**2.18 Formatted outputs of Python**

On some occasions, the output obtained by the simple print instructions is not suitable for display. So, there are some special methods to customize the view according to our desire. For that, we can incorporate the print statement with % operator and format string. The important format strings are the following.

|  |  |  |  |
| --- | --- | --- | --- |
| **Format string** | **Data type** | **Example** | **Result** |
| d | integer | %3d | Print as an integer. 3 spaces will be reserved |
| f | float | %7.3f | Print as a float. 7 spaces, including 3 decimals will be reserved |
| s | String | %30s | Print as a string. 30 spaces will be reserved. |
| e | exponential | %6.2e | Print as a floating-point exponential. The mantissa part will be a float. 6 spaces, including 2 decimals will be reserved. The exponent part always will be an integer of width 2. |

Normally the number will be with the right justification. So, a positive number as the width will justify the numerical value to the right. A negative number as width stands for a left-justified numeral. %3d stands for right justified integer of width 3, whereas %-3d stands for the left-justified integer of width 3. In the case of strings, positive numbers as width can be used to make left justification and negative numbers for right justification. The data along with the formatted print statement must be a tuple. The number of format strings must match in number with the elements in the tuple. Go through the following segments of print statements to clarify the proper usage.

*x=12.8*

*y=2.1*

*print( x*

*print(‘The answer is %7.2f’%(x))*

*print(‘The answer is %7d’%(x))*

*print(‘The sum of %8.2 and %8.2 will %9.3’ %(x,y,x+y))*

**Output**

*12.8*

*The answer is 12.80*

*The answer is 12*

*The sum of 12.80 and 2.10 will be 14.900*

This will be highly helpful in printing matrix, table values etc.