**Assignment- OOP**

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

Ans:-

**class mydata:**

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

**self.data = data**

**def push(self,item):**

**self.data.append(item)**

**print(f"Pushed {item} to mydata", self.data)**

**def pop(self):**

**data = self.data.pop()**

**print(f"Poped {data} from mydata",self.data)**

**mydata = mydata([1,2,3,4,5])**

**mydata.push(7)**

**mydata.pop()**

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.

Ans:

**class Node:**

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

**self.data = data**

**self.next = None**

**class List:**

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

**self.head = None**

**def display(self):**

**nodes = self.head**

**while nodes:**

**print(nodes.data)**

**nodes = nodes.next**

**def insert(self,data):**

**new\_node = Node(data)**

**if self.head is None:**

**self.head = new\_node**

**return**

**last\_node = self.head**

**while last\_node.next:**

**last\_node = last\_node.next**

**last\_node.next = new\_node**

**def delete(self,key):**

**current = self.head**

**if current and current.data == key:**

**self.head = current.next**

**current = None**

**return**

**prev = None**

**while current and current.data != key:**

**prev = current**

**current = current.next**

**if current is None:**

**print(f"Node with value {key} not found.")**

**return**

**prev.next = current.next**

**current = None**

**list = List()**

**list.display()**

**list.insert(2)**

**list.insert(12)**

**list.insert(22)**

**list.insert(23)**

**list.insert(13)**

**list.delete(13)**

**list.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 shopingcart():**

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

**self.items=[]**

**def additem(self,name,quantity,price):**

**self.items.append({**

**'name': name,**

**'quantity': quantity,**

**'price': price**

**})**

**print(f"Added item: {name} , Quantity: {quantity} ,Price: {price} to the cart.")**

**def display(self):**

**totalprice = 0**

**if not self.items:**

**print("Shopping cart is empty.")**

**return**

**for item in self.items:**

**data = item**

**totalprice = item['quantity'] \* item['price']**

**print(f"'Name:'{item['name']},'Quantity:'{item['quantity']},'Price:'{item['price']},'Totalprice:'{totalprice}")**

**def delete(self,item\_name):**

**for item in self.items:**

**if item['name'] == item\_name:**

**self.items.remove(item)**

**print(f"deleted {item\_name} from the cart.")**

**return**

**print(f"Item {item\_name} not found in the cart.")**

**def total(self):**

**subtotal = 0**

**total = 0**

**for item in self.items:**

**subtotal = item['quantity'] \* item['price']**

**total += subtotal**

**print(f"Total price is {total}")**

**shopingcart= shopingcart()**

**shopingcart.additem("Book",10,45)**

**shopingcart.additem("Pen",20,20)**

**shopingcart.additem("Pencil",10,10)**

**shopingcart.display()**

**shopingcart.delete('Book')**

**shopingcart.total()**

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

Ans:

**class mydata:**

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

**self.data = []**

**def push(self,item):**

**self.data.append(item)**

**print(f"Pushed {item} to mydata")**

**def pop(self):**

**data = self.data.pop()**

**print(f"Poped {data} from mydata")**

**def display(self):**

**return self.data**

**mydata = mydata()**

**mydata.push(11) #Pushed 11 to mydata**

**mydata.display()**

**mydata.pop()**

**mydata.display()**