COL1000: Introduction to Programming

Strings

Subodh Sharma | Lec 14 | Sept 03

Type & Conversion (RECAP)

- `int(float_data)`: truncates towards 0
- `round(x)` : rounds to the nearest integer (round half to nearest even)
- `math.floor(x)`: truncates towards ∞
- `Str(mostly_any_type_of_data)`: converts input data to string
- `tuple(lst):` Converts list to *immutable* tuple
- `set(lst)`: removes duplicates and gives an *unordered* collection

Containers: Lists, Tuples, Strings

Lists (RECAP)

Ordinal Containers

- Each list item has an ordinal position
- List functions
  - Append: `lst1 = [1,2,3]; ls1.append("e") # lst1 = [1, 2, 3, "e"]`
  - Insert: `lst1.insert(1, "svs") # lst1 = [1, "svs", 2, 3, "e"]`
  - Remove: `lst1.remove("e") # lst1 = [1, "svs", 2, 3]`
  - Reverse: `lst1.reverse() # lst1 = [3, 2, "svs", 1] – done in-place`
  - Sort: `lst1.sort() # lst1 = TypeError, < is unsupported for str and int`
    - If lst1 `= [2,1,5,3]` then `lst1.sort() = [1,2,3,5]`
  - Sort (reverse): `lst1.sort(reverse = true) # lst1 = [5,3,2,1]`

Lists (RECAP)

Shallow Copy vs Deep Copy

- **Shallow copy** — creates a new object, but reuse the references for internal entities
  - `A = [1,2, [3,4]]; B = A.copy(); id(A) != id(B); id(A[2]) == id(B[2])`
- **Deep copy** — create new object and recursively all entities internally — nothing is shared!
  - `A = [1,2, [3,4]]; import copy; B = copy.deepcopy(A); id(A) != id(B); id(A[2]) != id(B[2])`

Lists (RECAP)

Slicing

- `lst1 = [x for x in range(6)] # lst1=[0,1,2,3,4,5]`
- `lst1[2:5] # [2,3,4] – up to but excluding the “to” param`
- `lst1[:3] # [0,1,2]`
- `lst1[::2] # skips every 2nd element; [0,2,4]`
- `lst1[::-1] # reverses the list`
- `lst1[5:2:-1] # ?`

Lists (RECAP)

Combinations, Comprehensions, Built-in functions

- `lst1 = [[0]*3] # lst1=[[0,0,0]]`
- `A + B: concatenation of lists`
- `lst1=[x*x if x > 0 else -x*x for x in range(5)]`
- `min, max, len functions`

Tuples (RECAP)

- Ordered, immutable, heterogeneous container
  - Tuple id remains the same; but internal elements, if mutable, are allowed to change
- `head, *mid, tail = (1, 2, ‘svs’, 3, 4)`
  - `head = 1; tail = 4; mid = [2, ‘svs’, 3]`
- `_,y = (25, 46) # _ is used for ignoring during unpacking`
- Comprehensions work just the same as they do for lists
- min, max, reverse, sort functions work just same with a caveat:
  - `sorted(tup) # produces a list`
  - `tuple(sorted(tup)) # produces a sorted tuple`

Strings

- Ordered, immutable, container
  - `“svs”, ‘svs’` [They both are equivalent]
  - `‘Today is a “good day”!’` [quotes within string]
    - Use escape char \: `“Today is a \“good day\”!”`
  - Indexing/Slicing: `s=‘abcdef’; s[1:4] # ‘bcd’’ s[::-1] # ‘fedcba’`
    - `Try: s[4:2:-1]`
  - `s[0] = ‘g’ # TypeError: doesn’t support item assignment`
- Concatenation: s1 + s2 (produces a new object)
- Repetition: `s = “svs”; s*2 # “svssvs”`

Strings

- Raw strings:
  - `str = ”svs\’svs” # svs’svs; len(str) = 7`
  - `str = r”svs\’svs” # svs\\’svs; len(str) = 8`
- Formatted strings
  - `marks = 97.5678; str = f“average marks are {marks}”`
  - `str = f“average marks are {marks/2}”`
  - `str = f“average marks are {marks:.2f}”`
  - `str = f“average marks are {marks:3.2f}”`

Strings — Core Methods

- s = “Hello World”
  - `s.lower() # “hello world”`
  - `s.upper() # “HELLO WORLD”`
  - `s.title() # “Hello World`
  - `s.find(“lo”) # returns the index 3 ; is case-sensitive; -1 if the substring is not found`
  - `s.split() # [“hello”, “world”]`
  - `lst = [“03”, “09”, “2025”]; “/“.join(lst) -> .split(“/“) == lst?`
  - `“  Hello  “.strip() # “Hello”; “Hello”.strip(“lo”) # remove l or o from the ends -> “He”`

Strings — Core Methods

- s = “Hello World”
  - `s.replace(“l”, “m”) # “Hemmo Wormd”`
  - `s.replace(“l”, “m”, 1) # count-limited -> “Hemlo World”`
- Other methods:
  - isdigit, isnumeric, isdecimal, …
- **Refer: https://docs.python.org/3/library/stdtypes.html#string-methods**
- **UTF Encoding — Advanced material (may be later)!**

String Search — find()

- How do we implement it?