

STRING

1

1

STRING

Strings in python are surrounded by either single quotation marks, or double quotation marks.

'hello' is the same as "hello".

You can display a string literal with the `print()` function:

2

2

STRING

```
print("AbDc")
print('HyLap')
a = "Hello"
print(a)

s1 = """Python Programming"""
print(s1)
s2 = '''Python Programming'''
print(s2)
```

3

3

Strings are Arrays

```
► a = "Hello, World!"
  print(a[1])
```

4

4

Looping Through a String

```
for x in "banana":
    print(x)
```

5

5

Check String

```
txt = "The books are free!"
print("free" in txt)
```

6

6

Check String

```
txt = "The books are free!"  
if "free" in txt:  
    print("Yes, 'free' is present.")
```

7

7

Check if NOT

```
txt = "The books are free!"  
if "expensive" not in txt:  
    print("No, 'expensive' is NOT present.")
```

8

8

SLICING

```
b = "Hello, World!"  
print(b[2:5])  
print(b[:5])  
print(b[2:])  
print(b[-5:-2])
```

9

9

Upper/Lower case

```
s = "Hello, World!"  
print(s.upper())  
print(s.lower())
```

10

10

Remove Whitespace

```
a = " Hello, World! "  
print(a.strip()) # returns "Hello, World!"
```

11

11

Replace String

```
a = "Hello, World!"  
print(a.replace("H", "J"))
```

12

12

Split String

```
a = "Hello, World!"
print(a.split(",")) # returns ['Hello', ' World!']
```

13

13

String Format

```
age = 26
txt = "My name is Paul, I am " + str(age)
print(txt)
```

14

14

String Format

```
age = 26
txt = "My name is Paul, I am {} "
print(txt.format(age))
```

15

15

String Format

```
quantity = 2
itemno = 350
myorder = "I want {} pieces of item {}."
print(myorder.format(quantity, itemno))
```

16

16

Escape Characters

Code	Result
\'	Single Quote
\\	Backslash
\n	New Line
\r	Carriage Return
\t	Tab
\b	Backspace
\f	Form Feed
\ooo	Octal value
\xhh	Hex value

17

17

TUPLES

18

18

Python Tuples

#A tuple is a collection which is ordered
#and **unchangeable**.
#Tuples are written with round brackets.

```
thistuple = ("apple", "banana", "cherry")
print(thistuple)
```

```
t = ("apple", "banana", "cherry", "apple", "cherry")
print(t)
print(len(t))
```

19

19

Python Tuples

```
tuple1 = ("apple", "banana", "cherry")
tuple2 = (1, 5, 7, 9, 3)
tuple3 = (True, False, False)
tuple4 = ("abc", 34, True, 40, "male")
print(type(tuple4))
```

20

20

Tuples with one item

```
t = ("apple",)
print(type(t))
```

21

21

The tuple() Constructor

```
yourtuple = tuple(("apple", "banana", "cherry"))
print(yourtuple)
```

22

22

Access Tuple Items

```
thistuple = ("apple", "banana", "cherry", "orange", "kiwi")
print(thistuple[1])
print(thistuple[-1])
print(thistuple[0:1])
print(thistuple[:4])
print(thistuple[2:])
```

```
if "apple" in thistuple:
    print("Yes, 'apple' is in the fruits tuple")
```

23

23

Change Tuple

```
x = ("apple", "banana", "cherry", "orange")
y = list(x)
y[1] = "kiwi"
x = tuple(y)

print(x)
```

24

24

Add Items

```
thistuple = ("apple", "banana", "cherry")
y = list(thistuple)
y.append("orange")
thistuple = tuple(y)

z = ("orange",)
thistuple += z

print(thistuple)
```

25

25

Remove Items

```
thistuple = ("apple", "banana", "cherry")
y = list(thistuple)
y.remove("apple")
thistuple = tuple(y)

del thistuple
```

26

26

Unpacking a Tuple

```
fruits = ("apple", "banana", "cherry")

(green, yellow, red) = fruits

print(green)
print(yellow)
print(red)
```

27

27

Using Asterisk*

```
fruits = ("apple", "banana", "cherry", "strawberry", "raspberry")

(green, yellow, *red) = fruits

print(green)
print(yellow)
print(red)
```

28

28

Using Asterisk*

```
fruits = ("apple", "mango", "papaya", "pineapple", "cherry")

(green, *tropic, red) = fruits

print(green)
print(tropic)
print(red)
```

29

29

Loop

```
thistuple = ("apple", "banana", "cherry")
for x in thistuple:
    print(x)
```

30

30

Loop

```
thistuple = ("apple", "banana", "cherry")
for i in range(len(thistuple)):
    print(thistuple[i])
```

