DATA STRUCTURES IN PYTHON

```
numbers = [1,2,3,4,5]
mixed = [11,"hello",3.14]

print(numbers[0])
print(mixed[-1])
```

```
1
3.14
```

```
numbers[0] = 10
print(numbers)
```

```
[10, 2, 3, 4, 5]
```

MODIFYING ITEMS

```
#append
numbers.append(20)
print(numbers)
```

```
[10, 2, 3, 4, 5, 6, 2, 20]
```

```
#insert
numbers.insert(0,50)
print(numbers)
```

```
[50, 10, 10, 2, 3, 4, 5, 6, 2, 20]
```

```
#removing items
numbers.remove(4)
print(numbers)
```

```
[50, 10, 10, 2, 3, 5, 6, 2, 20]
```

```
#pop
numbers.pop()
print(numbers)
```

```
[50, 10, 10, 2, 3, 5, 6, 2]
```

```
#other operations
len(numbers)
```

```
8
```

```
sorted(numbers)
```

```
[2, 2, 3, 5, 6, 10, 10, 50]
```

```
numbers.reverse()
print(numbers)
```

```
[2, 6, 5, 3, 2, 10, 10, 50]
```

```
#tuple
#creating a tuple
coordinates=(10,20,30)
print(coordinates)
```

```
(10, 20, 30)
```

```
#dictionary
student={
    "name":"nida",
    "age":20,
    "marks":90
}

#accessing
print(student["name"])
```

```
nida
```

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

```
21
```

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

```
{'name': 'nida', 'age': 21, 'mmarks': 99}
```

```
# set
numbers = {1,2,3,4,5}
print(numbers)
```

```
{1, 2, 3, 4, 5}
```

```
#set operations
#adding items
numbers.add(5)
print(numbers)
```

```
{1, 2, 3, 4, 5}
```

```
#removing items
numbers.remove(4)
print(numbers)
```

```
{1, 2, 3, 5}
```

MANIPULATING LISTS

```
fruits=["apple","banana","cherry"]
fruits.append("oranges")
fruits.remove("banana")
fruits.insert(0,"kiwi")
fruits.pop()
print(fruits)
```

```
['kiwi', 'apple', 'cherry']
```

CREATING A DICTIONARY

```
book = {
    "title":"python basics",
    "author": "jhon doe",
    "year":2021
}
print(book["title"])
book["year"]=2022
print(book)
```

```
python basics
{'title': 'python basics', 'author': 'jhon doe', 'year': 2022}
```

WORKING WITH SETS

```
set1 = {1,2,3,4,5}
set2 = {4,5,6,7,8}
print("union:",set1 | set2)
print("intersection:",set1 & set2)
print("difference:",set1 - set2)
```

```
union: {1, 2, 3, 4, 5, 6, 7, 8}
intersection: {4, 5}
difference: {1, 2, 3}
```

```
#ques1
#merge two list
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]
```

```
#maximum and minimum
numbers = [10,20,30,40,50]
print("maximum:",max(numbers))
print("minimum:",min(numbers))

maximum: 50
minimum: 10

#frequency list
numbers = [1,2,2,3,3,4,4,4,4]
frequency = {}
for numbers in numbers:
  frequency[numbers] = frequency.get(numbers,0)+1
  print("frequency of elements:",frequency)

frequency of elements: {1: 1}
frequency of elements: {1: 1, 2: 1}
frequency of elements: {1: 1, 2: 2}
frequency of elements: {1: 1, 2: 2, 3: 1}
frequency of elements: {1: 1, 2: 2, 3: 2}
frequency of elements: {1: 1, 2: 2, 3: 2, 4: 1}
frequency of elements: {1: 1, 2: 2, 3: 2, 4: 2}
frequency of elements: {1: 1, 2: 2, 3: 2, 4: 3}
frequency of elements: {1: 1, 2: 2, 3: 2, 4: 4}
```

PALIDROMIC NUMBER

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

while temp > 0:
  digit = temp % 10
  reverse_number = reverse_number * 10 + digit
  temp = temp //10
if number == reverse_number:
    print(f" (number) is a palidromic number")
else:
    print(f" (number) is not a palidromic number")

enter a number202
 (number) is a palidromic number


def is_palindromic(number):
    num_str = str(number)
    return num_str == num_str[::-1]
start = int(input("Enter the start of the range: "))
end = int(input("Enter the end of the range: "))
print("Palindromic numbers in the range:")
```

```
for num in range(start, end + 1):
    if is_palindromic(num):
        print(num, end=" ")
```

```
Enter the start of the range: 202
Enter the end of the range: 505
Palindromic numbers in the range:
202 212 222 232 242 252 262 272 282 292 303 313 323 333 343 353 363
373 383 393 404 414 424 434 444 454 464 474 484 494 505
```

```
number = input("enter a number:")
if number == number[::-1]:
    print("palindrome")
else:
    print("not palindrome")
```

```
enter a number:222
palindrome
```

```
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
```