**Python 2 and Python 3**

|  |  |
| --- | --- |
| **Python 2** | **Python 3** |
| Released in 2000 | Released in 2008 |
| Print keyword is considered to be a statement and not a function. | Print keyword is considered to be a function and not a statement |
| Strings are stored as ASCII by default | Strings are stored as UNICODE by default |
| On the division of two integers, we get an integral value. Ex 5/2 yields 2 in python 2 | On the division of two integers we get a floating point value in python 3. Ex 5/2 yields 2.5in python 3 |
| Exception are enclosed in notations | Exceptions are enclosed in parentheses |
| The xrange() function has been defined for iteration | The Range() function was introduced to perform iterations |
| Contains complicated syntax than Python 3 | Easer syntax compared to python 2 |
| A lot of libraries of python 2 are not forward compatible. | A lot of libraries are created in python 3 to be strictly used with python 3 |
| Is no longer use since 2020 | Is more popular than python 2 and still in use todays use |
| Python 2 codes can be ported to python 3 with a lot of effort | Python 3 is not backward compatible with python 3 |
| DevOps engineers | Is used in lot of fields software engineer data science. |

**Input()**

Input() function is used to take input from the user. Whatever you enter as input, the input function converts it into a string.

Syntax

Input(prompt)

**printf()**

printf() is an built in function available in python to pirint the

specified message on the screen or other standard output device.

The message can be string or any object. the object will be converted into

a string before written to the screen.

Syntax.

print(object(a))

Here the object will be converted into string before printing.

Example:

print("Welcome to Python")

**Interpreter and compiler**

|  |  |
| --- | --- |
| **Interpreter** | **Compiler** |
| Translates program one statement at a time. | Scans the entire program and translates it as a whole into machine code. |
| Interpreters usually take less amount of time to analyze the source code. | Compilers usually take a less amount of time to analyze the source code. |
| The overall execution time is comparatively slower than compiler | The overall execution time is comparatively faster than interpreters |
| No object code is penetrated, hence are memory efficient | Generates object code with further requires linking, hence requires more memory |
| Programming languages like JavaScript, Python, Ruby use interpreters. | Programming languages like C, C++, Java use compilers. |