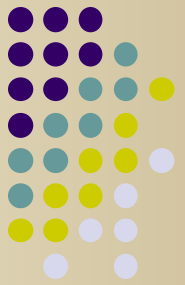
# ENGR/CS 101 CS Session Lecture 7

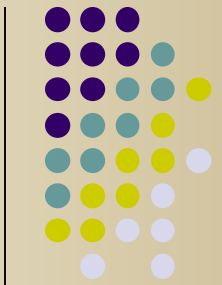


- Pre-registration and guidebooks
- Log into Windows/ACENET (reboot if in Linux)
- Start Python, open program from last time
- Has everyone finished the program from last class so that it can encipher a word in uppercase letters?

# Outline

- Go over Homework 1
- Problem: Input more than one word at a time
- Problem: Enciphering lowercase letters
- New Python construct
  - If-statements

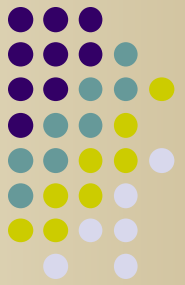
# Problem: Input Sentences



- Modify the cipher program to accept and encipher whole sentences in uppercase letters, rather than just a single word.
- Sample run (input in bold):

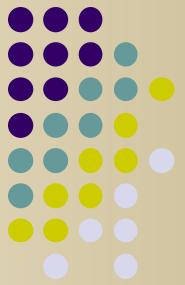
```
Enter a message in uppercase to encrypt (# to quit): GO ACES!
The encrypted message is: OW IKMA!
Enter a message in uppercase to encrypt (# to quit): #
All done
```

# If-Statements



- To prevent the program from enciphering non-uppercase letter characters, we want to execute the shifting code only when the plaintext letter is one of the uppercase alphabet letters.
- We do this with an ***if-statement***.

# If-Statements



- An if-statement has:
  - A condition test
  - A body to execute when the test is true
  - An optional body to execute when the test is false
- We used it to decide whether to encipher a character as follows:

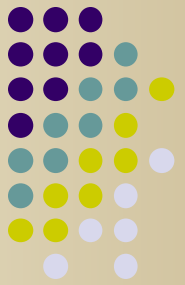
1. If the plaintext letter is an uppercase letter

   a. Compute corresponding ciphertext letter

   Else

   b. Set ciphertext letter to the plaintext letter

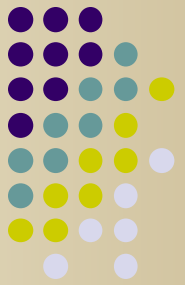
# If-Statements



- The syntax for a Python if-statement is

```
if <condition>:
    # steps to execute if condition is true
else: # this part is optional
    # steps to execute if condition is false
```

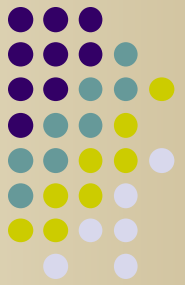
# Conditions



- A ***condition*** is an expression that is either true or false
- They may be formed using operators

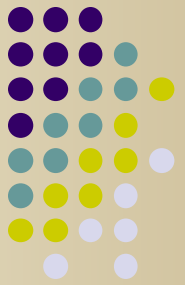
| | |
|---|---|
| `==`, `!=` | equal to, not equal to |
| `<`, `<=` | less than, less than or equal to |
| `>`, `>=` | greater than, greater than or equal to |
| `and` | logical AND - true is both operands are true |
| `or` | logical OR - true if one of the operands is true |
| `not` | logical NOT – true if operand is false and vice versa |

# Conditions



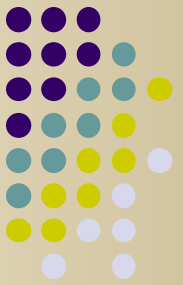
- What would be the condition to test if a number is greater than 10?
- What would be the condition to test if a number is less than 20?
- What would be the condition to test if a number is between 10 and 20 inclusive?
- What would be the condition to test if the plaintext letter is an uppercase letter?

# In-class Exercise, Part 1



- Modify the program to encipher a sentence of uppercase plaintext and display the ciphertext using an if-statement around the cipher letter computation in the function (code shown on the next slide)
- Test your program with the string "GO ACES!" => "OW IKMA!"
- See what happens if you type in "Go Aces!". We'll do lowercase letters next.

# Putting the Code Together

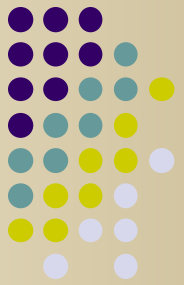


```
# Put an if statement around the ciphertext
# letter computation.  New stuff in bold.

# Compute the shift number
shiftNumber = ord(shiftKey) - ord('A')

# Check for uppercase letter
if ('A' <= plainLetter) and(plainLetter <= 'Z'):
   # Compute the ciphertext letter
   index = ((ord(plainLetter) - ord('A')
                      + shiftNumber) % 26)
   cipherLetter = chr(ord('A')+index)
else:  # do not encipher
   cipherLetter = plainLetter;
```

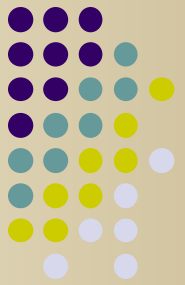
# Problem: Lowercase Letters



- Modify the program to accept and encipher words with both upper and lowercase letters.
- Sample run (input in bold):

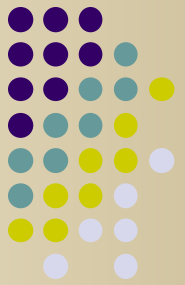
```
Enter a message to encrypt (# to quit): Go Aces!
The encrypted message is: Ow Ikma!
Enter a message to encrypt (# to quit): #
All done
```

# Handling Lowercase Letters



- We'd like the shift of the lower case letters to match that of the uppercase letters.  I.e., if the shift key is A->I, then 'a' should be shifted to 'i'.
- The "formula" for testing for lowercase letters is the same as for the uppercase letters except that 'a' and 'z' are used instead of 'A' and 'Z'.
- Likewise, computing the indexes for lowercase letters must use 'a' rather than 'A'

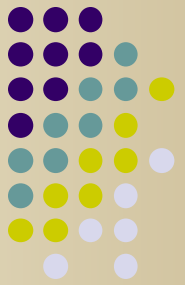
# Handling Lowercase Letters



- We can add another branch to the if-statement in our program and copy/modify the code.

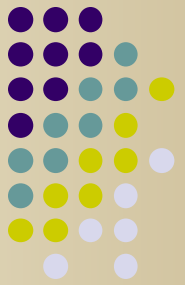
```
if ('A' <= plainLetter) and (plainLetter <= 'Z'):
    # Compute the uppercase ciphertext letter
    # ...
# Check for a lowercase letter
elif ('a' <= plainLetter) and (plainLetter <= 'z'):
    # Compute the lowercase ciphertext letter
    index = ((ord(plainLetter) - ord('a')
                          + shiftNumber) % 26)
    cipherLetter = chr(ord('a') + index)
else:  # do not encipher
    cipherLetter = plainLetter;
```

# Handling Lowercase Letters



- Coding notes
  - The code for doing the shift is the same for both kinds of letters except for the use of 'A' vs. 'a'. We could refactor the code to make this idea into its own function.

# In-class Exercise, Part 2



- Add the code on the previous slide to your program.
- Test your program with the string "Go Aces!" => "Ow Ikma!"

- Next class we'll see how to decipher a Caesar shifted message.