**Higher Order Function and Lambda Calculus**

**Question 1.**

Run the following code, observe the displayed result.

def add (x,y):

return x+y

def inc(f,x):

return f(x,1)

print(inc(add,2))

**Answer:** 3

**Question 2.**

Rewrite inc(f,x) using lambda calculus

**Answer:**

def inc(f,x):

return (lambda x,y: x+y) (x,1)

**Question 3.**

Develop the following functions using the similar manner.

1. add10(x)

**Answer:**

def add10(x):

return x + 10

1. dec(f,x)

**Answer:**

def dec(f,x):

return (lambda x,y: x-y) (x,1)

**Question 4.**

Run the following code, observe the displayed result.

def create\_adder(x):

def adder(y):

return x + y

return adder

add\_15 = create\_adder(15)

print(add\_15(10))

**Answer:** 25

**Question 5.**

Rewrite create\_adder using lambda

**Answer:**

def create\_adder(x):

return (lambda y: x + y)

**Question 6.**

1. Develop create\_lifter(x) to return x^y

**Answer:**

def create\_lifter(x):

return (lambda y: x\*\*y)

1. Use create\_lifter to develop area(n), which returns the area of a square whose side length is n.

**Answer:**

def area(n):

return create\_lifter(n)(2)

**List comprehension**

**Question 7.**

Run the following code and observe the result

a.

def rem(l,i):

return [x for x in l if x !=i]

print(rem([2,3,3,4],3))

**Answer:** [2, 4]

b.

def rem(l,i):

return [x if x!=i else -1 for x in l]

print(rem([2,3,3,4],3))

**Answer:** [2, -1, -1, 4]

**Question 8.**

Using list comprehension to accomplish the following functions

1. Extract list of even numbers from a list

**Answer:**

def even(l):

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

1. Return a squared list of a list

**Answer:**

def sqr(l):

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

1. Remove all of square numbers less than 100 in a list

**Answer:**

def rem\_sqr(l):

return [x for x in l if x\*\*2 > 100]

**Using map**

**Question 9.**

Run the following code

def f(x):

return x+1

l = list(map(f,[2,3,4]))

print(l)

**Answer:** [3, 4, 5]

**Question 10.**

Rewrite Question 9 using lambda expression

**Answer:**

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

print(l)

**Question 11.**

Use map and lambda to convert a list of numbers into the squared list.

**Answer:**

l = list(map(lambda x:x\*x,[2,3,4]))

**Customized class**

**Question 12.**

Run the following code

class C:

#n:int

def \_\_init\_\_(self, k):

self.n = k

def \_\_str\_\_(self):

return str(self.n)

print(C(3))

**Answer:** 3

**Question 13.**

Convert a list of numbers into a list of C objects using *map*

class C:

#n:int

def \_\_init\_\_(self, k):

self.n = k

def \_\_str\_\_(self):

return str(self.n)

def int2C(x):

return C(x)

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))

**Question 14.**

Rewrite Question 13 using lambda

**Answer:**

def convert(l):

return list(map(lambda x: C(x),l))

**Question 15.**

Declare a class Parent whose children are of class C as follows. Run the code and observe the result.

class Parent:

#decl:list(C)

def \_\_init\_\_(self, decl):

self.decl = decl

def \_\_str\_\_(self):

return "Parent([C" + ',C'.join(str(i) for i in self.decl) + "])"

class C:

#n:int

def \_\_init\_\_(self, k):

self.n = k

def \_\_str\_\_(self):

return str(self.n)

def convert(l):

return list(map(lambda x: C(x),l))

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

p = Parent(list\_C)

print(p)

**Answer:** Parent([C2,C3,C4])

**Question 16.**

Write a function using map and lambda to convert a list of number into the corresponding Parent object

**Answer:**

def convert\_list\_to\_parent(l):

return Parent(list(map(lambda x: C(x),l))