SASTRA
ENGINEERING · MANAGEMENT · LAW · SCIENCES · HUMANITIES · EDUCATION
DEEMED TO BE UNIVERSITY
(U/S 3 OF THE UGC ACT, 1956)
THINK MERIT | THINK TRANSPARENCY | THINK SASTRA

III B.Tech.

Computer Science & Engineering

CSE304: PYTHON PROGRAMMING WITH WEB FRAMEWORKS

UNIT – II: More on Numbers and Strings

By
Mrs. S. KAMAKSHI, AP-III / CSE
School of Computing

Math Module

Function / Constant	Description
pow(num, exp)	Raises the number to a specific power
sqrt(num)	Returns the square root of the number
ceil(num)	Rounds the floating point number up to the nearest integer
floor(num)	Rounds the floating point number down to the nearest integer
pi	The value of pi to 15 decimal positions

format() method of string

- Used in print statements to format the numbers along with string
- Syntax:
  - "{:format_specification} …".format(data_item …)
  - Format specification syntax:
    - [field_width][comma][.decimal_places][type_code]

Code	Meaning	Description
d	Integer	Decimal positions can't be specified
f	Float	Decimal positions can be specified
%	Percent	Multiplies the value by 100 and puts a % sign after this
e	Scientific Notation	Converts the number to scientific notation

Examples

```
num1 = 12345.6789
print(“{:.2f}”.format(num1))            #12345.68
print(“{:.4f}”.format(num1))            #12345.6789
print(“{:,.2f}”.format(num1))           #12,345.68
print(“{:15,.2f}”.format(num1))         #12,345.68

num2 = 12345
print(“{:d}”.format(num2))              #12345
print(“{:,d}”.format(num2))             #12,345

num3 = 0.12345
print(“{:.0%}”.format(num3))            #12%
print(“{:.1%}”.format(num3))            #12.3%

num4 = 12345.6789
print(“{:.2e}”.format(num4))            #1.23e+04
print(“{:.4e}”.format(num4))            #1.2346e+04
```

Formatting Output using width

```
print("{:15} {:>10} {:>5}".format("Description", "Price", "Qty"))
print("{:15} {:>10.2f} {:>5d}".format("Hammer", 9.99, 3))
print("{:15} {:>10.2f} {:>5d}".format("Nails", 14.50, 10))
```

Output:

```
Description          Price   Qty
Hammer                9.99     3
Nails                14.50    10
```

Using locale module

Function	Description
setlocale(category, locale)	Sets the locale for the specified category to the locale for the specified country code and returns a string for the locale. If category is set to LC_ALL, the locale is applied to all categories If locale is an empty string, it attempts to set the locale to the user’s default locale. If this is not possible, it returns a code of “C”
currency(num [, grouping])	Returns the specified number formatted as currency. If grouping is set to True, the number includes thousands separators
format_string(format, num [, grouping])	Returns the specified number formatted for the current locale. If grouping is set to True, the number includes thousands separators.

Codes for working with locales

Locale	Short Code	Long Code	Currency Format
English / United States	us	en_US	$12,345.15
English / United Kingdom	uk	en_UK	£12,345.15
German/Germany	de	de_DE	12.345,15 €

Examples

```
import locale as lc
lc.setlocale(lc.LC_ALL, "us")
print(lc.currency(12345.15, grouping = True))
print(lc.format_string("%d", 12345, grouping = True))
print(lc.format_string("%.2f", 12345.15, grouping = True))
$12,345.15
12,345
12,345.15
lc.setlocale(lc.LC_ALL, "uk")
print(lc.currency(12345.15, grouping = True))
print(lc.format_string("%.2f", 12345.15, grouping = True))
£12,345.15
12,345.15
lc.setlocale(lc.LC_ALL, "de")
print(lc.currency(12345.15, grouping = True))
print(lc.format_string("%.2f", 12345.15, grouping = True))
12.345,15 €
12.345,15
```

Decimal class from decimal module

- To create decimal numbers that are exact and don’t yield unexpected results as floating point numbers
- Floating point calculations are more faster than Decimal calculations
- Create decimal objecting by importing Decimal class from the decimal module and passing the decimal number as a string to the constructor of the class
- All arithmetic operations can be used with Decimal objects
- In expressions, int values can be mixed with Decimal Objects but float cannot be mixed with Decimal objects
- To round decimal values to the specified number of decimal places:
  - dec_obj.quantize(Decimal(“positions_code”) [, rounding _constant])
  - rounding _constant may be ROUND_HALF_UP or ROUND_HALF_DOWN or ROUND_HALF_EVEN

Example

```
from decimal import Decimal
from decimal import ROUND_HALF_UP,
                    ROUND_HALF_DOWN, ROUND_HALF_EVEN
dec_obj = Decimal(“0.05465”)
dec_obj.quantize(Decimal(“1.0000”) , ROUND_HALF_UP)
Decimal(“0.0547”)
dec_obj.quantize(Decimal(“1.0000”) , ROUND_HALF_DOWN)
Decimal(“0.0546”)
dec_obj.quantize(Decimal(“1.0000”) , ROUND_HALF_EVEN)
Decimal(“0.0546”)
```

String functions

- ord(char) – ordinal value of the character
- len(str) – length of the string
- Indexing – string[index]
- Slicing - string[start:end:step]
- Searching in string
  - substring in string – returns True or False
- Looping through characters in string:
  - for c in string:

    statements …
- Basic functions in string
  - isdigit(), isalpha(), islower(), isupper(), startswith(str), endswith(str), title(), upper(), lower(), strip(), rstrip(), lstrip(), ljust(width), rjust(width), center(width), find(str[, start] [, end]), replace(old, new[, num]), split(delimiter), join(delimiter)