(3.3) List

1. Introduction

If we want to represent a group of individual objects as a single entity where insertion order preserved and duplicates are allowed, then we should go for List.

*itself mutable data type but can store mutable and immutable

*it store references of object

*insertion order preserved.

*duplicate objects are allowed

*heterogeneous objects are allowed.

*List is dynamic because based on our requirement we can increase the size and decrease the size.

*In List the elements will be placed within square brackets and with comma seperator.

*We can differentiate duplicate elements by using index and we can preserve insertion order by using index. Hence index will play very important role.

*Python supports both positive and negative indexes. +ve index means from left to right

*where as negative index means right to left

Example

In [24]:
```
# 1. Homogeneous List (int)
nums = [ 10, 30, 70, 56, 78, 100 ]

# 2. Homogeneous List (string)
data = [ "sachin", 'rajat', 'kushal', 'simran']

# 3. Hetrogeneous List
info = [ "sachin", 90, "rajat", 100 ]

# 4. Nested List
data1 = [
    [ 'ram', 45, 67],
    ['shyam', 65, 78],
    ['hari', 40, 60],
    ['meera', 56, 65]
]

# 5. Nested List
data2 = [ 'hello', 'hi', [['hi', ['hello', ['hey','Arya']]]],'bye', 'good bye']
```

In [25]:
```
print(type(nums))
print(type(data))
```

```
print(type(info))
print(type(data1))
print(type(data2))
```

```
<class 'list'>
<class 'list'>
<class 'list'>
<class 'list'>
<class 'list'>
```

In [26]:

```
print(nums)
print(data)
print(info)
print(data1)
print(data2[2][0][1][1][1])
```

```
[10, 30, 70, 56, 78, 100]
['sachin', 'rajat', 'kushal', 'simran']
['sachin', 90, 'rajat', 100]
[['ram', 45, 67], ['shyam', 65, 78], ['hari', 40, 60], ['meera', 56, 65]]
Arya
```

2. Creation of List

Empty list

In [27]:

```
l = []
print(l,type(l))
```

```
[] <class 'list'>
```

With element

In [28]:

```
l = [10,20,30,40]
print(l,type(l))
```

```
[10, 20, 30, 40] <class 'list'>
```

with dynamic Input

In [29]:

```
#Enter element with []
l = eval(input("Enter List: "))
print(l,type(l))
```

```
Enter List: [10,20,30,40]
[10, 20, 30, 40] <class 'list'>
```

with single line input as one string

In [30]:

```
l1 = list(map(int, input("Enter list element in single line : ").split()))
print(l1)
```

```
Enter list element in single line : 10 20 30 40
[10, 20, 30, 40]
```

wiht range

In [31]:

```
l=list(range(0,10,2))
print(l)
```

```
[0, 2, 4, 6, 8]
```

In [32]:
```
s = "pankaj"
l = list(s)
print(l)
```

```
['p', 'a', 'n', 'k', 'a', 'j']
```

3. Accessing Element of list

By using index

List follows zero based index. ie index of first element is zero.
List supports both +ve and -ve indexes.
+ve index meant for Left to Right
-ve index meant for Right to Left

In [33]:
```
l = [10,20,30,40]
```

In [34]:
```
print(l[0])
print(l[-1])
print(l[10])
```

```
10
40
---------------------------------------------------------------------------
IndexError                                Traceback (most recent call last)
C:\Users\PANKAJ~1\AppData\Local\Temp/ipykernel_1272/3231153371.py in <module>
      1 print(l[0])
      2 print(l[-1])
----> 3 print(l[10])

IndexError: list index out of range
```

By using slicing

list2 = list1[start:stop:step]

start ==>it indicates the index where slice has to start default value is 0
stop   ==>It indicates the index where slice has to end default value is max allowed index of list ie length of the list
step ==>increment value default value is 1

In [35]:
```
n=[1,2,3,4,5,6,7,8,9,10]
```

In [36]:
```
print(n[2:7:2])
print(n[4::2])
print(n[3:7])
print(n[8:2:-2])
print(n[4:100])
```

```
[3, 5, 7]
[5, 7, 9]
[4, 5, 6, 7]
[9, 7, 5]
[5, 6, 7, 8, 9, 10]
```

4. List Vs mutability

Once we creates a list object, we can modify its content. Hence List Object are mutable.

