Python Programming

String Formatting 3

Mostafa S. Ibrahim
Teaching, Training and Coaching since more than a decade!

Artificial Intelligence & Computer Vision Researcher
PhD from Simon Fraser University - Canada
Bachelor / Msc from Cairo University - Egypt
Ex-(Software Engineer / ICPC World Finalist)

Precision

```
2   val = 71.012345678901234567890123456789012345678901234567890123456789

3
4   print(val)                      #71.01234567890124 ==> 14 decisimal precision printed
5   print('{:20}'.format(val))      #   71.01234567890124 ==> total 20 output units, right-aligned
6   print('{:11f}'.format(val))     #  71.012346 ==>  print 11 units. Use default precision (typically 6)
7   print('{:11.3f}'.format(val))   #     71.012 ==> 11 output units, 3 of them precision
8   print('{:3.5f}'.format(val))    #71.01235 ==> 5 precision. It will have more priority
9   print('{:.8f}'.format(val))     #71.01234568 ==> 8 precision. No specific alignments
10  #print('{.8f}'.format(val))     #AttributeError
11
12
13  val = 2.67
14  print(val)                     #2.67
15  print('{:11f}'.format(val))    #   2.670000 ==>  trailing zeros  : 11 output units (6 is precision)
16  print('{:11.2f}'.format(val))  #       2.67   (.2f use 2 precision)
17  print('{:11.1f}'.format(val))  #        2.7   rounding
18  print('{:11.0f}'.format(val))  #          3   rounding
19
20  print('{:11.0f}'.format(2.5))  #          2   rounding to 2
21  print('{:11.0f}'.format(-2.5))  #         -2   rounding to -2
22
23  # {value:width.precision}
```

F-string: The very modern way

```
3   name, age = 'mostafa', 33
4
5   # mostafa is 33 years old
6   print('{} is {} years old'.format(name, age))
7   print('{name} is {age} years old'.format(name=name, age=age))
8
9   print(f'{name} is {age} years old')
10
11
12  val = 71.012345678901234567890123 4
13
14  #      71.012
15  print('{:11.3f}'.format(val))
16  print(f'{val:11.3f}')
```

F-string: The very modern way

```
class Employee:
    def __init__(self, name, age):
        self.name = name
        self.age = age

    def __str__(self):
        return f'Employee {self.name} is {self.age} years old'

    def __repr__(self):
        return f'Employee(name="{self.name}", age={self.age})'

most = Employee('mostafa', 33)
print(f'{most}')        # Employee mostafa is 33 years old
# add !r to use the dunder repr
print(f'{most!r}')      # Employee(name="mostafa", age=33)

print(f"{2 * 3+ 1}")    # 7
name = 'mostafa'
print(f"{name.lower()} has udemy courses")  # mostafa has udemy courses
```

Modulus Operator: The very old way

```
3   # mostafa is 33 years and salary 100.578900
4   print('%s is %d years and salary %f' % ('mostafa', 33, 100.5789))
5
6   #mostafa is 33 years and salary 100.579
7   print('%s is %d years and salary %.3f' % ('mostafa', 33, 100.5789))
8
9   #mostafa is 33 years and salary          100.579
10  print('%s is %d years and salary %15.3f' % ('mostafa', 33, 100.5789))
11
12  print('%d' % 123)    # 123
13  #print('%d' % '123')    # 123         TypeError
```

"Acquire knowledge and impart it to the people."

"Seek knowledge from the Cradle to the Grave."