Python Printing

Contents

[**Python Printing** 3](#_Toc89087470)

[Input - string modulo operator(%) 3](#_Toc89087471)

[Output 3](#_Toc89087472)

[**Formatting output using the format method :** 4](#_Toc89087473)

[Output 1: 4](#_Toc89087474)

[Example 2: 4](#_Toc89087475)

[Output 2: 5](#_Toc89087476)

[Example 3: 5](#_Toc89087477)

[Output 3: 6](#_Toc89087478)

[**Formatting output using the String method :** 6](#_Toc89087479)

[Output: 6](#_Toc89087480)

[**Examples** 7](#_Toc89087481)

[Matillion / Star2Star Dimension Tables Print and set Python variables 7](#_Toc89087482)

[**Reference:** 8](#_Toc89087483)

# **Python Printing**

# Simple Print

print("Hello World")

first\_name = ‘Hal’

last\_name = ‘Baley’

print(first\_name , last\_name)

print("First Name: [{}] Last Name: [{}]".format(first\_name , last\_name))

# Python Package with .py see “How to Run Your Python Scripts”

## Input - string modulo operator(%)

# Python program showing how to use

# string modulo operator(%) to print

# fancier output

# print integer and float value

print("Geeks : %2d, Portal : %5.2f" % (1, 05.333))

# print integer value

print("Total students : %3d, Boys : %2d" % (240, 120))

# print octal value

print("%7.3o" % (25))

# print exponential value

print("%10.3E" % (356.08977))

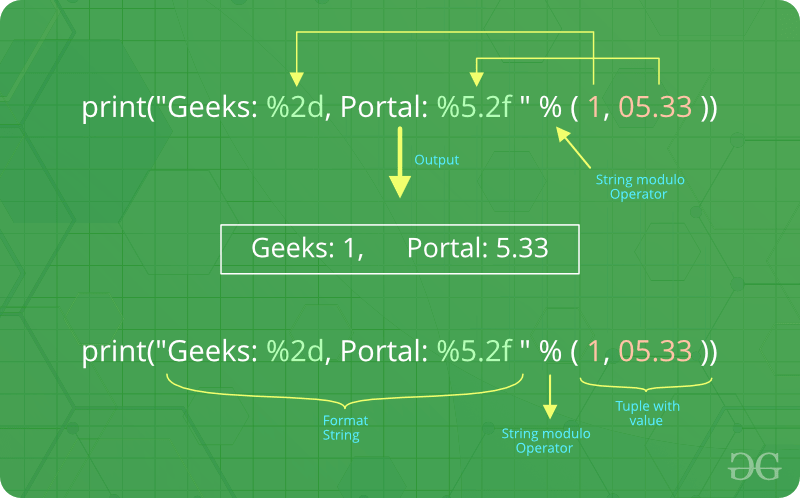
## Output

Geeks : 1, Portal : 5.33

Total students : 240, Boys : 120

031

3.561E+02



There are two of those in our example: “%2d” and “%5.2f”. The general syntax for a format placeholder is:

%[flags][width][.precision]type

# **Formatting output using the format method :**

# Python program showing

# use of format() method

# using format() method

print('I love {} for "{}!"'.format('Geeks', 'Geeks'))

# using format() method and referring

# a position of the object

print('{0} and {1}'.format('Geeks', 'Portal'))

print('{1} and {0}'.format('Geeks', 'Portal'))

# the above formatting can also be done by using f-Strings

# Although, this features work only with python 3.6 or above.

print(f"I love {'Geeks'} for \"{'Geeks'}!\"")

# using format() method and referring

# a position of the object

print(f"{'Geeks'} and {'Portal'}")

## Output 1:

I love Geeks for "Geeks!"

Geeks and Portal

Portal and Geeks

## Example 2:

# Python program showing

# a use of format() method

# combining positional and keyword arguments

print('Number one portal is {0}, {1}, and {other}.'

