```
print(4%3);
# dict1 = {a:[1,2,3], b:(1,2,3)};
# print(dict1);
```

```
1
```

```
print(2<=2);
```

```
True
```

```
print(2>=2)
```

```
True
```

```
print(3>=2)
```

```
True
```

Intro to strings

1. how to define strings
2. strings are immutable
3. strings follow sequence means index start from 0
4. strings has len() function to calculate length of a string

```
#intro to strings
a = "information"
type(a)
#indexing start from 0 in string
#string is immutable
```

```
str
```

```
my_first_string = "algebra"
print(my_first_string + "cde");
```

```
algebracde
```

```
my_first_string = "algebra"
len(my_first_string) # function
```

```
7
```

```
str1 = "principal component analysis"
print(str1);
print(str1[::3]);
print(str1[::2])
print(str1[::-1])
print(str1[4:2:-1])
s = "foobar"
print(s[4:2:-1])
```

```
principal component analysis
pnp mntnys
```

```
picplcmoetaayi
sisylana tnenopmoc lapicnirp
cn
ab
```

CONCATINATION

```
#concatenate strings
a = "Tanishak"
b = "Singhal"
print(a + " " + b);
print(a + "4" + b);
print("abc"*4)
```

```
Tanishak Singhal
Tanishak4Singhal
abcabcabcabc
```

lower() method:

```
a = "TANISHAK";
a.lower() #method -> string class
a.upper() #method -> string class
```

```
'TANISHAK'
```

count() method

```
a = "Tanishak"
a.count(" ")
```

```
0
```

find() method

```
a = "Tanishak"
b = "is"
a.find("m")
```

```
-1
```

replace() method

```
algorithm = "Neural Networks"
algorithm.replace(" ", "-") #creates new string and return it as strings are immutable
```

```
'Neural-Networks'
```

```
first_name = "Tanishak"
last_name = "Singhal"
first_name = f'{first_name} {last_name}'
print(first_name)
```

```
Tanishak Singhal
```

Lists

```
list1 = ["A string", 23, 100, 232, "o", True];
len(list1)
list1[0].lower()
list1[0].find("A")
list1[0].replace(" ", "-")
list1[0] = f"everything matters {list1[0]} {list1[4]}"
print(list1[0]);
```

```
everything matters A string o
```

sum(), max(), min(), sorted() method

```
my_list = [1,2,3,4,5]
list1 = ["A string", 23, 100, 232, "o", True];

sum(my_list)
max(my_list)
min(my_list)
mean = sum(my_list)/len(my_list)
print(mean)
sorted(my_list)
```

```
3.0
[1, 2, 3, 4, 5]
```

append() method

```
my_list = [1,2,3,4,5]
# my_list.append([6,7,8]) #append only one element at a time
my_list.extend([6,7,8])
print(my_list)
my_list.extend([9,10,11])#append multiple elements at a time
print(my_list)
```

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

pop() method

```
my_list = [1,2,3,4,5];
my_list.pop(1) #takes index as a argument
```

```
2
```

remove() method

```
my_list = [1,2,3,4,5];
my_list.remove(4) #takes element of a list as an argument
print(my_list)
```

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

Nested lists

```
my_lists = [[1,2,3], ["b", "a", "d"], [7,8,9]] #list inside a list
print(my_lists[0][0:2]) #slicing in nested list
```

```
[1, 2]
```