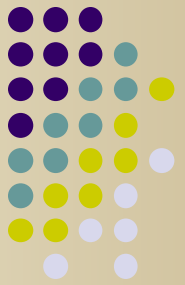
ENGR/CS 101 CS Session Lecture 9

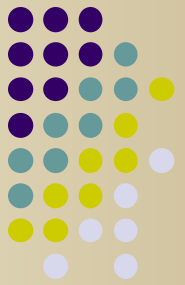


- Log into Windows/ACENET (reboot if in Linux)
- Start Python, open program from last time.
- Has everyone finished the program from last class so that it can encipher and decipher an arbitrary message entered by the user?

Outline

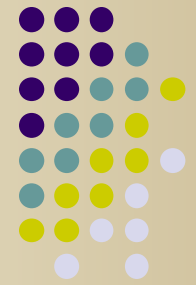
- Finish decoding messages
- Problem: Messages in files
  - Python file objects
- Error checking
- Homework 2

Review: Deciphering



- One of the reasons the Caesar shift cipher is not a good method for keeping secrets is that it is easy to decipher when you know the key.
- We could decipher by subtracting the shift number from the cipherletter index rather than adding it.
- Unfortunately, the % (modulus) operator only works reliably on positive numbers.

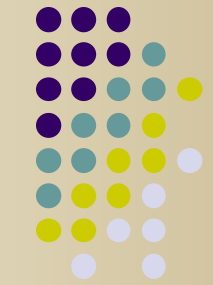
Complementary Shift Number



- But in modular arithmetic, subtracting is the same as adding its complementary number (modulus base minus the number). In our case, this will be 26 minus the shift number.
- For example, shift key 'I', gives a shift number of 8. The complementary shift number for deciphering is 26 - 8 = 18. (This is equivalent to a shift key of 'S'.)

```
>>> encodePlaintext('Ow Ikma!', 'S')
>>> Go Aces!
```

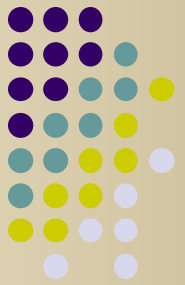
Complementary Shift Number



- This means that we can decipher using the exact same algorithm as for enciphering, except that the shift number computation becomes:

```
shift = 26 - (ord(key) - ord('A'))
```

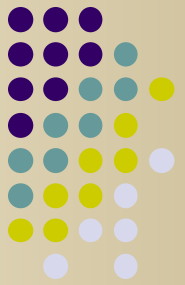
One Function for Both



- Since deciphering is exactly the same as enciphering, we can use the same function as long as we pass in the shift number rather than the shift key.
- First, we move the computation of the shift number from the function to the main program in the if-statement for encrypting.

```
if choice == '1':
    shift = ord(key) - ord('A')
    ...
```

One Function for Both



- Then we change the second parameter of the function to **`shiftNumber`**

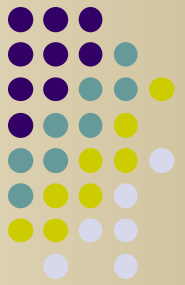
```
>>> encodePlaintext('Ow Ikma!', 18)
>>> Go Aces!
```

- Finally, we change the second argument to the function call to be the shift number (rather than the shift key letter)

```
secret = encodePlaintext(message, shift)
```

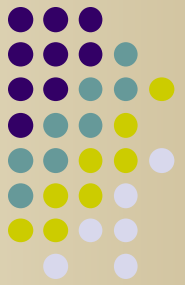
- Run the program. It should behave the same as before.

In-class Exercise, Part 1



- In the main program loop, add a branch to the if-statement to process a choice of '2' to decrypt a secret message. The steps are
  a. Compute the ***complementary*** shift number
  b. Ask the user for a ciphertext message
  c. Compute the plaintext message by calling the function with the secret message and the complementary shift number as arguments
  d. Display the plaintext message

In-class Exercise, Part 1

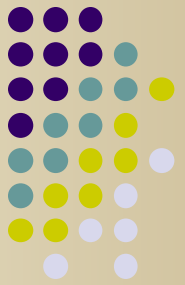


- Run the program and test the decrypt choice

```
Choose an option:
   1. Encrypt a message
   2. Decrypt a message
   3. Quit
Enter your choice: 2
Enter a message to decrypt: Ow Ikma!
The decrypted message is: Go Aces!
```

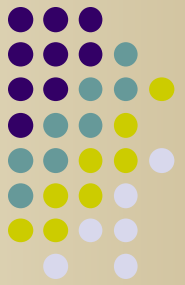
- At this point, the name of the function should be something generic like **`shiftText`**

Problem: Messages in Files



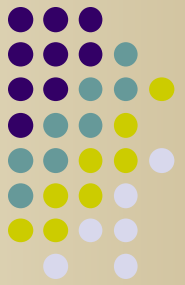
- Typing in long messages is tedious and error prone. Also, message often come in textfiles.
- Keeping a record of the resulting messages would require writing down the messages on a piece of paper, also tedious and error prone.
- We would like to enhance our project to be able to read messages from and write the results to a textfile.

Textfiles



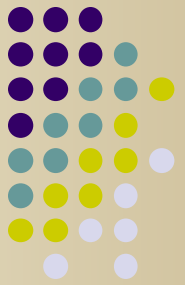
- Like everything else in a computer, the data in files is in the form of a sequence of bytes.
- Textfile data is in ASCII format and generally is considered to be a sequence of lines. The newline character (`'\n'` in Python) marks the end of a line.

Opening a File



- To access data from a file, a program must ***open*** it and associate it with a file object.
- The Python **`open`** function receives the name of the file and a mode, where a mode can be **`'r'`** for reading, **`'w'`** for writing, or **`'a'`** for appending.
- The **`open`** function returns a file object, which is then used to access the data in the associated file.

