1. Python 3 – What is New?

The __future__ module

Python 3.x introduced some Python 2-incompatible keywords and features that can be imported via the in-built __future__ module in Python 2. It is recommended to use __future__ imports, if you are planning Python 3.x support for your code.

For example, if we want Python 3.x's integer division behavior in Python 2, add the following import statement.

```
from __future__ import division
```

The print Function

Most notable and most widely known change in Python 3 is how the **print** function is used. Use of parenthesis () with print function is now mandatory. It was optional in Python 2.

```
print "Hello World" #is acceptable in Python 2
print ("Hello World") # in Python 3, print must be followed by ()
```

The print() function inserts a new line at the end, by default. In Python 2, it can be suppressed by putting ',' at the end. In Python 3, "end=' '" appends space instead of newline.

```
print x,           # Trailing comma suppresses newline in Python 2
print(x, end=" ")  # Appends a space instead of a newline in Python 3
```

Reading Input from Keyboard

Python 2 has two versions of input functions, **input()** and **raw_input()**. The input() function treats the received data as string if it is included in quotes '' or "", otherwise the data is treated as number.

In Python 3, raw_input() function is deprecated. Further, the received data is always treated as string.

```
In Python 2
>>> x=input('something:')
something:10 #entered data is treated as number
>>> x
10
>>> x=input('something:')
something:'10' #eentered data is treated as string
```

```
>>> x
'10'
>>> x=raw_input("something:")
something:10 #entered data is treated as string even without ''
>>> x
'10'
>>> x=raw_input("something:")
something:'10' #entered data treated as string including ''
>>> x
"'10'"
In Python 3
>>> x=input("something:")
something:10
>>> x
'10'
>>> x=input("something:")
something:'10' #entered data treated as string with or without ''
>>> x
"'10'"
>>> x=raw_input("something:") # will result NameError
Traceback (most recent call last):
  File "", line 1, in

    x=raw_input("something:")
NameError: name 'raw_input' is not defined
```

Integer Division

In Python 2, the result of division of two integers is rounded to the nearest integer. As a result, 3/2 will show 1. In order to obtain a floating-point division, numerator or denominator must be explicitly used as float. Hence, either 3.0/2 or 3/2.0 or 3.0/2.0 will result in 1.5

Python 3 evaluates 3 / 2 as 1.5 by default, which is more intuitive for new programmers.

Unicode Representation

Python 2 requires you to mark a string with a **u** if you want to store it as Unicode.

Python 3 stores strings as Unicode, by default. We have Unicode (utf-8) strings, and 2 byte classes: byte and byte arrays.

xrange() Function Removed

In Python 2 range() returns a list, and xrange() returns an object that will only generate the items in the range when needed, saving memory.

In Python 3, the range() function is removed, and xrange() has been renamed as range(). In addition, the range() object supports slicing in Python 3.2 and later .

raise exceprion

Python 2 accepts both notations, the 'old' and the 'new' syntax; Python 3 raises a SyntaxError if we do not enclose the exception argument in parenthesis.

```
raise IOError, "file error" #This is accepted in Python 2
raise IOError("file error") #This is also accepted in Python 2
raise IOError, "file error" #syntax error is raised in Python 3
raise IOError("file error") #this is the recommended syntax in Python 3
```

Arguments in Exceptions

In Python 3, arguments to exception should be declared with 'as' keyword.

```
except Myerror, err: # In Python2
except Myerror as err: #In Python 3
```

next() Function and .next() Method

In Python 2, next() as a method of generator object, is allowed. In Python 2, the next() function, to iterate over generator object, is also accepted. In Python 3, however, next(0 as a generator method is discontinued and raises **AttributeError.**

```
gen = (letter for letter in 'Hello World') # creates generator object
next(my_generator) #allowed in Python 2 and Python 3
my_generator.next() #allowed in Python 2. raises AttributeError in Python 3
```

2to3 Utility

Along with Python 3 interpreter, 2to3.py script is usually installed in tools/scripts folder. It reads Python 2.x source code and applies a series of fixers to transform it into a valid Python 3.x code.

```
Here is a sample Python 2 code (area.py):
def area(x,y=3.14):
    a=y*x*x
    print a
    return a
```

```
a=area(10)
print "area",a
To convert into Python 3 version:
$2to3 -w area.py
Converted code :
def area(x,y=3.14): # formal parameters
    a=y*x*x
    print (a)
    return a
a=area(10)
print("area",a)
```

2. Python 3 – Overview

Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently whereas the other languages use punctuations. It has fewer syntactical constructions than other languages.

- **Python is Interpreted:** Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.

- **Python is Interactive:** You can actually sit at a Python prompt and interact with the interpreter directly to write your programs.

- **Python is Object-Oriented:** Python supports Object-Oriented style or technique of programming that encapsulates code within objects.

- **Python is a Beginner's Language:** Python is a great language for the beginner-level programmers and supports the development of a wide range of applications from simple text processing to WWW browsers to games.

History of Python

Python was developed by Guido van Rossum in the late eighties and early nineties at the National Research Institute for Mathematics and Computer Science in the Netherlands.

- Python is derived from many other languages, including ABC, Modula-3, C, C++, Algol-68, SmallTalk, and Unix shell and other scripting languages.

- Python is copyrighted. Like Perl, Python source code is now available under the GNU General Public License (GPL).

- Python is now maintained by a core development team at the institute, although Guido van Rossum still holds a vital role in directing its progress.

- Python 1.0 was released in November 1994. In 2000, Python 2.0 was released. Python 2.7.11 is the latest edition of Python 2.

- Meanwhile, Python 3.0 was released in 2008. Python 3 is not backward compatible with Python 2. The emphasis in Python 3 had been on the removal of duplicate programming constructs and modules so that "There should be one -- and preferably only one -- obvious way to do it." Python 3.5.1 is the latest version of Python 3.

Python Features

Python's features include-

- **Easy-to-learn:** Python has few keywords, simple structure, and a clearly defined syntax. This allows a student to pick up the language quickly.
- **Easy-to-read:** Python code is more clearly defined and visible to the eyes.
- **Easy-to-maintain:** Python's source code is fairly easy-to-maintain.
- **A broad standard library:** Python's bulk of the library is very portable and cross-platform compatible on UNIX, Windows, and Macintosh.
- **Interactive Mode:** Python has support for an interactive mode, which allows interactive testing and debugging of snippets of code.
- **Portable:** Python can run on a wide variety of hardware platforms and has the same interface on all platforms.
- **Extendable:** You can add low-level modules to the Python interpreter. These modules enable programmers to add to or customize their tools to be more efficient.
- **Databases:** Python provides interfaces to all major commercial databases.
- **GUI Programming:** Python supports GUI applications that can be created and ported to many system calls, libraries and windows systems, such as Windows MFC, Macintosh, and the X Window system of Unix.
- **Scalable:** Python provides a better structure and support for large programs than shell scripting.

Apart from the above-mentioned features, Python has a big list of good features. A few are listed below-

- It supports functional and structured programming methods as well as OOP.
- It can be used as a scripting language or can be compiled to byte-code for building large applications.
- It provides very high-level dynamic data types and supports dynamic type checking.
- It supports automatic garbage collection.
- It can be easily integrated with C, C++, COM, ActiveX, CORBA, and Java.