In [37]:
```
n=[10,20,30,40]
print(n)
n[1]=777
print(n)
```

```
[10, 20, 30, 40]
[10, 777, 30, 40]
```

In [38]:
```
lst = ['java','c','python','go']
```

In [39]:
```
lst[1] = 'c++'
```

In [40]:
```
print(lst)
```

```
['java', 'c++', 'python', 'go']
```

We can't change in string , tuple

In [41]:
```
t = ( 'java', 'c', 'python', 'go')
t[1] = 'c++'
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
C:\Users\PANKAJ~1\AppData\Local\Temp/ipykernel_1272/1215615815.py in <module>
      1 t = ( 'java', 'c', 'python', 'go')
----> 2 t[1] = 'c++'

TypeError: 'tuple' object does not support item assignment
```

5. Operation on list

In [42]:
```
print(dir(lst))
```

```
['__add__', '__class__', '__contains__', '__delattr__', '__delitem__', '__dir__', '__doc__', '__eq__', '__format__', '__ge__', '__getattribute__', '__getitem__', '__gt__', '__hash__', '__iadd__', '__imul__', '__init__', '__init_subclass__', '__iter__', '__le__', '__len__', '__lt__', '__mul__', '__ne__', '__new__', '__reduce__', '__reduce_ex__', '__repr__', '__reversed__', '__rmul__', '__setattr__', '__setitem__', '__sizeof__', '__str__', '__subclasshook__', 'append', 'clear', 'copy', 'count', 'extend', 'index', 'insert', 'pop', 'remove', 'reverse', 'sort']
```

A. How to add element

```
list.append(item)
list.insert(loc, item)
list.extend(iterables)
```

In [43]:
```
lang = ["C","C++"]
```

In [44]:
```
#append
lang.append("Java")
print(lang)
```

```
['C', 'C++', 'Java']
```

In [45]:
```
#append
lang.append('ruby')
print(lang)
```

```
['C', 'C++', 'Java', 'ruby']
```

In [46]:
```
#insert
lang.insert(2,'Python')
print(lang)
```

```
['C', 'C++', 'Python', 'Java', 'ruby']
```

Note: If the specified index is greater than max index then element will be inserted at last position.

If the specified index is smaller than min index then element will be inserted at first position.

In [47]:
```
#extend
lang.extend(['go','swift','.net'])
print(lang)
```

```
['C', 'C++', 'Python', 'Java', 'ruby', 'go', 'swift', '.net']
```

In [ ]:

In [48]:
```
lst = []

lst.append(['java','c'])
print(lst)

lst.append('python')
print(lst)

lst.append({'name':'sachin','age': 26})
print(lst)
```

```
[['java', 'c']]
[['java', 'c'], 'python']
[['java', 'c'], 'python', {'name': 'sachin', 'age': 26}]
```

In [ ]:

In [49]:
```
lst = []

#int is  not iterable
lst.extend(1234)

print(lst)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
C:\Users\PANKAJ~1\AppData\Local\Temp/ipykernel_1272/3805288660.py in <module>
      2
      3 #int is  not iterable
```

```
----> 4 lst.extend(1234)
      5
      6 print(lst)

TypeError: 'int' object is not iterable
```

In [50]:
```
lst = []

lst.extend('1234')

print(lst)
```

```
['1', '2', '3', '4']
```

In [ ]:
```

```

In [51]:
```
lst = []

lst.extend([["python","python"],'java','c','c++'])
print(lst)
```

```
[['python', 'python'], 'java', 'c', 'c++']
```

In [ ]:
```

```

In [52]:
```
lst = []
lst.append('java')
lst.insert(0, 'ruby')
lst.insert(1,'perl')
lst.extend(['c','c++','swift'])
lst.insert(3,'go')
lst.append(['.net','php'])
lst.extend('python')
lst.append('dart')
lst.insert(1,'python')
print(lst)
```

```
['ruby', 'python', 'perl', 'java', 'go', 'c', 'c++', 'swift', ['.net', 'php'], 'p', 'y',
't', 'h', 'o', 'n', 'dart']
```

In [53]:
```
from pprint import pprint
```

In [54]:
```
pprint(lst)
```

```
['ruby',
 'python',
 'perl',
 'java',
 'go',
 'c',
 'c++',
 'swift',
 ['.net', 'php'],
 'p',
 'y',
 't',
 'h',
 'o',
```

```
 'n',
 'dart']
```

