**Presentation Notes**

1. What does the ASCII acronym stand for?

* American Standard Code for Information Interchange

1. What is the ASCII code used for?

* Representing and storing text in computers
* Text numbers must be encoded as numbers

1. Encoding characters (i.e. letters on the keyboard) into ASCII code numbers  
   1. What is the ASCII code for the letter “A” – 65
   2. What is the ASCII code for the letter “a” – 97
   3. Why are they different?

- Upper case and lower case are different symbols. The computer doesn't really know what the alphabet is or how to read and write.

* 1. What is the ASCII code for the space bar?
* The ASCII code also includes some "un-printable" characters

1. Decoding ASCII code numbers into characters and letters   
   1. What character corresponds to ASCII code 61 decimal - =
   2. What character corresponds to ASCII code 8 decimal – back space
   3. Why is the character 8 not the same as ASCII code 8

- Character "8" is text symbol, code 8 is an number. Symbols and numbers are different things to a computer.

* 1. What is the range of non-printable characters in ASCII

- Codes 0 to 31

1. How would you code the string “Hello” in ASCII?  
   H e l l o

72 101 108 108 111

1. How would you code the string “127” in ASCII?  
   1 2 7

49 50 55

1. What is the difference between 127 and “127”?

127 is an integer number. Computers don't need to use ASCII for numbers.  
"127" is a string of text symbols. A human might see this as the number 127. A computer doesn't know it's a number.

**Student Questions**

1. Why do computers have to convert characters (i.e. letters on the keyboard) into numbers? Why can’t computers just use the letters directly?  
   - Representing text. When any key on a keyboard is pressed, it needs to be converted into a binary number so that it can be processed by the computer
2. How do computers communicate with people who speak different languages and use different alphabets? What is used instead of the ASCII code table?

* An assembly language can be used to tell the computer how to do what and when and what the person is saying. The ASCII table tells the computer about what the person is saying. The computer can use some other languages in computer form. Such as uni code and UTF.

1. Research online-documentation for the Python **ord()** function. Provide some sample code that demonstrates the use of the **ord()** function.

* Some function that could be used is value =ord("A")

1. Research online-documentation for the Python **chr()** function. Provide some sample code that demonstrates the use of the **chr()** function.

An sample code for this would be print(chr(71), chr(101),

chr(101), chr(107)

1. Write a Python program that uses the Ord() and chr() functions to do the following:
   1. Read a single character (i.e. single letter or keyboard symbol) from the console input.
   2. Convert the character to an ASCII code number.
   3. Add 3 to the code number.
   4. Convert the new code number back to a character (i.e. single letter or keyboard symbol)
   5. Print the new character to the console output.

The letter I chose is a

chrIn = input ('a')

ordIn = ord(chrIn)

ord0ut = ordIn + 3

chr0ut = chr(ord0ut)

print('The character shifted by 3 is:', chr0ut)

1. Enhance your program to add the following features:
   1. After reading the single character from console input, check to make sure that the character is a letter (i.e. a to z or A to Z). Print a warning message if the character is not a letter.
   2. After converting the code number back to a character, print a “\*” if the character is not a letter.

chrIn = input ('a')

ordIn = ord(chrIn)

ord0ut = ordIn + 3

chr0ut = chr(ord0ut)

print('The character shifted by 3 is:', chr0ut)

**Extension (Optional)**

1. Extend your program to operate on a string read in from the console input.
   1. Use a loop to process the string as a sequence of single characters
   2. Use your original code process the characters
   3. Append the characters to make a new output string
   4. Print the new string to console output