III B.Tech.

Computer Science & Engineering

CSE304: PYTHON PROGRAMMING WITH WEB FRAMEWORKS

Lambda Functions

By

Mrs. S. KAMAKSHI, AP-III / CSE

School of Computing

Anonymous Functions: lambda

- Lambda - an expression form that generates function objects
- Like def, this expression creates a function to be called later, but it returns the function instead of assigning it to a name.
- Lambdas are sometimes known as anonymous (i.e., unnamed) functions.
- General Form:
  - lambda argument1, argument2,... argumentN : expression using arguments
- Function objects returned by running lambda expressions work exactly the same as those created and assigned by defs

Example

- lambda is an expression, not a statement.
- lambda’s body is a single expression, not a block of statements.
- Eg.

```
def func(x, y, z):
        return x + y + z
```

- Same as

```
f = lambda x, y, z: x + y + z
f(2, 3, 4)
9
```

- lower = (lambda x, y: x if x < y else y)
- lower('bb', 'aa')
- 'aa
- With Default Arguments
  - x = (lambda a="fee", b="fie", c="foe": a + b + c)
  - x("wee")

Named Functions vs. Lambda Functions

- Named Function

```
def f1(x):
      return x ** 2
def f2(x):
      return x ** 3
def f3(x):
      return x ** 4
L = [f1, f2, f3]
for f in L:
     print(f(2))
print(L[0](3))
```

- Lambda function

```
L = [  lambda x: x ** 2,
          lambda x: x ** 3,
          lambda x: x ** 4]
for f in L:
      print(f(2))
print(L[0](3))
```

Named Functions vs. Lambda Functions

- Named Function

```
def f1(x): return x + x
def f2(x): return x * x
def f3(x): return x ** 4
D={‘f1': f1, ‘f2': f2, ‘f3': f3}
fname = ‘f3’
D[fname](4)
64
```

- Lambda function

```
D = {‘f1':     (lambda x: x + x),
     ‘f2':     (lambda x: x * x),
     ‘f3':     (lambda x: x ** 4)}
fname = ‘f1’
D[fname](4)
8
```

Nested Lambda Functions

- Nested lambda
  - My_function = (lambda x: (lambda y: x + y))
  - act = My_function(99)
  - act(3)
  - act(40)

- Direct calling of nested lambda function
  - ((lambda x: (lambda y: x + y))(99))(4)

Anonymous Function in Interactive Mode

- Nested lambda
  - My_function = (lambda x: (lambda y: x + y))
  - act = My_function(99)
  - act(3)
  - act(40)

- Direct calling of nested lambda function
  - ((lambda x: (lambda y: x + y))(99))(4)