In [ ]:

In [55]:
```
nl = [ [1, 2, 3], [4, 5, 6], [7, 8,9 ], [10, 11, 12]]
#print(nl)
pprint(nl, width=20)
```

```
[[1, 2, 3],
 [4, 5, 6],
 [7, 8, 9],
 [10, 11, 12]]
```

B. How to delete elements from a list

```
item = list.pop() - will delete last element from list

item = list.pop(loc) - will delete item at index loc from list

list.remove(item) - if item in list than delete first occurence of item from list
otherwise throw an ValueError
```

In [67]:
```
lang = ['java','c','c++','ruby','perl']
```

In [68]:
```
item = lang.pop()
print(f"{item} has been deleted from list \n{lang}")
item = lang.pop(2)
print(item)
print(lang)
```

```
perl has been deleted from list
['java', 'c', 'c++', 'ruby']
c++
['java', 'c', 'ruby']
```

This is only function which manipulates list and returns some element.

If the list is empty then pop() function raises IndexError

In [102…
```
n = []
print(n.pop())
```

```
---------------------------------------------------------------------------
IndexError                                Traceback (most recent call last)
C:\Users\PANKAJ~1\AppData\Local\Temp/ipykernel_1272/3055743881.py in <module>
      1 n = []
----> 2 print(n.pop())

IndexError: pop from empty list
```

In [ ]:

In [69]:
```
lang.insert(1,"python")
lang.insert(3,"python")
lang.extend(['c++','perl','python'])
print(lang)
```

```
['java', 'python', 'c', 'python', 'ruby', 'c++', 'perl', 'python']
```

In [70]:
```
lang.remove('python')
print(lang)
```

```
['java', 'c', 'python', 'ruby', 'c++', 'perl', 'python']
```

Note: Hence before using remove() method first we have to check specified element present in the list or not by using in operator.

C. Count()

In [74]:
```
from random import randint
nums = [randint(1,5) for _ in range(15)]
print(nums)
```

```
[2, 1, 5, 5, 3, 3, 4, 3, 4, 4, 3, 2, 1, 4, 4]
```

`list.count(item)` - will return frequency of item in list

In [75]:
```
#delete all 5 from the list

print(nums)
item = 5
c = nums.count(item)
for _ in range(c):
    nums.remove(item)

print(nums)
```

```
[2, 1, 5, 5, 3, 3, 4, 3, 4, 4, 3, 2, 1, 4, 4]
[2, 1, 3, 3, 4, 3, 4, 4, 3, 2, 1, 4, 4]
```

D. Index()

`i = list.index(item)` - will return first index of item

In [76]:
```
lang = ['java',"python",'c','c++','ruby','python']
```

In [77]:
```
i = lang.index('python') # index(item,start,end)
print(i)

i = lang.index("python",2)
print(i)
```

```
1
5
```

Note: If the specified element not present in the list then we will get ValueError.Hence before index() method we have to check whether item present in the list or not by using in operator.

E. clear()

`list.clear()` - will delete each and every element from list

In [79]:
```
nums
```

Out[79]:
```
[2, 1, 3, 3, 4, 3, 4, 4, 3, 2, 1, 4, 4]
```

In [80]:
```
nums.clear()
print(nums)
```

```
[]
```

F. len()

In [81]:
```
n = [10,20,30,40,50]
print(len(n))
```

```
5
```

G. sort list

note - list should be homogenous to sort

list.sort() - will arrange item in ascending order wrt values

list.sort(reverse=True) - will arrange item in descending order

In [82]:
```
nums = [randint(1,10) for _ in range(20)]
print(nums)
```

```
[2, 10, 10, 10, 1, 3, 3, 4, 3, 1, 6, 5, 8, 7, 8, 9, 6, 5, 9, 10]
```

In [83]:
```
nums.sort()
print(nums)

nums.sort(reverse=True)
print(nums)
```

```
[1, 1, 2, 3, 3, 3, 4, 5, 5, 6, 6, 7, 8, 8, 9, 9, 10, 10, 10, 10]
[10, 10, 10, 10, 9, 9, 8, 8, 7, 6, 6, 5, 5, 4, 3, 3, 3, 2, 1, 1]
```