.format('Geeks', 'For', other ='Geeks'))

# using format() method with number

print("Geeks :{0:2d}, Portal :{1:8.2f}".

format(12, 00.546))

# Changing positional argument

print("Second argument: {1:3d}, first one: {0:7.2f}".

format(47.42, 11))

print("Geeks: {a:5d}, Portal: {p:8.2f}".

format(a = 453, p = 59.058))

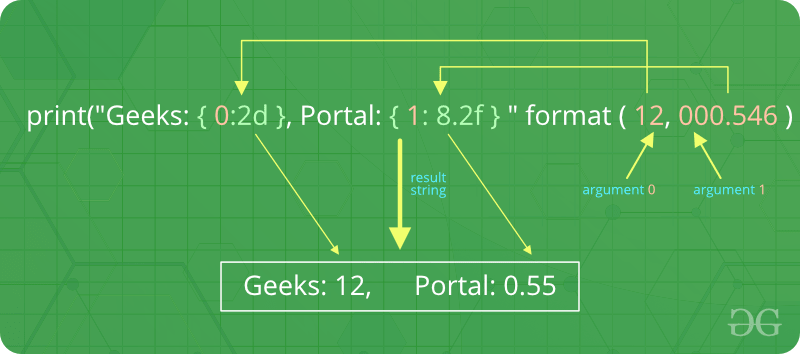
## Output 2:

Number one portal is Geeks, For, and Geeks.

Geeks :12, Portal : 0.55

Second argument: 11, first one: 47.42

Geeks: 453, Portal: 59.06



## Example 3:

# Python program to

# show format () is

# used in dictionary

tab = {'geeks': 4127, 'for': 4098, 'geek': 8637678}

# using format() in dictionary

print('Geeks: {0[geeks]:d}; For: {0[for]:d}; '

'Geeks: {0[geek]:d}'.format(tab))

data = dict(fun ="GeeksForGeeks", adj ="Portal")

# using format() in dictionary

print("I love {fun} computer {adj}".format(\*\*data))

## Output 3:

Geeks: 4127; For: 4098; Geeks: 8637678

I love GeeksForGeeks computer Portal

# **Formatting output using the String method :**

This output is formatted by using string slicing and concatenation operations. The string type has some methods that help in formatting output in a fancier way. Some of method which help in formatting a output are str.rjust(), str.rjust(), str.centre()

# Python program to

# format a output using

# string() method

cstr = "I love geeksforgeeks"

# Printing the center aligned

# string with fillchr

print ("Center aligned string with fillchr: ")

print (cstr.center(40, '#'))

# Printing the left aligned

# string with "-" padding

print ("The left aligned string is : ")

print (cstr.ljust(40, '-'))

# Printing the right aligned string

# with "-" padding

print ("The right aligned string is : ")

print (cstr.rjust(40, '-'))

## Output:

Center aligned string with fillchr:

##########I love geeksforgeeks##########

The left aligned string is :

I love geeksforgeeks--------------------

The right aligned string is :

--------------------I love geeksforgeeks

# **Examples**

## Matillion / Star2Star Dimension Tables Print and set Python variables

print('file\_settings array from grid:')

print()

file\_settings = context.getGridVariable('file\_settings') [0]

#print(file\_settings)

print()

print('create and initialize variables for python local session:')

print()

context.updateVariable('transpose', file\_settings[0])

context.updateVariable('file\_type', file\_settings[1])

context.updateVariable('sheet', file\_settings[2])

context.updateVariable('skip\_rows', file\_settings[3])

context.updateVariable('drop\_rows', file\_settings[4])

context.updateVariable('skip\_columns', file\_settings[5])

context.updateVariable('drop\_columns', file\_settings[6])

print('s3\_settings array from grid:')

s3\_settings = context.getGridVariable('s3\_settings') [0]

print(s3\_settings)

print()

print('create and initialize variables for python local session:')

print()

context.updateVariable('bucket', s3\_settings[0])

context.updateVariable('prefix', s3\_settings[1])

snowflake\_settings = context.getGridVariable('snowflake\_settings')[0]

print(snowflake\_settings)

print()

print('create and initialize variables for python local session:')

print()

context.updateVariable('database', snowflake\_settings[0])

context.updateVariable('schema', snowflake\_settings[1])

context.updateVariable('table', snowflake\_settings[2])

print("Settings Imported:")

print("Bucket: [{}] Prefix: [{}] Sheet Name: [{}]".format(bucket, prefix,sheet))

print("Rows: SKIP [{}] DROP [{}]\nColumns: SKIP [{}] DROP [{}]".format(skip\_rows, drop\_rows,skip\_columns,drop\_columns))

print("Database: [{}] Schema: [{}] Table: [{}]".format(database, schema,table))

print("Transpose: [{}]".format(transpose))

# **Reference:**

* [Python | Output Formatting](https://www.geeksforgeeks.org/python-output-formatting/)