will properly setup all the bells and whistles it needs to be in this format:

1st rotor	2nd rotor	3rd rotor	1st rotor setting	2nd rotor setting	3rd rotor setting	# of plugs	1st pair of letters	2nd pair of letters	And so on
(1-3)	(1-3)	(1-3)	(00-25)	(00-25)	(00-25)	(0-9)	(a-z)	(a-z)	

A proper key would be: 3210203033abcdef

3210203033abcdef
'3/2/1' – first 3 characters are the rotor order. They need to be different numbers

3210203033abcdef
'02/03/03 – these numbers are the rotor settings of each rotor. They do not need to be different

3210203033abcdef
'3' – number of plugs to attach

3210203033abcdef
'ab/cd/ef' – since the number of plugs is 3, you need 3 pairs of letters to connect. Do not repeat letters

This will reset the Enigma machine back to its original settings every time a new key is set. Then it will apply this key to the original settings of the Enigma machine.

machine.prepareText(*some string variable* **)**

This is where the cypher/plain text is entered

machine.newText

Where the output of the encryption/decryption is stored.

machine.resetEnigma()

This will reset the Enigma machine WITHOUT resetting the key. Will allow for incoming and outgoing messages to be encrypted with the same key.