In [ ]:
```

```

In [84]:
```
names = ['sachin','rajat','nidhi','simran','ravi','gaurav']
names.sort()
print(names)
```

```
['gaurav', 'nidhi', 'rajat', 'ravi', 'sachin', 'simran']
```

In [ ]:
```

```

In [85]:
```
data = [
    ['sachin', 80],
    ['rajat', 85],
    ['simran',90],
    ['kushal',60],
    ['yadvendra',65]
]

data.sort()
data
```

```
[['kushal', 60],
```

Out[85]:
```
['rajat', 85],
['sachin', 80],
['simran', 90],
['yadvendra', 65]]
```

In [ ]:
```
```

In [86]:
```
from operator import itemgetter
f = itemgetter(2)
```

In [93]:
```
lst = ['python','java','c++','ruby']
f(lst)
```

Out[93]:
```
'c++'
```

In [89]:
```
f('hello')
```

Out[89]:
```
'l'
```

In [ ]:
```
```

In [90]:
```
data = [
    [ 'sachin', 80],
    [ 'rajat',  85 ],
    [ 'simran', 90],
    [ 'kushal', 60],
    ['yadvendra', 65]
]
```

In [94]:
```
data.sort(reverse=True,key=itemgetter(1))
pprint(data)
```

```
[['simran', 90],
 ['rajat', 85],
 ['sachin', 80],
 ['yadvendra', 65],
 ['kushal', 60]]
```

Note: In Python 2 if List contains both numbers and Strings then sort() function first sort numbers followed by strings

```
Eg :
    n = [20,"B",10,"A"]
    n.sort()
    print(n) #[10,20,'A','B']
```

H. Reverse list

`list.reverse()` - reverse string inplace

In [74]:
```
lst = ['java','c','c++','python']
```

In [95]:
```
# new_list = lst[::-1]
```

```
lst.reverse()
print(lst)
```

```
['ruby', 'c++', 'java', 'python']
```

I. Copy list

Aliasing and Cloning of List objects:-

The process of giving another reference variable to the existing list is called aliasing.

The problem in this approach is by using one reference variable if we are changing content,then those changes will be reflected to the other reference variable.

In [96]:
```
lst = [ 1, 2, 3, 4]

copy = lst #same list will be refrenced by copy

lst.append(5)

item = copy.pop(0)
```

In [97]:
```
print(lst) # ?
print(copy) # ?
```

```
[2, 3, 4, 5]
[2, 3, 4, 5]
```

To overcome this problem we should go for cloning.

The process of creating exactly duplicate independent object is called cloning.

We can implement cloning by using slice operator or by using copy() function

In [99]:
```
lst = [ 1, 2, 3, 4]

copy = lst[:] # another object will be create for copy
lst.append(5)

item = copy.pop(0)

print(lst) # ?

print(copy) # ?
```

```
[1, 2, 3, 4, 5]
[2, 3, 4]
```

In [ ]:
```

```

In [100…
```
lst = [ 1, 2, 3, 4]

copy = lst.copy() # shallow copy
lst.append(5)

item = copy.pop(0)

print(lst) # ?
```

```
print(copy) # ?
```

```
[1, 2, 3, 4, 5]
[2, 3, 4]
```

6. Using Mathematical operators for List Objects

We can use + and * operators for List objects.

A. Concatenation operator(+)

In [104…

```
a = [10,20,30]
b = [40,50,60]
c = a+b
print(c)
```

```
[10, 20, 30, 40, 50, 60]
```

Note: To use + operator compulsory both arguments should be list objects,otherwise we will get TypeError.

In [105…

```
c = 10 + [10,20,30]
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
C:\Users\PANKAJ~1\AppData\Local\Temp/ipykernel_1272/2470199193.py in <module>
----> 1 c = 10 + [10,20,30]

TypeError: unsupported operand type(s) for +: 'int' and 'list'
```

B. Repetition Operator(*)

In [106…

```
x = [10,20,30]
y = x*3
print(y)
```

```
[10, 20, 30, 10, 20, 30, 10, 20, 30]
```

7. Comparing List objects

In [107…

```
x=["Dog","Cat","Rat"]
y=["Dog","Cat","Rat"]
z=["DOG","CAT","RAT"]

print(x==y) #True
print(x==z) #False
print(x != z) #True
```

```
True
False
True
```

Note:

Whenever we are using comparison operators(==,!=) for List objects then the following

should be considered
The number of elements
The order of elements
The content of elements (case sensitive)

Note:

When ever we are using relatational operators(<,<=,>,>=) between List objects,only first element comparison will be performed.

