Inclass assignment 1

Class ID : 26 Vamshi Thatikonda

1. State differences between Python 2 and Python 3 version.

   Print: In Python 2, “print” is treated as a statement rather than a function. There is no need to wrap the text you want to print in parentheses, although we can if we want.
   In contrast, Python 3 explicitly treats “print” as a function, which means you have to pass the items you need to print to the function in parentheses in the standard way, or you will get a syntax error.
   Integer Division – Implicit typecast: Python 2 treats numbers that you type without any digits after the decimal point as integers, which can lead to some unexpected results during division. For example, if you type the expression 3 / 2 in Python 2 code, the result of the evaluation will be 1, not 1.5 as we might expect. This is because Python 2 assumes that we want the result of our division to be an integer, so it rounds the calculation down to the nearest whole number. In order to get the result 1.5, we would have to write 3.0 / 2.0 to tell Python that we want it to return a float, that is, to include digits after the decimal point in the result. Python 3 evaluates 3 / 2 as 1.5 by default.
   List Comprehension Loop Variables: In previous versions of Python, giving the variable that is iterated over in a list comprehension the same name as a global variable could lead to the value of the global variable being changed. However in Python 3, we can use a variable name we already used for the control variable in your list comprehension without worrying about it leaking out and messing with the values of the variables in the rest of our code.
   Unicode Strings: Python 3 stores strings as Unicode by default, whereas Python 2 requires us to mark a string with a “u” if we want to store it as Unicode. Unicode strings are more versatile than ASCII strings, which are the Python 2 default, as they can store letters from foreign languages as well as emoji and the standard Roman letters and numerals.
   Exceptions Syntax:
   Python 3 requires different syntax for raising exceptions. If we want to output an error message to the user, we need to use the syntax:
   raise IOError(“your error message”)

   This syntax works in Python 2 as well. The following code works only in Python 2, not Python 3 :raise IOError, “your error message”