**Assignment- OOP**

1. Write a Python program to create a class representing a stack data structure. Include methods for pushing and popping elements.

Nikhil:

class stack:

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

self.stack = []

def push(self,item):

self.stack.append(item)

print("Item pushed into stack")

def pop(self):

if(len(self.stack)==0):

print("stack is empty")

else:

item = self.stack.pop()

print("Item popped from stack")

return item

def display(self):

print("Item is stack :",self.stack)

stk = stack()

stk.push(1)

stk.push(2)

stk.push(3)

stk.display()

stk.pop()

stk.display()

1. Write a Python program to create a class representing a linked list data structure. Include methods for displaying linked list data, inserting and deleting nodes.

Nikhil:

class node:

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

self.data = data

self.next = None

class linklist:

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

self.head = None

def insert(self,data):

new\_node = node(data)

new\_node.next = self.head

self.head = new\_node

print("Data inserted into link list.")

def delete(self,key):

temp =self.head

if temp is not None:

if temp.data==key:

self.head =temp.next

temp = None

print("data deleted from link list")

return

prev = None

while temp is not None:

if temp.data ==key:

break

prev = temp

temp =temp.next

if temp is None:

print("data not found in linked list to delete")

return

prev.next = temp.next

temp = None

print("data deleted from linked list")

def display(self):

current = self.head

print("Linked list data :")

while current:

print(current.data,end ="-->")

current = current.next

print("None")

l = linklist()

l.insert(5)

l.insert(9)

l.insert(3)

l.insert(2)

l.insert(7)

l.display()

l.delete(3)

l.display()

1. Write a Python program to create a class representing a shopping cart. Include methods for adding and removing items, and calculating the total price.

class mycart():

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

self.items = {}

def add(self,item,price):

self.items[item] = price

print(f"{item} with price {price} added in cart.")

def remove(self,item):

if item in self.items:

del self.items[item]

print(f"{item} removed.")

else:

print(f"{item} is not present.")

def calTotal(self):

total = sum(self.items.values())

print(f"Total price of items is : {total}")

def display(self):

if not self.items:

print("empty cart")

else:

print("Items in the cart")

for item,price in self.items.items():

print(f"{item} : {price}")

c = mycart()

c.add("apple",34)

c.add("banana",23)

c.add("orange",50)

c.display()

c.calTotal()

c.remove("banana")

c.display()

c.calTotal()

1. Write a Python program to create a class representing a stack data structure. Include methods for pushing, popping and displaying elements.

Nikhil : It is same like above question 1

**Assignment**

**1.** Write a Python program to create a lambda function that adds 15 to a given number passed in as an argument, also create a lambda function that multiplies argument x with argument y and prints the result.  
Sample Output:  
25  
48

Nikhil:

num = lambda x :x+15

print( num(10))

multi = lambda x, y: x\*y

print(multi(4,12))

**2.** Write a Python program to create a function that takes one argument, and that argument will be multiplied with an unknown given number.  
Sample Output:  
Double the number of 15 = 30  
Triple the number of 15 = 45  
Quadruple the number of 15 = 60  
Quintuple the number 15 = 75

Nikhil :

def multi(val):

return lambda x: x\*val

double = multi(2)

triple = multi(3)

quadruple = multi(4)

quintuple = multi(5)

print("Double the number of 15 :", double(15))

print("Triple the number of 15 :", triple(15))

print("Quadruple the number of 15:", quadruple(15))

print("Quintuple the number 15 :", quintuple(15))

**3.** Write a Python program to sort a list of tuples using Lambda.  
Original list of tuples:  
[('English', 88), ('Science', 90), ('Maths', 97), ('Social sciences', 82)]  
Sorting the List of Tuples:  
[('Social sciences', 82), ('English', 88), ('Science', 90), ('Maths', 97)]

Nikhil:

t1 = [('English', 88), ('Science', 90), ('Maths', 97), ('Social sciences', 82)]

t2 = sorted (t1,key=lambda x : x[1])

print ("Original list of tuples:")

print(t1)

print ("Sorted list of tuples:")

print(t2)

**4.** Write a Python program to sort a list of dictionaries using Lambda.  
Original list of dictionaries :  
[{'make': 'Nokia', 'model': 216, 'color': 'Black'}, {'make': 'Mi Max', 'model': '2', 'color': 'Gold'}, {'make': 'Samsung', 'model': 7, 'color': 'Blue'}]  
Sorting the List of dictionaries :  
[{'make': 'Nokia', 'model': 216, 'color': 'Black'}, {'make': 'Samsung', 'model': 7, 'color': 'Blue'}, {'make': 'Mi Max', 'model': '2', 'color': 'Gold'}]

