Find 3 differences between a compiler and an interpreter.

Sol:

- The interpreter: Translates program one statement at a time but compiler Scans the entire program and translates it as a whole into machine code.

- The interpreter: It takes less amount of time to analyze the source code but the overall execution time is slower. But the compiler takes large amount of time to analyze the source code but the overall execution time is comparatively faster.

- Programming language like Python, Ruby use interpreters. Programming language like C, C++ use compilers.

Find the difference between Python 2 and 3?

1. PYTHON 2 IS LEGACY, PYTHON 3 IS THE FUTURE.

Since Python 2 has been the most popular version for over a decade and a half, it is still entrenched in the software at certain companies.

However, since more companies are moving from Python 2 to 3, someone who wants to learn Python programming for beginners may wish to avoid spending time on a version that is becoming obsolete.

2. PYTHON 2 AND PYTHON 3 HAVE DIFFERENT (SOMETIMES INCOMPATIBLE) LIBRARIES

Since Python 3 is the future, many of today's developers are creating libraries strictly for use with Python 3.

Similarly, many older libraries built for Python 2 are not forwards-compatible.

You may be able to port a 2.x library to 3.x., but this can be difficult and complicated; it's definitely not a “Python for beginners” type of activity.

3. THERE IS BETTER UNICODE SUPPORT IN PYTHON 3

In Python 3, text strings are Unicode by default. In Python 2, strings are stored as ASCII by default–you have to add a “u” if you want to store strings as Unicode in Python 2.x.

This is important because Unicode is more versatile than ASCII. Unicode strings can store foreign language letters, Roman letters and numerals, symbols, emojis, etc., offering you more choices.

4. PYTHON 3 HAS IMPROVED INTEGER DIVISION

In Python 2, if you write a number without any digits after the decimal point, it rounds your calculation down to the nearest whole number.

For example, if you’re trying to perform the calculation 5 divided by 2, and you type 5 / 2, the result will be 2 due to rounding. You would have to write it as 5.0 / 2.0 to get the exact answer of 2.5.

However, in Python 3, the expression 5 / 2 will return the expected result of 2.5 without having to worry about adding those extra zeroes.

This is one example of how Python 3 syntax can be more intuitive, making it easier for newcomers to learn Python programming.

5. THE TWO VERSIONS HAVE DIFFERENT PRINT STATEMENT SYNTAXES

This is only a syntactical difference–and some may consider it trivial–so it doesn’t affect the functionality of Python. That said, it is still a big and visible difference you should know about.

Essentially, in Python 3, the print statement has been replaced with a print () function.

For example, in Python 2 it is print “hello” but in Python 3 it is print (“hello”).

If you're going to learn Python programming for the first time, it shouldn't affect you much. But if you started with Python 2, the change may trip you up a few times.

What is ASCII and UTF-8?

ASCII defines 128 characters, which map to the numbers 0–127. Unicode defines (less than) 2^{21}characters, which, similarly, map to numbers 0–2^{21} (though not all numbers are currently assigned, and some are reserved).
Unicode is a superset of ASCII, and the numbers 0–128 have the same meaning in ASCII as they have in Unicode. For example, the number 65 means "Latin capital 'A'".

Because Unicode characters don't generally fit into one 8-bit byte, there are numerous ways of storing Unicode characters in byte sequences, such as UTF-32 and UTF-8.