Exercise 1:

Result = 3

Exercise 2:

add = lambda x, y: x + y

inc = lambda f, x: f(x,10)

print(inc(add, 2))

Exercise 3:

a)

add = lambda x, y: x + y

add10 = lambda f, x: f(x,10)

print(add10(add, 2)

b)

minus = lambda x, y: x - y

dec = lambda f, x: f(x, 1)

print(dec(minus, 2))

Exercise 4:

Result = 25

Exercise 5:

create\_adder = lambda x, y: x + y

print(add\_15(10))

Exercise 6:

a)

create\_lifter = lambda x, y: x \*\* y

b)

create\_lifter = lambda x, y: x \*\* y

area = lambda f, x: f(x, 2)

print(area(create\_lifter, 2))

Exercise 7:

1. Result = [2, 4]
2. Result = [2, -1, -1, 4]

Exercise 8:

a)

def even(l,i):

return [x for x in l if x % 2 == 0]

b)

def squared(l,i):

return [x \*\* 2 for x in l]

c)

def remove\_squared\_less\_100(l):

remv\_list = [x \*\* 2 for x in range(9)]

return [x for x in l if not(x in remv\_list)]

Exercise 9:

Result = [3, 4, 5]

Exercise 10:

l = list(map(lambda x:x+1, [2,3,4]))

print(l)

Exercise 11:

l = [2, 3, 4]

l = list(map(lambda x: x \*\* 2, l))

print(l)

Exercise 12:

Result = 3

Exercise 13:

def convert(l):

return list(map(int2C,l))

list\_C = convert([2,3,4])

print(list\_C)

print(" ".join(str(i) for i in list\_C))

Exercise 14:

def convert(l):

return reduce(lambda x, y: x + str(y) + " ", l[0:-1], "") + str(l[-1])

l = [2,3,4]

print(convert(l))

Exercise 15:

Result = Parent([C2,C3,C4])

Exercise 16:

def convert(l):

return reduce(lambda x, y: x + "C" + str(y) + ",", l[0:-1], "Parent([") + "C" + str(l[-1])+ "])"

l = [2,3,4]

print(convert(l))