Nikhil:

phones = [

{'make': 'Nokia', 'model': 216, 'color': 'Black'},{'make': 'Mi Max', 'model': '2', 'color': 'Gold'},{'make': 'Samsung', 'model': 7, 'color': 'Blue'}

]

sorted\_phones = sorted(phones, key=lambda x: x['make'])

print("Sorted List of Dictionaries:")

print(sorted\_phones)

1. Write a Python program to filter a list of integers using Lambda.  
   Original list of integers:  
   [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]  
   Even numbers from the said list:  
   [2, 4, 6, 8, 10]  
   Odd numbers from the said list:  
   [1, 3, 5, 7, 9]

Nikhil:

original\_list = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

even\_numbers = list(filter(lambda x: x % 2 == 0, original\_list))

odd\_numbers = list(filter(lambda x: x % 2 != 0, original\_list))

print("Original list of integers:")

print(original\_list)

print("Even numbers from the said list:")

print(even\_numbers)

print("Odd numbers from the said list:")

print(odd\_numbers)

**6.** Write a Python program to square and cube every number in a given list of integers using Lambda.  
Original list of integers:  
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]  
Square every number of the said list:  
[1, 4, 9, 16, 25, 36, 49, 64, 81, 100]  
Cube every number of the said list:  
[1, 8, 27, 64, 125, 216, 343, 512, 729, 1000]

Nikhil:

original\_list = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

squared\_list = list(map(lambda x: x\*\*2, original\_list))

print("Square every number of the said list:")

print(squared\_list)

cubed\_list = list(map(lambda x: x\*\*3, original\_list))

print("Cube every number of the said list:")

print(cubed\_list)

**7.** Write a Python program to find if a given string starts with a given character using Lambda.  
Sample Output:  
True  
False

Nikhil:

starts\_with = lambda s, c: s.startswith(c)

print(starts\_with("Hello, World!", "H")) # Output: True

print(starts\_with("Goodbye!", "H"))

**8.** Write a Python program to extract year, month, date and time using Lambda.  
Sample Output:  
2020-01-15 09:03:32.744178  
2020  
1  
15  
09:03:32.744178

Nikhil:

from datetime import datetime

date\_time = datetime.now()

year = lambda x: x.year

month = lambda x: x.month

day = lambda x: x.day

time = lambda x: x.strftime('%H:%M:%S.%f')

print(date\_time)

print(year(date\_time))

print(month(date\_time))

print(day(date\_time))

print(time(date\_time))

**9.** Write a Python program to check whether a given string is a number or not using Lambda.  
Sample Output:  
True  
True  
False  
True  
False  
True  
Print checking numbers:  
True  
True

Nikhil:

is\_number = lambda x: str(x).isnumeric()

print(is\_number(123))

print(is\_number("123"))

print(is\_number("123.45"))

print(is\_number(-123))

print(is\_number("+123"))

print(is\_number(-123.45))

**10.** Write a Python program to create Fibonacci series up to n using Lambda.  
Fibonacci series upto 2:  
[0, 1]  
Fibonacci series upto 5:  
[0, 1, 1, 2, 3]  
Fibonacci series upto 6:  
[0, 1, 1, 2, 3, 5]  
Fibonacci series upto 9:  
[0, 1, 1, 2, 3, 5, 8, 13, 21]

Nikhil:

def fibonacci\_series(n):

fib\_series = []

fib = lambda x: x if x < 2 else fib(x - 1) + fib(x - 2)

for i in range(n):

fib\_series.append(fib(i))

return fib\_series

for n in [2, 5, 6, 9]:

print(f"Fibonacci series up to {n}: {fibonacci\_series(n)}")

**11.** Write a Python program to find the intersection of two given arrays using Lambda.  
Original arrays:  
[1, 2, 3, 5, 7, 8, 9, 10]  
[1, 2, 4, 8, 9]  
Intersection of the said arrays: [1, 2, 8, 9]

Nikhil:

array1 = [1, 2, 3, 5, 7, 8, 9, 10]

array2 = [1, 2, 4, 8, 9]

intersection = list(filter(lambda x: x in array2, array1))

print("Intersection of the said arrays:", intersection)