31

31

Loop

```
thistuple = ("apple", "banana", "cherry")
i = 0
while i < len(thistuple):
    print(thistuple[i])
    i = i + 1
```

32

32

Join two tuples

```
tuple1 = ("a", "b", "c")
tuple2 = (1, 2, 3)

tuple3 = tuple1 + tuple2
print(tuple3)
```

33

33

Multiply tuples

```
fruits = ("apple", "banana", "cherry")
mytuple = fruits * 2

print(mytuple)
```

34

34

Tuple Methods

Method	Description
<code>count()</code>	Returns the number of times a specified value occurs in a tuple
<code>index()</code>	Searches the tuple for a specified value and returns the position of where it was found

35

35

SETS

36

36

SETS

- › Sets are used to store multiple items in a single variable.
- › A set is a collection which is *unordered*, *unchangeable*, and *unindexed*.
- › Sets are written with curly brackets.
- ›

```
thisset = {"apple", "banana", "cherry"}  
print(thisset)
```

37

37

Duplicates Not Allowed

```
thisset = {"apple", "banana", "cherry", "apple"}  
print(thisset)
```

38

38

Duplicates Not Allowed

True và 1 is considered the same value

```
thisset = {"apple", "banana", "cherry", True, 1, 2}  
print(thisset)
```

39

39

Duplicates Not Allowed

False và 0 is considered the same value

```
thisset = {"apple", "banana", "cherry", False, True, 0}  
print(thisset)
```

40

40

Length of a SET

Get the number of items in a set

```
thisset = {"apple", "banana", "cherry"}  
print(len(thisset))
```

41

41

SET items – Data types

```
set1 = {"apple", "banana", "cherry"}  
set2 = {1, 5, 7, 9, 3}  
set3 = {True, False, False}  
set4 = {"abc", 34, True, 40, "male"}  
  
print(type(set1))
```

42

42

The set() constructor

```
thisset = set(("apple", "banana", "cherry"))  
print(thisset)
```

43

43

Access Items

```
thisset = {"apple", "banana", "cherry"}  
  
for x in thisset:  
    print(x)  
  
print("banana" in thisset)
```

44

44

Add Items

```
thisset = {"apple", "banana", "cherry"}  
  
thisset.add("orange")  
  
print(thisset)
```

45

45

Add SETs

```
thisset = {"apple", "banana", "cherry"}  
tropical = {"pineapple", "mango"}  
  
thisset.update(tropical)  
  
print(thisset)
```

46

46

Add Any Iterable

```
thisset = {"apple", "banana", "cherry"}  
mylist = ["kiwi", "orange"]  
  
thisset.update(mylist)  
  
print(thisset)
```

47

47

Remove Item

```
thisset = {"apple", "banana", "cherry"}  
  
thisset.remove("banana")  
  
print(thisset)
```

48

48

Remove Item

```
thisset = {"apple", "banana", "cherry"}  
thisset.discard("banana")  
print(thisset)
```

49

49

Remove Item

```
thisset = {"apple", "banana", "cherry"}  
x = thisset.pop()  
print(x)  
print(thisset)
```

50

50

Remove Item

```
thisset = {"apple", "banana", "cherry"}  
thisset.clear()  
print(thisset)
```

51

51

Remove Item

```
thisset = {"apple", "banana", "cherry"}  
del thisset  
print(thisset)
```

52

52

Join Two SETs

```
set1 = {"a", "b", "c"}  
set2 = {1, 2, 3}  
  
set3 = set1.union(set2)  
print(set3)  
  
set1.update(set2)  
print(set1)
```

53

53

Keep Only the Duplicates

```
x = {"apple", "banana", "cherry"}  
y = {"google", "microsoft", "apple"}  
  
x.intersection_update(y)  
  
print(x)
```

54

54

Keep Only the Duplicates

```
x = {"apple", "banana", "cherry"}
y = {"google", "microsoft", "apple"}

z = x.intersection(y)

print(z)
```

55

55

Keep All, But NOT the Duplicates

```
x = {"apple", "banana", "cherry"}
y = {"google", "microsoft", "apple"}

x.symmetric_difference_update(y)
print(x)

z = x.symmetric_difference(y)
print(z)
```

56

56

Keep All, But NOT the Duplicates

```
x = {"apple", "banana", "cherry", True}
y = {"google", 1, "apple", 2}

z = x.symmetric_difference(y)

print(z)
```

57

57

DICTIONARIES

58

58

Dictionary

- Dictionaries are used to store data values in key:value pairs.
- A dictionary is a collection which is changeable, ordered* and do not allow duplicates.

