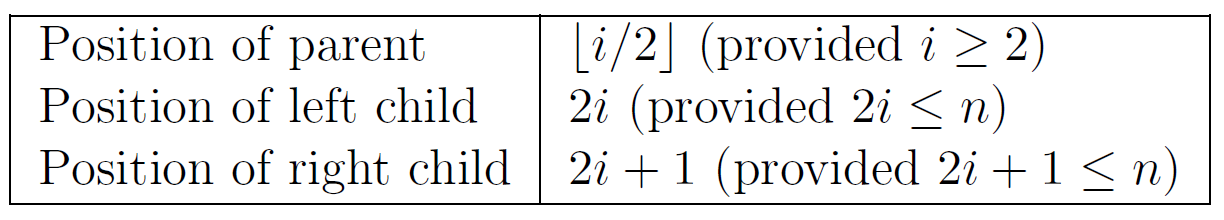
**Heap**

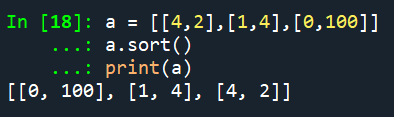
* Heaps as arrays (index starts from 1):



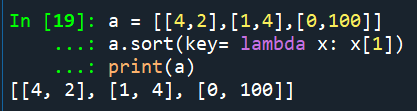
* Running time:
  + Insert: O(log *n*)
  + Extract min: O(log *n*)
  + Build a heap (heapify) O(n)
  + Heap sort: O(n log *n*)

**Sort**

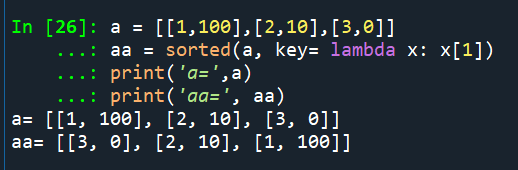
* Sort a list of lists
  + The default sort() function sort in line and by the first element



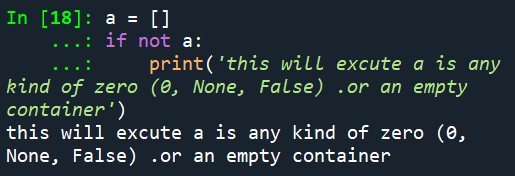
* + Sort by other elements using lambda, inline.



* + Return a copy using sorted

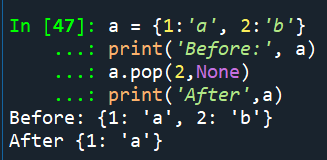


**Logic**

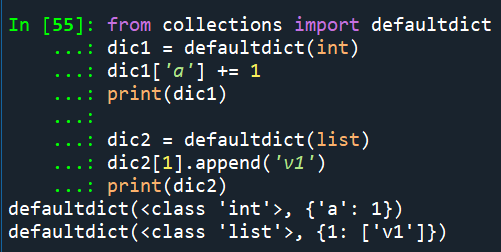


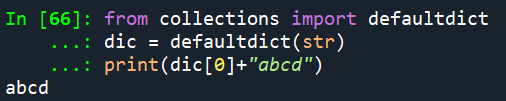
**Dictionary**

* Delete a key:

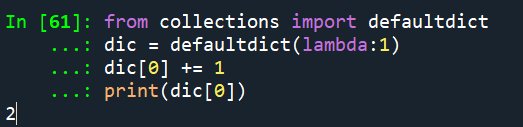


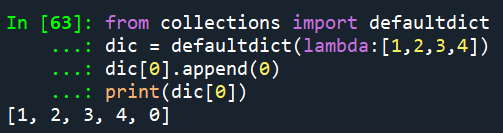
* Default dictionary:



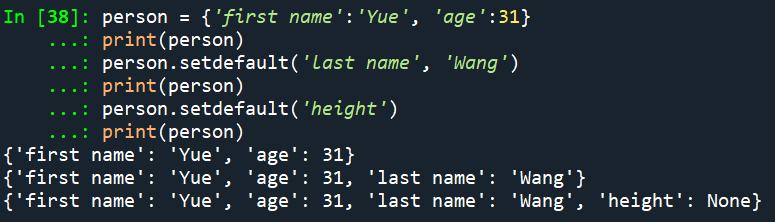


Default dictionary with other default values:



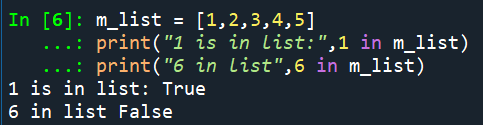


* Set default value of a normal dictionary



**List**

* Whether an element is in the list:



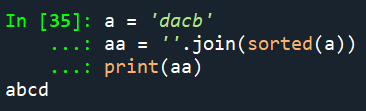
**Binary**

* An algorithm for finding the binary representation of a positive number
  + Step1. Divide the value by 2 and record the remainder
  + Step2. As long as the quotient obtained is not 0, continue to divide the newest quotient by 2 and record the remainder
  + Now that a quotient of 0 has been obtained, the binary representation is the remainders listed from right to left in the order they were recorded.
* Fractions in binary
  + 
* An algorithm for decoding two’s complement representations:
  + Copy the original number form right to left until a 1 has been copied, then complement the remaining its as they are transferred to the final bit pattern.

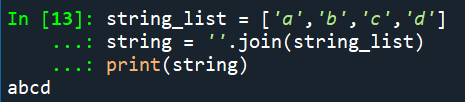


**String and char**

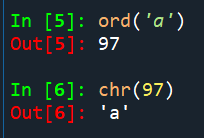
* Sort a string



* Convert a list of char to a string

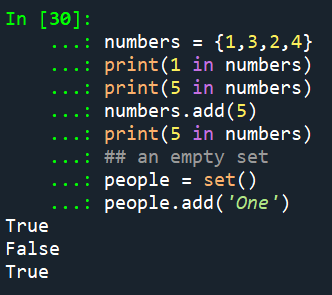


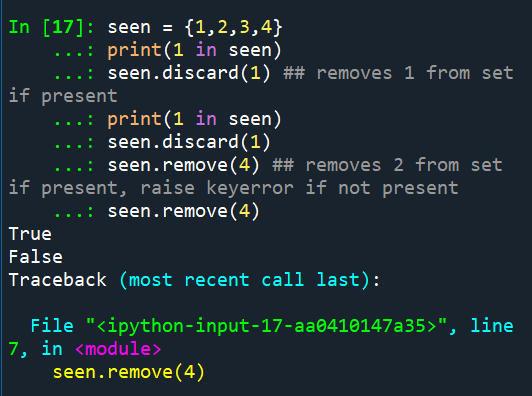
* Convert between char and ARSII number



**Set**

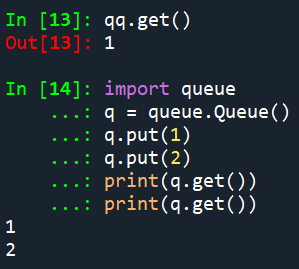
* Initialize and add elements into a set





**Queue**

* The queue module



* The deque module

