|  |
| --- |
| filter  The filter is useful for taking **data and removing** any information that you don’t need  The filter function **returns a sequence** from those **elements for which returns True**  Syntax: filter(function, iterable) |

|  |
| --- |
| #Even and Odd number using lambda function number = **lambda** x : **True if** x % 2 == 0 **else False** # lambda arguments : expression even = number(4) #which are divisible by 2 print(even) # True odd = number(5) #which are not divisible by 2 print(odd) # False |

|  |
| --- |
| #Even Numbers, using lambda function and filter() with function **def** d1(x):  **return** x % 2 == 0  lst = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 0, 1, 2, 3] result = filter(**lambda** x: x % 2 == 0, lst) print(list(result)) # [0, 2, 4, 6, 8, 0, 2] |

|  |
| --- |
| Case Study  employes = [  {"firstName":"Sai", "lastName":"Kiran", "age":27},  {"firstName":"Pradeep", "lastName":"Reddy", "age":29},  {"firstName":"Praneeth", "lastName":"Reddy", "age":35},  {"firstName":"Ranjith", "lastName":"Yadav", "age":30} ] print(employes)  f = filter(lambda x : x['age']<30, employes) # lambda arguments : expression, iterables print(list(f))  [{'firstName': 'Sai', 'lastName': 'Kiran', 'age': 27}, {'firstName': 'Pradeep', 'lastName': 'Reddy', 'age': 29}, {'firstName': 'Praneeth', 'lastName': 'Reddy', 'age': 35}, {'firstName': 'Ranjith', 'lastName': 'Yadav', 'age': 30}]  [{'firstName': 'Sai', 'lastName': 'Kiran', 'age': 27}, {'firstName': 'Pradeep', 'lastName': 'Reddy', 'age': 29}] |