59

59

Dictionary

```
thisdict = {
    "brand": "Ford",
    "model": "Mustang",
    "year": 1964
}
print(thisdict)
```

60

60

Dictionary Items

```
thisdict = {  
    "brand": "Ford",  
    "model": "Mustang",  
    "year": 1964  
}  
print(thisdict["brand"])
```

61

61

Duplicates Not Allowed

```
thisdict = {  
    "brand": "Ford",  
    "model": "Mustang",  
    "year": 1964,  
    "year": 2020  
}  
print(thisdict)
```

62

62

Dictionary Length

```
thisdict = {  
    "brand": "Ford",  
    "model": "Mustang",  
    "year": 2020  
}  
print(len(thisdict))
```

63

63

Dictionary Items – Data Types

```
thisdict = {  
    "brand": "Ford",  
    "electric": False,  
    "year": 1964,  
    "colors": ["red", "white", "blue"]  
}  
  
print(type(thisdict))
```

64

64

The dict() constructor

```
thisdict = dict(name = "John", age = 36, country  
= "Norway")  
  
print(thisdict)
```

65

65

Accessing Items

```
thisdict = {  
    "brand": "Ford",  
    "model": "Mustang",  
    "year": 1964  
}  
x = thisdict["model"]  
x = thisdict.get("model")
```

66

66

Get Keys

```
thisdict = {
    "brand": "Ford",
    "model": "Mustang",
    "year": 1964
}
x = thisdict.keys()
```

67

67

Get Keys

```
car = {
    "brand": "Ford",
    "model": "Mustang",
    "year": 1964
}
x = car.keys()
print(x)
car["color"] = "white"
print(x)
```

68

68

Get Values

```
car = {
    "brand": "Ford",
    "model": "Mustang",
    "year": 1964
}
x = car.values()
print(x)
car["year"] = 2020
print(x)
```

69

69

Get Values

```
car = {
    "brand": "Ford",
    "model": "Mustang",
    "year": 1964
}
x = car.values()
print(x)
car["color"] = "red"
print(x)
```

70

70

Check if Key exists

```
thisdict = {
    "brand": "Ford",
    "model": "Mustang",
    "year": 1964
}

if "model" in thisdict:
    print("'model' là một key trong từ điển")
```

71

71

Change Values – Adding Items

```
thisdict = {
    "brand": "Ford",
    "model": "Mustang",
    "year": 1964
}
thisdict["year"] = 2018
thisdict["color"] = "red"
```

72

72

Change Values – Adding Items

```
thisdict = {
    "brand": "Ford",
    "model": "Mustang",
    "year": 1964
}
thisdict.update({"year": 2020})
thisdict.update({"color": "red"})
```

73

73

Removing Items

```
thisdict = {
    "brand": "Ford",
    "model": "Mustang",
    "year": 1964
}
thisdict.pop("model") # del thisdict["model"]
print(thisdict)
```

74

74

Removing Items

```
thisdict = {
    "brand": "Ford",
    "model": "Mustang",
    "year": 1964
}
thisdict.clear()
print(thisdict)

del thisdict
```

75

75

Loop through a Dictionary

```
thisdict = {
    "brand": "Ford",
    "model": "Mustang",
    "year": 1964
}

#Print all key names in the dictionary, one by one:
for x in thisdict:
    print(x)
```

76

76

Loop through a Dictionary

```
thisdict = {
    "brand": "Ford",
    "model": "Mustang",
    "year": 1964
}

#Print all values in the dictionary, one by one:
for x in thisdict:
    print(thisdict[x])
```

77

77

Loop through a Dictionary

```
thisdict = {
    "brand": "Ford",
    "model": "Mustang",
    "year": 1964
}
for x in thisdict.values():
    print(x)
for x in thisdict.keys():
    print(x)
for x, y in thisdict.items():
    print(x, y)
```

78

78

Copy a Dictionary

```
thisdict = {
    "brand": "Ford",
    "model": "Mustang",
    "year": 1964
}
mydict = thisdict.copy()
print(mydict)
```

79

79

Copy a Dictionary

```
thisdict = {
    "brand": "Ford",
    "model": "Mustang",
    "year": 1964
}
mydict = dict(thisdict)
print(mydict)
```

80

80

Netsed Dictionaries

Create a dictionary that contain three dictionaries:

```
myfamily = {
    "child1" : {
        "name" : "Emil",
        "year" : 2004
    },
    "child2" : {
        "name" : "Tobias",
        "year" : 2007
    },
    "child3" : {
        "name" : "Linus",
        "year" : 2011
    }
}
```

81

81

Netsed Dictionaries

Create three dictionaries, then create one dictionary that will contain the other three dictionaries:

```
child1 = {
    "name" : "Emil",
    "year" : 2004
}
child2 = {
    "name" : "Tobias",
    "year" : 2007
}
child3 = {
    "name" : "Linus",
    "year" : 2011
}
myfamily = {
    "child1" : child1,
    "child2" : child2,
    "child3" : child3
}
```

82

82