Girls' Programming Network

Cryptography!

Extensions

7. Extension: Random Keys?

So far we have had to choose a number to shift the letters by. Now, we want to let the computer choose what number to use by typing ‘random’ when you ask how many letters to rotate by.

Task 7.0: Let’s get random!

First we need to `import random` at the top of our program!

Task 7.1: Letters or numbers?

Now, when we ask the user for a key they can either type a number (the key) or the word “random”.

1. Add an if statement to check if the key they entered is the word `"random"` before the rest of your program.
2. Also add an else so that if they put in a number, the key is set to that number!

Hint

Don’t forget to use `int()` when you set the key to be the number

Task 7.2: So random

If the input is ‘`"random"` the computer should pick a random number and store it in the key variable.

Write code that chooses a random number between 1 and 25 and stores it in the key variable.

Hint

To get a random number between 1 and 10 you would write it like this:

```
number = random.randrange(1, 11)
```

Task 7.1: Write it down!

We need to know what the random number is so that we can give it to our friends to decode the message later. Add a print statement that prints the key before the secret message.

8. Extension: While loop

What if we have heaps of messages to encrypt and decrypt? Let's loop our whole program so that we can use it for heaps of messages!

Task 8.1: Whiling away the while

To make the code run over and over and over again so that we can keep encrypting all of our secret messages, we want to add a while loop near the top of the program.

1. Add a `while True` loop to your program!

Hint

Remember that you only want to put things you want to do EVERY TIME inside the while loop, so printing out the welcome statement and making your alphabet variable should be before the loop.

Remember while loops look like this:

```
while <SOME CONDITION>:
    statement
```

For our loop to run continuously, we can use `while True:`

Task 8.2: Add new line

Did you notice that `Do you want to encrypt or decrypt` doesn't appear on a new line when it runs in a loop? Look at the text in red in the below box.

```
Welcome to the Caesar Cipher!
Do you want to encrypt or decrypt (e/d): e
What is your message: Hello
How many letters to rotate by: 4
ippsoDo you want to encrypt or decrypt (e/d):
```

1. Use and empty print statement to print a blank line at the end of each loop

Hint

We want it to appear like this:

```
Welcome to the Caesar Cipher!
Do you want to encrypt or decrypt (e/d): e
What is your message: Hello
How many letters to rotate by: 4
ippso
Do you want to encrypt or decrypt (e/d):
```

Task 8.3: How to stop the loop?

Our while loop is running over and over and over forever. Let's make it stop when we want.

If the user enters a blank line we want to end the program.

1. After your input message line, create an `if` statement.
2. The `if` statement should check if the message entered is an empty string, `""`
3. If it is, then we want to `break`. We do this by writing `break` inside our `if` statement.
4. Don't forget to indent your `break`.

9. Extension: Reading Files

If you want to send and receive secret messages from your friends you'd want to use a file to store them. We are going to get your program to read the file and encrypt your message, without you having to type it!

Task 9.1: Save your file.

1. Download the file message.txt from http://bit.ly/gpn-crypto1
2. Make sure you save it in the same folder as your python file.

Task 9.1: No need to ask

Since we are getting the message from a file, we don't need to ask the user for a message. Comment out the input statement that gets the message from the user.

Task 9.2: What's inside?

Now we need to access what is inside the file.

1. At the beginning of your code (but after any imports you've added) we want to `open` the file
2. Read the 1 line that is there
3. Strip the extra space from the end
4. Make sure you store the message in a variable called `message`
5. Print out the message and see if it's what was in the file

Hint

To open and read a line from a file you can do this:

```
with open('file.txt') as f:
    line = f.read()
    line = line.strip()
```

Here we also used strip to remove the new line from the end of the message.

Task 8.1: Encode your message!

You have a message in your variable again. Your code should just work when you run it now!

1. See if it decrypts the message from the file.

10. Extension: Writing Files

Now we want to send a secret message to a friend, we need to put our message in a file!

Task 10.1: Storing your secret message

Instead of just printing out our message as we go we want to build up the encrypted message in a variable so we can store it in a file at the end.

1. Before your for loop create a variable called `encrypted_message`, set it to the empty string `""`.

2. You currently print out the encrypted letter, or print out the regular character if it wasn't in the alphabet. We need to replace both of these lines. Instead you need to add the character you would have printed to the end of `encrypted_message`.

3. To test that your code still works print out your `encrypted_message` after your for loop ends.

Hint

You can add to a string and overwrite what was there before like this:

```
name = "harry"
last_name = "potter"
name = name + last_name
```

Task 10.2: Putting it in a file

Now we need to make a file and put our message in!

1. Go to where you tested printing your `encrypted_message`, we're going to put our file code here.

   Use this line to make a file that we can write to (that's what `'w'` does):
   ```
   with open('output.txt', 'w') as f:
   ```

2. Inside the `with` statement you can write to the file with this line:
   ```
   f.write(encrypted_message)
   ```
   Remember to indent!

3. Run your code and see if a file is created with your encrypted message in it!