Strings: Definitions and Operations

Thomas Schwarz, SJ

String Definition

- Strings are defined as text enclosed with single or double quotation mark
  - Within the text we can use the other quotation marks without problem
  -
```
a_string = 'He said: "Hello."'

another_string = "What's its worth?"
```

String Definition

- Example:

```
>>> astring = "Hello World"
>>> bstring = 'Hello World'
>>> astring == bstring
True
```

String Definition

- To import long strings with newlines:
  - Use the triple quotes:

```
results = '''
a=145, b=345
a=250, b=332
a=307, b=301
a=346, b=298
'''
```

- But we can also use the newline character:

```
results = '\na=145,b=345\na=50, b=332\n'
```

String Definition

- Some characters within a string are compounded:
  - Backslash followed by a letter
    - Newline: \n translated by Python into the appropriate sequence for the OS,
      - can also use \r carriage return or \f formfeed
  - Tab: \t
  - \' , \" the single and double quotation mark
  - \\ the backslash
  - \v a vertical tab

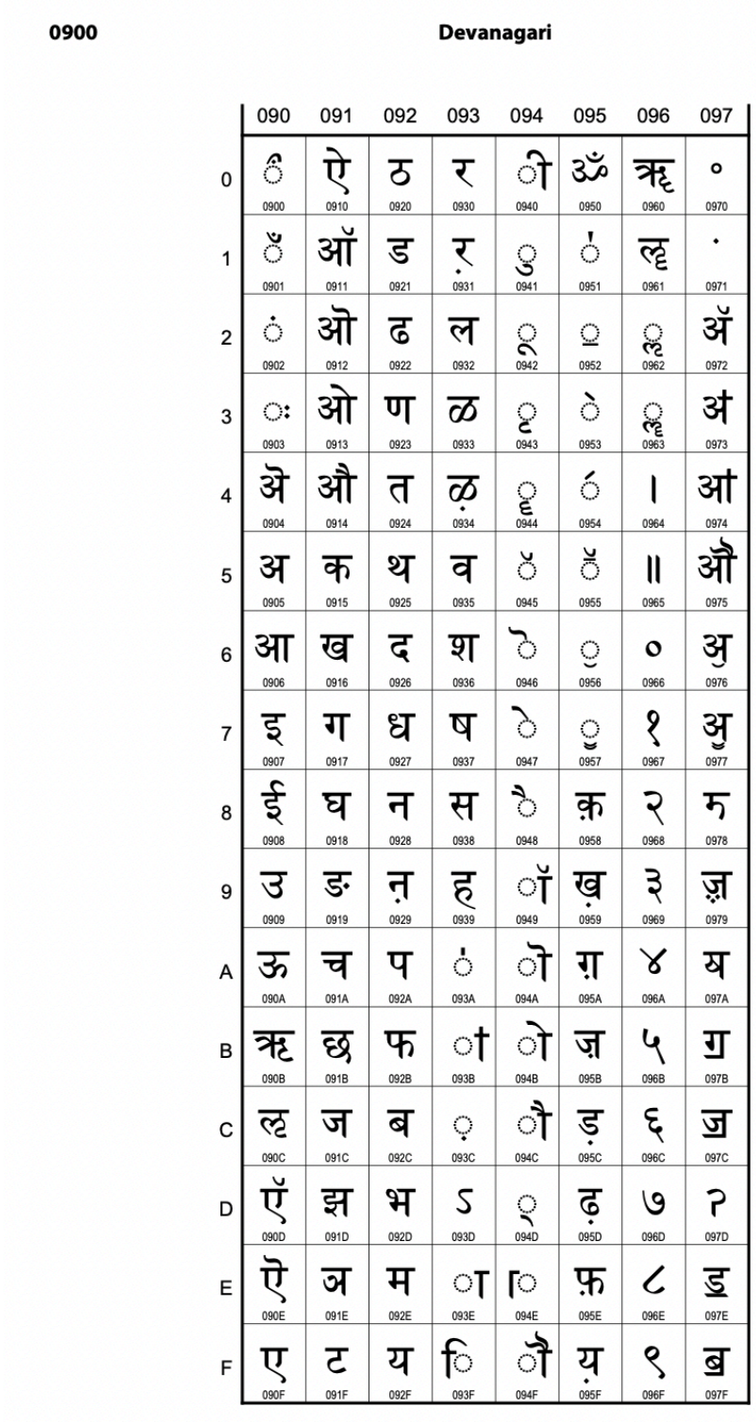
String Definition

- The backslash is the ***escape*** character
- Python is very good at interpreting your intent
  - E.g. my attempt at a swimmer doodle
    - Internally, backslashes are inserted
- Basically, an invalid escape sequence uses a literal backslash

```
>>> astring='\__o/\ '
>>> astring
'\\__o/\\ '
>>> print(astring)
\__o/\
```

String Definition

- Special characters not on your keyboard:
  - Can insert unicode with
    - \u…. sequence
  - Replace the dots with the unicode
  - E.g.: `print('\u0904)` yields ऄ



