s.upper()	Converts the string in uppercase
s.lower()	Converts the string in lowercase
s.startswith(substr)	Return true ,if the given string starts with substr
s.endswith(substr)	Returns true if the string ends with substr
s.find(substr,[start,end])	Return the position of 1 st occurrence of substr , if the start and end is not given then it searches the full string, otherwise it searches in the given range, it returns -1 if substr not found
s.rfind(substr,[start,end])	Return the position of last occurrence of substr , if the start and end is not given then it searches the full string, otherwise it searches in the given range, it return -1 if substr not found
s.index(substr,[start,end])	Return the position of 1 st occurrence of substr , if the start and end is not given then it searches the full string, otherwise it searches in the given range, but it throws exception if the substr not found
s.rindex(substr,[start,end])	Return the position of last occurrence of substr , if the start and end is not given then it searches the full string, otherwise it searches in the given range, but it throws exception if the substr not found
s.strip(string)	It will delete all occurrences of the characters from leftmost and rightmost side of the string
s.lstrip(string)	It will delete all occurrences of the characters from leftmost side of the string
s.rstrip(string)	It will delete all occurrences of the characters from rightmost side of the string
s.split(delimiter)	It will brek the string into parts at delimiter character and returns a list
“delimiter”.join(lst)	It will join all the strings from the list separated by delimiter
S.replace(oldsubstr,newsubstr,[count])	It will replace all occurrences of oldsubstr with newsubstr, if you specify count, then it will replace only first count number of occurrences
s.isalpha()	Returns true, if all the characters in the string are alphabets
s.isalnum()	Returns true if the string has all characters as either alphabet or digits, no special characters are allowed
s.count(substr)	It will show, the number of occurrences of the given string

String Type	Example	Python `.isdecimal()`	Python `.isdigit()`	Python `.isnumeric()`
Base 10 Numbers	`'0123'`	True	True	True
Fractions and Superscripts	`'⅔', '2²'`	False	True	True
Roman Numerals	`'Ⅾ'`	False	False	True

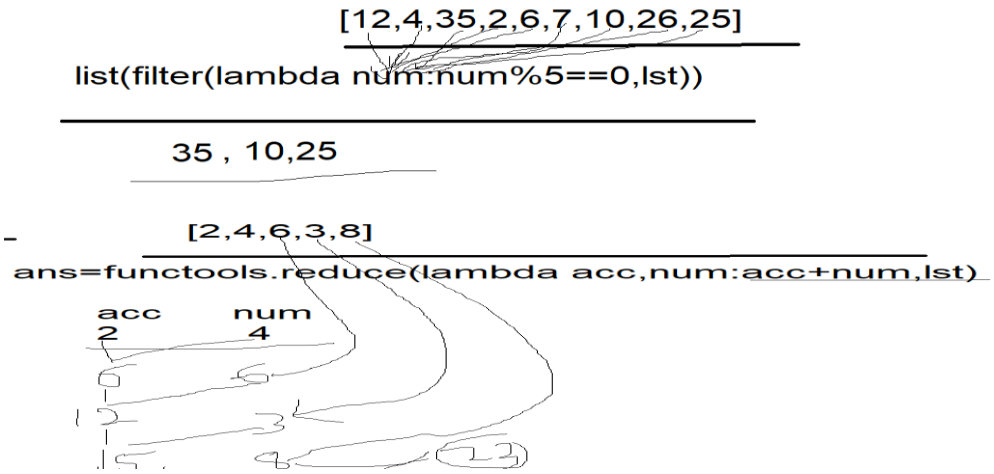
Other data types

1. Lists
    a. Lists allows you to store heterogeneous data.
    b. It is mutable data structure.
    c. Duplicate values are allowed.
    d. It is represented using []
    e. It is ordered collection, means the data will be stored in same sequence, as it is entered.
    f. Since it is ordered, using index is possible, hence data can be retrieved randomly.

lst.index(value)	Returns the position of first occurrence of the given value
Lst.append(value)	To add one value at the end of the list, and increase the length of the list by 1
Lst.extend(iterable)	To add multiple values from the given iterable at the end of the list, and increases the length by len(iterable)
Lst.insert(pos,value)	To add the value at the given position, if position is beyond the length of the list, then it will add at the end
Lst.pop([num])	To delete the last value from the list, if num is not given, but if we specify the num, then it will delete it from the given position, if the num is within limits of the length of the list, or it should not be empty
Lst.remove(value)	To delete the first occurrence of the given value from the list
Lst.count(value)	Display number of occurrences of the given value
Lst.clear()	To delete all the values from the list
Lst.copy()	To create a shallow copy of the list
Lst.reverse()	To reverse the list contents, it will change the original list
Lst.sort()	To arrange the data in the sorted order, by default it will arrange the data in ascending order,

	To arrange it in descending order use lst.sort(reverse=True) But sorting is possible only on the list which is homogeneous

filter(function,iterable)	Will internally generate the loop to navigate through the list one by one, and apply the function, if function returns True, then keep the value, else remove it Original iterable will remain as it is
map(function,iterable)	Will internally generate the loop to navigate through the list one by one, and apply the function, return the answer of the function, one for each value in the list
Reduce(function,iterable)	Will internally generate the loop to navigate through the list one by one, and apply the function, return the answer single value for all values from the list



2. Tuple
    a. It is immutable.
    b. It is represented using ().
    c. Mainly tuples are used to return multiple values from a function.
    d. Ordered collection, so we can access data by using index
    e. Since the indexing is possible, access to the data is random
    f. Duplicate values are allowed.
    g. Tuples are also iterable.
3. Set
4. Frozenset
5. dictionary