Python Encrypted Messages

destination exploration

1. **It's time to send secret messages! First, we'll have to build an encryption system to hide our messages. Let's start by asking a) what that message is and b) who is sending it and then assigning each answer to the variables** `message` **and** `numkey` **respectively**.

```
message = input ("Enter your message, your secret is safe with me: ")
numkey = input("What is your name? ")
```

2. **Build the bank of letters that may be in a message as a *string*. These are all the possible letter options in a message including capitals, spaces, numbers and punctuation.** *This string has a length of 71*

```
message = input ("Enter your message, your secret is safe with me: ")
numkey = input("What is your name? ")

bank = "ABCDEFGHIJKLMNOPQRSTUVWXYZ abcdefghijklmnopqrstuvwxyz,'?!.$*&0123456789"
```

3. **To create the encryption, we'll use the length of the sender's name as the key. Make a variable named *key* that is the length of the *numkey* string by using** *`len()`.*

```
message = input ("Enter your message, your secret is safe with me: ")
numkey = input("What is your name? ")

bank = "ABCDEFGHIJKLMNOPQRSTUVWXYZ abcdefghijklmnopqrstuvwxyz,'?!.$*&0123456789"

key=len(numkey)
```

4. **Create an empty *string* called** `encrypt` **to hold the new encrypted message.**

```
message = input ("Enter your message, your secret is safe with me: ")
numkey = input("What is your name? ")

bank = "ABCDEFGHIJKLMNOPQRSTUVWXYZ abcdefghijklmnopqrstuvwxyz,'?!.$*&0123456789"

key=len(numkey)

encrypt = ""
```

When encrypting any message, we are generally replacing one letter with a specific new letter based on a key or cypher. For this simple encryption, we are going to replace the original letter with the letter *key* (say 5) **positions away from it in our** *bank*.

For example: Say the letter we are trying to replace is "c". Our key is based on the sender's name, "Billy" therefore the length of this string is 5. Our new encrypted letter will be 5 positions further in our bank or alphabet that the original. *c → (d e f g) → h.* Our new encrypted letter is "h".

Python Encrypted Messages

5. **To change each letter in a *string*, we can *index* (`i`) or look up each part of the *string* and preform a few actions before moving on the next letter. This is done in a `for` loop.**

```
message = input ("Enter your message, your secret is safe with me: ")
numkey = input("What is your name? ")

bank = "ABCDEFGHIJKLMNOPQRSTUVWXYZ abcdefghijklmnopqrstuvwxyz,'?!.$*&0123456789"

key=len(numkey)

encrypt = ""

for i in message:       # for each position or letter in the string "message"
                        # do the following:
```

6. **For each letter or character in the message, we are first going to find its position within our *bank*. If our message is "**Science is cool**", first look at** S **= 19 position,** c = **30 position etc.**

```
message = input ("Enter your message, your secret is safe with me: ")
numkey = input("What is your name? ")

bank = "ABCDEFGHIJKLMNOPQRSTUVWXYZ abcdefghijklmnopqrstuvwxyz,'?!.$*&0123456789"

key=len(numkey)

encrypt = ""

for i in message:
      position = bank.find(i)
```

7. **Use the *key* to create the position of the new letter.** The *%71* is to account for letters at the end of the *bank* where we would have to continue counting from the beginning again.

```
message = input ("Enter your message, your secret is safe with me: ")
numkey = input("What is your name? ")

bank = "ABCDEFGHIJKLMNOPQRSTUVWXYZ abcdefghijklmnopqrstuvwxyz,'?!.$*&0123456789"

key=len(numkey)

encrypt = ""

for i in message:
      position = bank.find(i)
      newposition = (position + key) % 71
```

destination exploration

8. **Finally we have to add the letter in this new position** `bank[newposition]` **to our** `encrypt = “”` **string by using** `+=`**. Remember that these 3 steps must be indented under the** `for` **loop.**

```
message = input (“Enter your message, your secret is safe with me: “)
numkey = input(“What is your name? “)

bank = “ABCDEFGHIJKLMNOPQRSTUVWXYZ abcdefghijklmnopqrstuvwxyz,’?!.$*&0123456789”

key=len(numkey)

encrypt = “”

for i in message:
      position = bank.find(i)
      newposition = (position + key) % 71
      encrypt += bank[newposition]
```

