Single Linked List Implementation

```
class Node:
    def __init__(self,data):
        self.data = data
        self.next = None

class SLL:
    def __init__(self):
        self.head = None

    def insert(self,data):
        new_node = Node(data)
        if(self.head is None):
            self.head = new_node
            return
        current = self.head
        while(current.next is not None):
            current = current.next
        current.next = new_node

    def insertFirst(self,data):
        new_node = Node(data)
        new_node.next = self.head
        self.head = new_node

    def insertAtPosition(self,pos,data): #1 2 3 4
        new_node = Node(data)
        if(pos == 0):
            new_node.next = self.head
            self.head = new_node
        current = self.head
        pointer = 0
        while(current and pointer < pos - 1):
            current = current.next
            pointer += 1
        new_node.next = current.next
        current.next = new_node

    def delete_at_position(self,pos):
        if(self.head is None):
            return("List is Empty...")
        if(pos == 0):
            delete_node = self.head
            self.head = self.head.next
            delete_node.next = None
            return("Deletion successful...")
        current = self.head
        pointer = 0
        while(current and pointer < pos - 1):
            current = current.next
            pointer += 1
        delete_node = current.next
        current.next = delete_node.next
        delete_node.next = None
        return "Deletion successful..."



    def print_list(self):
        current = self.head
        while(current):
            print(current.data,end = "-->")
            current = current.next
        print()

    def print_reverse_list(self):
        lst = []
        current = self.head
        while(current is not None):
            lst.append(current.data)
            current = current.next
        print(lst[::-1])

    def size_of_list(self):
        count = 0
        current = self.head
        while(current is not None):
            count+=1
            current = current.next
        print(count)

    def search_element(self,key):
        if(self.head is None):
            return "List is empty..."
        current = self.head
        pos = -1
        while(current is not None):
            pos+=1
            if(current.data == key):
                return pos
            current = current.next
        return -1



sll = SLL()
sll.insert(12)
sll.insert(123)
sll.insert(24)
sll.insert(33)
sll.insert(29)
sll.print_list()
sll.insertAtPosition(2,52)
sll.print_list()
sll.print_reverse_list()
sll.size_of_list()
sll.delete_at_position(3)
sll.print_list()
sll.size_of_list()
print(sll.search_element(111))
```