In [108…

```
x=[50,20,30]
y=[40,50,60,100,200]

print(x>y) #True
print(x>=y) #True
print(x<y) #False
print(x<=y) #False
```

```
True
True
False
False
```

In [109…

```
x=["Dog","Cat","Rat"]
y=["Rat","Cat","Dog"]

print(x>y) #False
print(x>=y) #False
print(x<y) #True
print(x<=y) #True
```

```
False
False
True
True
```

8. Membership operators

in operator
not in operator

In [110…

```
n=[10,20,30,40]

print(10 in n)
print(10 not in n)
print(50 in n)
print(50 not in n)
```

```
True
False
False
True
```

9. Matrix

In [112…

```
n=[[10,20,30],[40,50,60],[70,80,90]]

print(n)
```

```
print("Elements by Row wise:")
for r in n:
    print(r)

print("Elements by Matrix style:")
for i in range(len(n)):
    for j in range(len(n[i])):
        print(n[i][j],end=' ')
    print()
```

```
[[10, 20, 30], [40, 50, 60], [70, 80, 90]]
Elements by Row wise:
[10, 20, 30]
[40, 50, 60]
[70, 80, 90]
Elements by Matrix style:
10 20 30
40 50 60
70 80 90
```

10. List Comprehension

It is very easy and compact way of creating list objects from any iterable objects(like list,tuple,dictionary,range etc) based on some condition.

list = [expression for item in list if condition]

In [113…

```
s=[ x*x for x in range(1,11)]
print(s)

v=[2**x for x in range(1,6)]
print(v)

m=[x for x in s if x%2==0]
print(m)
```

```
[1, 4, 9, 16, 25, 36, 49, 64, 81, 100]
[2, 4, 8, 16, 32]
[4, 16, 36, 64, 100]
```

In [ ]:

In [114…

```
words=["Balaiah","Nag","Venkatesh","Chiranjeevi"]
l=[w[0] for w in words]
print(l)
```

```
['B', 'N', 'V', 'C']
```

In [ ]:

In [115…

```
words="the quick brown fox jumps over the lazy dog".split()
print(words)

l=[[w.upper(),len(w)] for w in words]
print(l)
```

```
['the', 'quick', 'brown', 'fox', 'jumps', 'over', 'the', 'lazy', 'dog']
[['THE', 3], ['QUICK', 5], ['BROWN', 5], ['FOX', 3], ['JUMPS', 5], ['OVER', 4], ['THE',
3], ['LAZY', 4], ['DOG', 3]]
```

In [ ]:

In [124…

```
arr = [ (i, j)
        for i in range(1, 3)
        for j in range(2)
    ]

print(arr)
```

```
[(1, 0), (1, 1), (2, 0), (2, 1)]
```

In [ ]:

In [118…

```
lst = [ print(f"hello {i}") for i in range(1, 6) ]
print(lst)
```

```
hello 1
hello 2
hello 3
hello 4
hello 5
[None, None, None, None, None]
```

With enumerate

```
range(5) -> [0, 1, 2, 3, 4]

enumerate(range(5))

enumerate([0, 1, 2, 3, 4])

(0, 0), (1, 1), (2, 2), (3, 3), (4, 4)
```

In [119…

```
def func(a, b):
    return a**2 + b

lst = [func(i, j) for i,j in enumerate(range(5))]

print(lst)
```

```
[0, 2, 6, 12, 20]
```

Q. write a code to make a list with 10 random numbers in it

In [116…

```
from random import randint

n = 10 #int(input("n: "))
lst = []
for _ in range(n):
    lst.append(randint(1, 50))
print(lst)
```

```
[1, 30, 12, 32, 6, 6, 50, 11, 5, 42]
```

In [117…

```
lst = [ randint(1, 50) for _ in range(10) ]
print(lst)
```

```
[39, 40, 42, 37, 30, 10, 25, 18, 46, 21]
```

Q. make a list of prime number in range 1 to 100

In [120…
```
def prime(num):
    if num <= 1:
        return False
    if num <= 3:
        return True
    if num % 2 == 0:
        return False
    for check in range(2, num//2):
        if num % check == 0:
            return False
    return True
```