9. **Use the** *`print()`* **function to display the new encrypted message to the user.**

```
message = input (“Enter your message, your secret is safe with me: “)
numkey = input(“What is your name? “)

bank = “ABCDEFGHIJKLMNOPQRSTUVWXYZ abcdefghijklmnopqrstuvwxyz,’?!.$*&0123456789”

key=len(numkey)

encrypt = “”

for i in message:
      position = bank.find(i)
      newposition = (position + key) % 71
      encrypt += bank[newposition]

print(“Your encrypted message is: “ +encrypt)
```

The first part is now done! We have an encrypted message. Now let’s get that message to the right person. For the decryption code, we are going to do the almost the same thing. Can you give it a try?

You’ll have to make new variables for the decryption positions and change the math in the for loop. Try it out before you jump to the next page.

Python Encrypted Messages

destination exploration

10. Let's look at just the decryption code for now. First, we have to make sure that only the right person can see the decrypted message. Let's ask what the magic number is. Tell the user the trick (the number of letters in your name)**. We will use this for our new decryption key or** `dkey`. **Make sure you turn it from a** `string` **to an** `integer` **so we can use it in our future equation.**

```
print("\n \n Are you ready for your message?")      # \n will give us a new line
dkey = input("Then what is the magic number? ")

dkey = int(dkey)
```

11. Create the variable `decrypt` **and make it an empty string like the original** `encrypt` **variable.**

```
print("\n \nAre you ready for your message?")      # \n will give us a new line
dkey = input("Then what is the magic number? ")

dkey = int(dkey)

decrypt = ""
```

12. Create a similar `for` **loop for the decryption. We'll use the same bank but will have to make slightly different variable names for the encrypted message positions.**

```
print("\n \nAre you ready for your message?")      # \n will give us a new line
dkey = input("Then what is the magic number? ")

dkey = int(dkey)

decrypt = ""

for i in encrypt:
      pos = bank.find(i)
      newpos = (pos - dkey) % 71
      decrypt += bank[newpos]
```

12. Now all you have to do is print the decrypted message for your user!

```
print("\n \nAre you ready for your message?")      # \n will give us a new line
dkey = input("Then what is the magic number? ")

dkey = int(dkey)

decrypt = ""

for i in encrypt:
      pos = bank.find(i)
      newpos = (pos - dkey) % 71
      decrypt += bank[newpos]

print(numkey+ " wishes to tell you... " +decrypt)
```

Python Encrypted Messages

destination exploration

Putting the two parts together,

your final code should look like this:

```
message = input ("Enter your message, your secret is safe with me: ")
numkey = input("What is your name? ")

bank = "ABCDEFGHIJKLMNOPQRSTUVWXYZ abcdefghijklmnopqrstuvwxyz,'?!.$*&0123456789"

key=len(numkey)

encrypt = ""

for i in message:
      position = bank.find(i)
      newposition = (position + key) % 71
      encrypt += bank[newposition]

print("Your encrypted message is: " +encrypt)

##########################################################

print("\n \n Are you ready for your message?")
dkey = input("Then what is the magic number? ")

dkey = int(dkey)

decrypt = ""

for i in encrypt:
      pos = bank.find(i)
      newpos = (pos - dkey) % 71
      decrypt += bank[newpos]

print(numkey+ " wishes to tell you... " +decrypt)
```

Right now, anyone looking at the screen can see the original message. Can you make it so the user would have to input the encrypted message into the program? This way your friends can open up the "Decrypter Program" and input the message and magic number to get their message.

Hint: You'll have to split the code into two programs.