data structures in python

```
#Creating list with same data type
a = [1,2,3,4,5,6,7,8,9,10]
print(a)
```

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

```
#creating list with differat data type
b = [1,2.5,'manju']
print(b)
```

```
[1, 2.5, 'manju']
```

list operations

"Accessing Items"

```
print(a[8])
print(b[-1])
```

```
10
manju
```

modifying **Items**

```
a[3]=12
print(a)
```

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

```
b[2] = 'manju'
print(b)
```

```
[1, 2.5, 'manju']
```

adding items

```
#append
a.append(10)
a
```

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

```
#insert
b.insert(6,'manju')
b
```

```
[1, 2.5, 'manju', 'manju', 'manju', 'manju']
```

Removing items

```
#remove
a.remove(10)
a
```

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

```
#pop
b.pop(2)
b
```

```
[1, 2.5, 'manju', 'manju']
```

Other operations

```
#len
len(a)
```

```
9
```

```
a.sort()
a.reverse()
a
```

```
[77, 77, 66, 6, 5, 3, 3, 2, 1]
```

Iterating through a list

```
a = (1,3,5,76,56,5,9,4)
for i in a:
  print(i)
```

```
1
3
5
76
56
5
9
4
```

Tuples

```
a = ( 1,2,3,4,'hello',5.5)
print(a[0])
print(a[-1])
```

```
1
5.5
```

```
#accessing items in a tuple
print(a[3])
```

```
4
```

Dictionary

```
student ={
    "name": "manju",
    "age": 18,
    "grade": "A",
    "city": "new york"
}
print(student)
```

```
{'name': 'manju', 'age': 18, 'grade': 'A', 'city': 'new york'}
```

Accessing and modifying Items

```
#Accessing
print(student["age"])
```

```
18
```

```
#Modifying
student["age"] = 21
print(student)
```

```
{'name': 'manju', 'age': 21, 'grade': 'A', 'city': 'new york'}
```

```
#adding
student["gender"] = "male"
print(student)
```

```
{'name': 'manju', 'age': 21, 'grade': 'A', 'city': 'new york',
'gender': 'male'}
```

```
#removing
del student["grade"]
print(student)
```

```
{'name': 'manju', 'age': 21, 'city': 'new york', 'gender': 'male'}
```

Iterating Thought a Dictionary

```
for key, value in student.items():
  print(key,value)
```

```
name manju
age 21
city new york
gender male
```

Set

Creating a set

```
#creating a set
num = {1,2,3,4,5,6,7,8,9}
print(num)
```

```
{1, 2, 3, 4, 5, 6, 7, 8, 9}
```

Set Operations

```
#Adding Items
num.add(10)
print(num)
```

```
{1, 2, 3, 4, 5, 6, 7, 8, 9, 10}
```

```
#Removing Items
num.remove(8)
print(num)
```

```
{1, 2, 3, 4, 5, 6, 7, 9, 10}
```

```
#set operations
#union
a = {1,2,3,4,5}
b = {4,5,6,7,8,}
a | b
```

```
{1, 2, 3, 4, 5, 6, 7, 8}
```

```
#intersection
a & b
```

```
{4, 5}
```

```
#Difference
a - b
```

```
{1, 2, 3}
```

Hands on practice

```
#manipulating Lists
fruits = ['apple','banana','orange','cherry']
fruits.append('mango')
fruits.remove('banana')
print(fruits)
```

```
['apple', 'orange', 'cherry', 'mango']
```

```
#Creating a Dictionary
book = {
    "title": "Python Basics",
    "author": "John Doe",
    "year": 2023

}
print(book["title"])
book["year"]=2024
print(book)
```

```
Python Basics
{'title': 'Python Basics', 'author': 'John Doe', 'year': 2024}
```

```
#Working with Sets
set1 = {1,2,3,4}
set2 = {3,4,5,6}
print("Union:",set1 | set2)
print("Intersection:",set1 & set2)
print("Difference:",set1 - set2)
```

```
Union: {1, 2, 3, 4, 5, 6}
Intersection: {3, 4}
Difference: {1, 2}
```

problems **solving**

```
#Merage Two Lists
list1 = [1,2,3]
list2 = [4,5,6]
merged_list = list1 + list2
print("Merged List:",merged_list)
```

```
Merged List: [1, 2, 3, 4, 5, 6]
```

```
#Dictionary Operations
student = {"name": "john","age":21, "marks":85}
print("Name:",student["name"])
student["marks"] = 90
print("Updated Marks:",student["marks"])
```

```
Name: john
Updated Marks: 90
```

```
#find the maximum and minimum in a list
num = [10,20,30,40,50,]
print("Maximum:",max(num))
print("Minimum:",min(num))
```

```
Maximum: 50
Minimum: 10
```

```
#Count Frequency of Elements

numbers = [1,2,2,3,3,3,4,4,4,4]
frequency = {}
```

find wheather it is **palindrome**

```
number = int(input("Enter a number:"))
reverse_number = 0
temp = number

while temp > 0:
  digit = temp % 10
  reverse_number = reverse_number * 10 + digit
  temp //= 10

if number == reverse_number:
  print(f"{number} is a palindrome")
else:
  print(f"{number} is not a palindrome")
```

```
Enter a number:212
212 is a polindrome
```

```
#palindrome
number = input("Enter a number: ")
if number == number[::-1]:
  print(f"{number} is a  palindrone")
else:
  print(f"{number} is not a palindrome")
```

```
Enter a number: nayan
nayan is a  polindrone
```

```
class Solution(object):
    def isPalindrome(self, x):
        """
        :type x: int
        :rtype: bool
        """
        if x < 0 or (x % 10 == 0 and x != 0):
            return False

        reversed_half = 0
        while x > reversed_half:

            reversed_half = reversed_half * 10 + x % 10
            x //= 10


        return x == reversed_half or x == reversed_half // 10


solution = Solution()
print(solution.isPalindrome(121))
print(solution.isPalindrome(-121))
print(solution.isPalindrome(10))
print(solution.isPalindrome(0))
```

```
True
False
False
True
```