In [121…
```
arr = [ num for num in range(1, 101)
       if prime(num)]
```

In [122…
```
print(arr)
```

```
[2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79, 83, 8
9, 97]
```

Q. Print HIndi alphabet in a list

In [127…
```
print([chr(var) for var in range(2309, 2350)])
```

```
['अ', 'आ', 'इ', 'ई', 'उ', 'ऊ', 'ऋ', 'ऌ', 'ऍ', 'ऎ', 'ए', 'ऐ', 'ऑ', 'ऒ', 'ओ', 'औ',
'क', 'ख', 'ग', 'घ', 'ङ', 'च', 'छ', 'ज', 'झ', 'ञ', 'ट', 'ठ', 'ड', 'ढ', 'ण', 'त', 'थ',
'द', 'ध', 'न', 'ऩ', 'प', 'फ', 'ब', 'भ']
```

Q.print dir(dict) which is not start with '_'

In [129…
```
print(*[ f for f in dir(dict) if f[0] != '_' ],
      sep='\n')
```

```
clear
copy
fromkeys
get
items
keys
pop
popitem
setdefault
update
values
```

Comprehension for other types

In [130…
```
d = { v for v in range(5) }
print(type(d))
print(d)
```

```
<class 'set'>
{0, 1, 2, 3, 4}
```

In [ ]:

In [131…
```
hindi = ['अ', 'आ', 'इ', 'ई', 'उ', 'ऊ']
english = [ 'a', 'aa', 'e', 'ee', 'u', 'uu']
```

In [132…
```
d = { e:h for e, h in zip(english, hindi) }
print(d)
```

```
{'a': 'अ', 'aa': 'आ', 'e': 'इ', 'ee': 'इ', 'u': 'उ', 'uu': 'ऊ'}
```

In [133…
```
d['aa']
```

Out[133…
```
'आ'
```

In [ ]:

In [134…
```
d = { i:i**2 for i in range(5) }
print(d)
```

```
{0: 0, 1: 1, 2: 4, 3: 9, 4: 16}
```

11. Problems

Question

In [83]:
```
users =    [ 'ram', 'shyam', 'ghyanshyam', 'hari', 'mohan']
password = [ '123', '456',    '654445',     'irah', 'nahom']
accounts = [ 1001,  1002,      1003,        1004,    1005]
balance =  [ 50000, 60000,     10000,       25000,   90000]

acc = int(input("Account Number: "))
amount = int(input("amount: "))

i = accounts.index(acc)
bal = balance[i]

if amount <= bal:
    print(f"{amount} is debited from account")
    bal = bal - amount
    print(f"Remaining Amount is {bal}.")
    balance[i] = bal
else:
    print("Insufficient Balance")
```

```
Account Number: 1002
amount: 20000
20000 is debited from account
Remaining Amount is 40000.
```

In [84]:
```
print(balance)
```

```
[50000, 40000, 10000, 25000, 90000]
```

Question

In [101…

```
data = []

data.append(5)
data.insert(0, 'hello')
data.extend('123')
data.pop(2)
data.append('world')
data.insert(3, 'python')
data.append('python')
data.pop(0)
data.remove('python')
data.reverse()
print(data)
```

```
['python', 'world', '3', '2', 5]
```

Question

Problem Statement

```
you are given n rounds of scores of two teams

Write a program to print name of winner team

if both teams score same print Draw
```

Input Format

```
first line contains an integer n which denotes no of rounds
second linecontains 2*n space seprated integers such that
first n integers are performance of team 1 in n rounds
last  n integers are performance of team 2 in n rounds
```

Output Format

```
print "TEAM 1" if team 1 wins the game
other wise print "TEAM 2" if team 2 wins the game
else print "DRAW"
```

constraints

```
2 <= n <= 10
0 <= arr[i] <= 100
```

Test Case 1:

```
Input

    3
```

1 2 3 3 2 1

Output

DRAW

Test Case 2:

Input
5
3 5 3 2 6 7 4 3 2 1

Output

TEAM 1

map(function, iterable)

In [3]:

```
n = int(input())
scores = list(map(int,input().split()))

t1 = sum(scores[:n])
t2 = sum(scores[n:])

if t1 == t2:
    print("DRAW")
elif t1 > t2:
    print("Team 1")
else:
    print("Team 2")
```

```
5
1 4 5 6 1 3 2 8 9 3
Team 2
```