Reading from a File



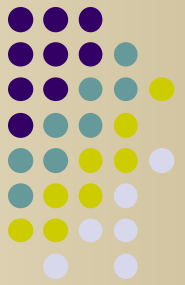
- To read from a file, first we ask the user for a file name and open it with mode **`'r'`**

```
inputFilename = \
   raw_input ('Enter the name of the input file: ')
inputFile = open(inputFilename, 'r')
```

- There are several ways to read data from a file. For this program we'll read the file line by line using a for-loop

```
for line in inputFile: # line includes the newline
    # Do something with line
```

Writing to a File



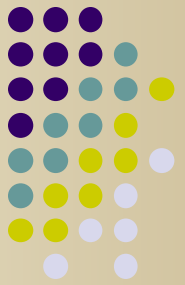
- Similarly, to write to a file, we ask the user for a file name and open it with mode **`'w'`**

```
outputFilename = \
   raw_input ('Enter the name of the output file: ')
outputFile = open(outputFilename, 'w')
```

- To write data to the file do the following

```
outputFile.write (<string>)
```

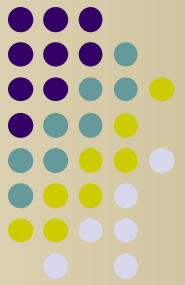
Closing a File



- When a program is done using a file, the file object should be ***closed***, i.e., disassociated from the physical file. This is especially important for output files, since this ensures that all of the data is written onto the storage media.
- In Python, this is done as follows:

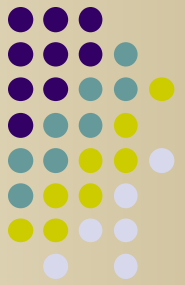
```
inputFile.close()
outputFile.close()
```

In-class Exercise, Part 2



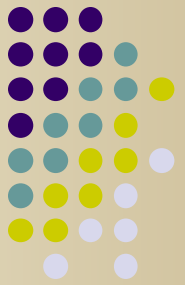
- Save As the current program to a new file.
- Create a New File and Save As with name **`message.txt`**. Make sure it is in the same folder as the program. Type in a message and save the file.

In-class Exercise, Part 2



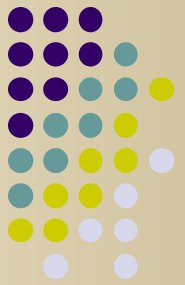
- Modify the main program so that it asks the user for the names of an input file and an output file, then opens them for reading and writing, respectively. This should go inside the loop, but ***before*** the if-statement.

In-class Exercise, Part 2



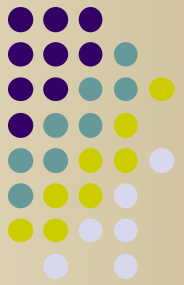
- For each choice in the if-statement, modify the code so that a for-loop reads the lines from the input file. For each line make the appropriate function call, then write the result to the output file (instead of printing it to the screen).

In-class Exercise, Part 2



- Finally, close the input and output files just before getting the user's next choice of action.
- Run the program and test that both encrypting and decrypting work. See sample run on next slide. The new files can be opened with IDLE or any text editor like Notepad.

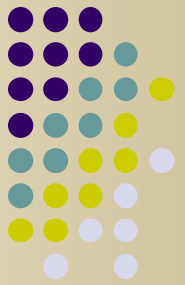
Sample Run (input in bold)



```
Enter a shift key in
uppercase: I
Choose an option:
   1. Encrypt a message
   2. Decrypt a message
   3. Quit
Enter your choice: 1
Enter the name of the input
file: message.txt
Enter the name of the output
file: secret.txt
Choose an option:
   1. Encrypt a message
   2. Decrypt a message
   3. Quit
Enter your choice: 2
```

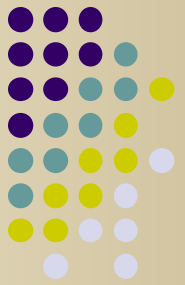
```
Enter the name of the input
file: secret.txt
Enter the name of the output
file: message2.txt
Choose an option:
   1. Encrypt a message
   2. Decrypt a message
   3. Quit
Enter your choice: 3
All done
```

Error handling



- When a user provides an illegal input, a program should try to handle such an error so that the user is informed that the input is illegal and so that the program doesn't provide erroneous results or crash.
- E.g., what happens if the user inputs a non-uppercase alphabetic character for the shift key? What happens when user enters 4 for the menu choice?

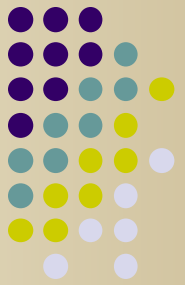
In-class Exercise, Part 3



- Add code to the main program so that if the user does not enter an uppercase letter for the shift key, the program displays an error message and quits

```
Enter a shift key in uppercase: i
The shift key must be in uppercase!
All done
```

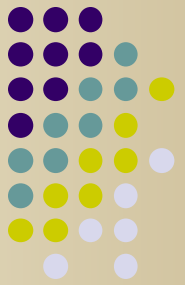
In-class Exercise, Part 3



- Add an else section to the if-statement in the main program loop that displays an error message if the user does not enter 1, 2, or 3

```
Choose an option:
   1. Encrypt a message
   2. Decrypt a message
   3. Quit
Enter your choice: 4
Bad choice, try again
Choose an option:
   1. Encrypt a message
   ...
```

Homework 2



- Also posted to class webpage. Due next Wednesday. Submission system will not be available until Monday.
- KC-267 is open Tuesday, Thursday, and Friday afternoons. Cypherlock code is:

- Friday class will be to work on this program. Instructor will be taking attendance.