Msp

Ai Task 2

```
# first >> creat a node
class Node:
    def __init__(self, data):
        self.data = data
        self.next = None


class LinkedList:

    def __init__(self):
        self.head = None

# Insert
    def insertAtBeginning(self, new):
        new_node = Node(new)

        new_node.next = self.head
        self.head = new_node

    # Inserting
    def insertAfter(self, prev_node, new):

        if prev_node is None:
            print("eror.")
            return

        new_node = Node(new)
        new_node.next = prev_node.next
        prev_node.next = new_node

```

```
        last = self.head
        while (last.next):
            last = last.next

        last.next = new_node

    # Deleting a node
    def deleteNode(self, position):

        if self.head is None:
            return

        temp = self.head

        if position == 0:
            self.head = temp.next
            temp = None
            return


        for i in range(position - 1):
            temp = temp.next
            if temp is None:
                break

        # If the key is not present
        if temp is None:
            return

        if temp.next is None:
            return

        next = temp.next.next

        temp.next = None

        temp.next = next

```

```
In [28]: runfile('D:/pics/screenshoots/code/linked list.py', wdir='D:/pics/screenshoots/
code')
linked list:
3 2 5 1 4
_____________________________________________________________________________

After deleting an element:
3 2 5 4
3 is found
Sorted List:
2 3 4 5

In [29]:
```

```
124  if __name__ == '__main__':
125
126      llist = LinkedList()
127      llist.insertAtEnd(1)
128      llist.insertAtBeginning(2)
129      llist.insertAtBeginning(3)
130      llist.insertAtEnd(4)
131      llist.insertAfter(llist.head.next, 5)
132
133      print('linked list:')
134      llist.printList()
135      print("_" *80)
136      print("\nAfter deleting an element:")
137      llist.deleteNode(3)
138      llist.printList()
139      print()
140      item_to_find = 3
141      if llist.search(item_to_find):
142          print(str(item_to_find) + " is found")
143      else:
144          print(str(item_to_find) + " is not found")
145
146      llist.sortLinkedList(llist.head)
147      print("Sorted List: ")
148      llist.printList()
```

```
84      #to search for an elemnt
85      def search(self, key):
86
87          current = self.head
88
89          while current is not None:
90              if current.data == key:
91                  return True
92
93              current = current.next
94
95          return False
96
97
98      def sortLinkedList(self, head):
99          current = head
100         index = Node(None)
101
102         if head is None:
103             return
104         else:
105             while current is not None:
106
107                 index = current.next
108
109                 while index is not None:
110                     if current.data > index.data:
111                         current.data, index.data = index.data, current.data
112
113                     index = index.next
114                 current = current.next
115
116
117     def printList(self):
118         temp = self.head
119         while (temp):
120             print(str(temp.data) + " ", end="")
121             temp = temp.next
```