**Presentation Notes**

1. What does the ASCII acronym stand for?
   1. American Standard Code for Information Interchange
2. What is the ASCII code used for?
   1. Representing an d storing text in computers
      1. Computers can only understand numbers (binary)
      2. Text symbols must be written as numbers
   2. Encoding text for electronic communications
      1. Sending and receiving numbers must both agree and understand the same encoding standard
3. Encoding characters (i.e. letters on the keyboard) into ASCII code numbers  
   1. What is the ASCII code for the letter “A”
      1. 65
   2. What is the ASCII code for the letter “a”
      1. 97
   3. Why are they different?
      1. Uppercase and lowercase letters are different symbols and the computer does not understand that they are two letters with the same meaning.
   4. What is the ASCII code for the space bar?
      1. 32
4. Decoding ASCII code numbers into characters and letters   
   1. What character corresponds to ASCII code 61 decimal
      1. =
   2. What character corresponds to ASCII code 8 decimal
      1. Backspace
   3. Why is the character 8 not the same as ASCII code 8
      1. Character "8" is text symbol, code 8 is a number. Symbols and numbers are different things to a computer.
   4. What is the range of non-printable characters in ASCII
      1. Codes 0-31
5. How would you code the string “Hello” in ASCII?
   1. H E L L O
   2. 72 101 108 108 111
6. How would you code the string “127” in ASCII?
   1. 1 2 7
   2. 49 50 55
7. What is the difference between 127 and “127”?
   1. 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?
   1. This happens because the computer does not understand that humans what letters to be displayed onto the screen as it does not understand any human language and only speaks in binary code.
2. How do computers communicate with people who speak different languages and use different alphabets? What is used instead of the ASCII code table?
   1. In order to communicate with humans a computer will convert the ASCII language to letters, numbers, symbols, and punctuation marks so that humans can understand what the computer is to communicate. These characters are within the UFT and Unicode.
3. Research online-documentation for the Python **ord()** function. Provide some sample code that demonstrates the use of the **ord()** function.
   1. The ord() function in Python accepts a string of length 1 as an argument and returns the unicode code point representation of the passed argument. For example, ord('B') returns 66 which is a unicode code point value of character ‘B’.

# inbuilt function return an

# integer representing the Unicode code

value = ord("A")

# writing in ' ' gives the same result

value1 = ord('A')

# prints the unicode value

1. print value, value1 Research online-documentation for the Python **chr()** function. Provide some sample code that demonstrates the use of the **chr()** function.
   1. As you can see above, the chr() function takes a single parameter and returns the corresponding character of the integer ASCII value.
   2. >>> print(chr(98))

B

>>> print(chr(555))

ȫ

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.

myCharacter = input ("Please enter a character ")

print (ord(myCharacter))

myCode = (ord(myCharacter))

print (myCode + 3)

print ("Your new character is:")

print(chr(myCode + 3))

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.

myCharacter = input ("Please enter a character ")

myCode = (ord(myCharacter))

if (127 > myCode or myCode < 65):

print ('warning: this character is not a letter')

print (ord(myCharacter))

print (myCode + 3)

print ("Your new character is:")

print(chr(myCode + 3))

myNewCode = (myCode + 3)

if (myNewCode > 127 or myNewCode < 65):

print ("\*")

**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

myCharacter = input ("Please enter a character ")

index = 0

while (index < 4):

print(ord(myCharacter[index]))

index = index + 1