NPTEL MOOC

PROGRAMMING, DATA STRUCTURES AND ALGORITHMS IN PYTHON

Week 5, Lecture 5

Madhavan Mukund, Chennai Mathematical Institute
http://www.cmi.ac.in/~madhavan

Formatted printing

- Recall that we have limited control over how `print()` displays output
- Optional argument `end="..."` changes default new line at the end of print
- Optional argument `sep="..."` changes default separator between items

String format() method

- By example

```
>>> "First: {0}, second: {1}".format(47,11)
'First: 47, second: 11'

>>> "Second: {1}, first: {0}".format(47,11)
'Second: 11, first: 47'
```

- Replace arguments by position in message string

format() method …

- Can also replace arguments by name

```
>>> "One: {f}, two: {s}".format(f=47,s=11)
'One: 47, two: 11'

>>> "One: {f}, two: {s}".format(s=11,f=47)
‘One: 47, two: 11'
```

Now, real formatting

```
>>> "Value: {0:3d}".format(4)
```

- `3d` describes how to display the value `4`
- `d` is a code specifies that `4` should be treated as an integer value
- `3` is the width of the area to show `4`

```
'Value:   4'
```

Now, real formatting

```
>>> "Value: {0:6.2f}".format(47.523)
```

- `6.2f` describes how to display the value `47.523`
- `f` is a code specifies that `47.523` should be treated as a floating point value
- `6` — width of the area to show `47.523`
- `2` — number of digits to show after decimal point

```
"Value:  47.52"
```

Real formatting

- Codes for other types of values
  - String, octal number, hexadecimal …
- Other positioning information
  - Left justify
  - Add leading zeroes
- Derived from `printf()` of C, see Python documentation for details