Lambda Functions or lambda Expressions

What is lambda function or lambda expression?
Lambda function is anonymous function.
A function which does not have any name is called anonymous function.
Lambda function is single line function.

Applications of lambda functions
1. Lambda functions are used when working higher order functions
2. Callback functions
3. Function as an argument

What is higher order function?
A function which receives input as another function to perform operation is called higher order function or callback function.

Syntax:
lambda [parameters]:expression/statement

lambda function is defined with parameters or without parameters.

Example:
```
a=lambda:print("Hello")
print(a)

a()
a()
a()
```

Output:
```
<function <lambda> at 0x000001CC0C5304A0>
Hello
Hello
Hello
```

Example:
```
add=lambda a,b:a+b
sub=lambda a,b:a-b
```

```
res1=add(10,5)
res2=sub(5,2)
print(res1,res2)
```

Output:
15 3

filter(function, iterable)
Construct an iterator from those elements of iterable for which function is true. iterable may be either a sequence, a container which supports iteration, or an iterator. If function is None, the identity function is assumed, that is, all elements of iterable that are false are removed.

Example:
```
list1=[1,2,3,4,5,6,7,8,9,10]

evenList=list(filter(lambda num:num%2==0,list1))
print(evenList)
oddList=list(filter(lambda num:num%2!=0,list1))
print(oddList)
```

Output:
[2, 4, 6, 8, 10]
[1, 3, 5, 7, 9]

Example:
```
strList=["abc","ABC","xyz","XYZ","PQR","mno"]
strList1=list(filter(str.isupper,strList))

print(strList)
print(strList1)
```

Output:
['abc', 'ABC', 'xyz', 'XYZ', 'PQR', 'mno']
['ABC', 'XYZ', 'PQR']

Example:
```
gradesList=[['naresh','A'],
            ['suresh','B'],
```

```
        ['kishore','C'],
        ['ramesh','A'],
        ['kiran','B']]

gradeListA=list(filter(lambda stud:stud[1]=='A',gradesList ))
gradeListB=list(filter(lambda stud:stud[1]=='B',gradesList))
gradeListC=list(filter(lambda stud:stud[1]=='C',gradesList))

print(gradesList)
print(gradeListA)
print(gradeListB)
print(gradeListC)
```

Output:

```
[['naresh', 'A'], ['suresh', 'B'], ['kishore', 'C'], ['ramesh', 'A'], ['kiran', 'B']]
[['naresh', 'A'], ['ramesh', 'A']]
[['suresh', 'B'], ['kiran', 'B']]
[['kishore', 'C']]
```

map(function, iterable, *iterables)

Return an iterator that applies function to every item of iterable, yielding the results. If additional iterables arguments are passed, function must take that many arguments and is applied to the items from all iterables in parallel. With multiple iterables, the iterator stops when the shortest iterable is exhausted.

Example:

```
 list1=[1,2,3,4,5]
list2=[10,20,30,40,50]

list3=list(map(lambda n1,n2:n1+n2,list1,list2))
print(list1)
print(list2)
print(list3)
```

Output:

```
[1, 2, 3, 4, 5]
```

```
[10, 20, 30, 40, 50]
[11, 22, 33, 44, 55]
```

Example:

```
marksList=[['naresh',50,60],['suresh',70,80],['ramesh',50,30]]

marksList1=list(map(lambda
stud:[stud[0],stud[1],stud[2],stud[1]+stud[2]],marksList))
print(marksList)
print(marksList1)
```

Output:

```
[['naresh', 50, 60], ['suresh', 70, 80], ['ramesh', 50, 30]]
[['naresh', 50, 60, 110], ['suresh', 70, 80, 150], ['ramesh', 50, 30, 80]]
```

functools.reduce(function, iterable[, initializer])

Apply function of two arguments cumulatively to the items of iterable, from left to right, so as to reduce the iterable to a single value. For example, reduce(lambda x, y: x+y, [1, 2, 3, 4, 5]) calculates ((((1+2)+3)+4)+5). The left argument, x, is the accumulated value and the right argument, y, is the update value from the iterable.

Example:

```
import functools
list1=[1,2,3,4,5]

res1=functools.reduce(lambda x,y:x+y,list1)
print(res1)
res2=functools.reduce(lambda x,y:x if x>y else y,list1)
print(res2)
res3=functools.reduce(lambda x,y:x if x<y else y,list1)
print(res3)
```

Output:

```
15
5
1
```

What is the difference between lambda function and regular function in Python?

In Python, lambda is an anonymous function and hence, is defined without a name. While normal functions are defined using the “def” keyword in Python, the anonymous functions are defined using the “lambda” keyword.

Modules and Packages