String Definition

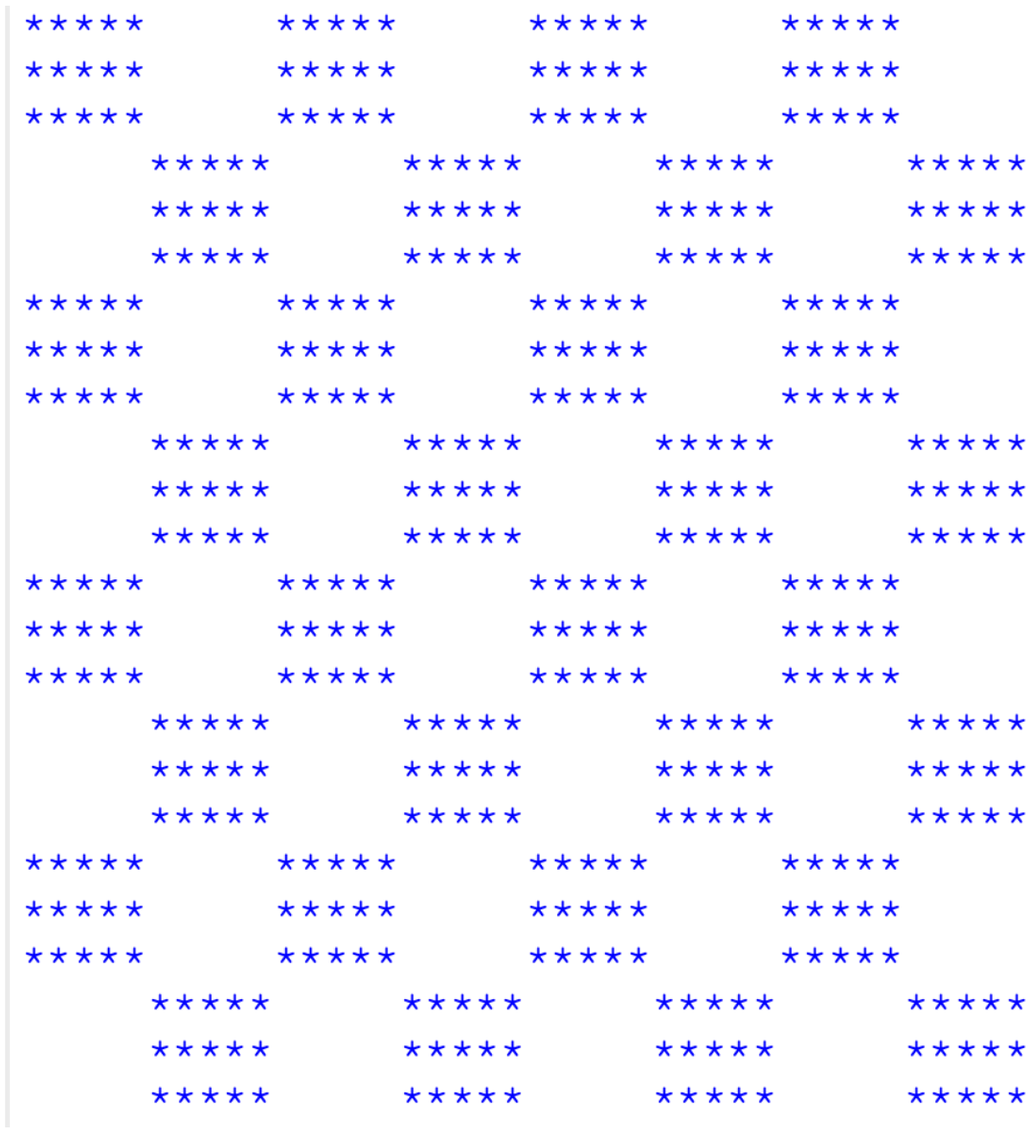
- If you want to disable the escape character, use *raw* strings
  - If in the near future, you need to create strings such as http contents, etc,
    - Place an r in front of the string
      - E.g. `print(r'\t\n\r\f')` yields `\t\n\r\f`

String Operations

- Recall:
  - We can add two strings : concatenation
  - We can multiply a string with an integer (in any order):
  - Example: Print out a line:
    - `print('-'*25)`

String Operations

- Printing an ASCII chessboard



String Operations

- The first line consists of:
  - five asterisks (stars) followed by five white spaces
    - `5*'*' + 5*' '`
  - repeated four times:
    - `4*(5*'*'+5*' ')`
  - generalize this by giving names to the arguments
    - `print(fields*(width*'*'+width*' '))`

String Operations

- And put it into two functions:
  - ```
    def left(width, height, fields):
        for _ in range(height):
            print(fields*(width*'*'+width*' '))
    ```

```
def right(width, height, fields):
    for _ in range(height):
        print(fields*(width*' '+width*'*'))
```

- And put them together:

```
for _ in range(4):
    left(width=5, height=3, fields=4)
    right(width=5, height=3, fields=4)
```
```