

**Program Name** Please execute the program and check the output  
demo1.py

```
s="hello"  
print(s)  
print(type(s))
```

**Output**

**Program Name** Please execute the program and check the output  
demo2.py

```
s="hello"  
print(length(s))
```

**Output**

**Program Name** Please execute the program and check the output  
demo3.py

```
s="hello"  
  
print(len(s))
```

**Output**

**Program Name** Please execute the program and check the output  
demo4.py

```
s1="hello"  
s2="hello"  
print(id(s1))  
print(id(s2))
```

**Output**

**Program Name** Please execute the program and check the output  
demo5.py

```
s1="hello"  
s2="hello"  
print(s1 is s2)
```

**Output**

**Program Name** Please execute the program and check the output  
demo6.py

```
s1="hello"  
s2="hello"  
s3="hi"  
print(s1 is s2)  
print(s2 is s3)
```

**Output**

**Program Name** Please execute the program and check the output  
demo7.py

```
s1="hello"  
s2="hello"  
s3="hi"  
print(s1 == s2)  
print(s2 == s3)
```

**Output**

**Program Name** Please execute the program and check the output  
demo8.py

```
print("a"+"bc")
```

**Output**

**Program Name** Please execute the program and check the output  
demo9.py

```
print("abcd"[2:])
```

**Output**

**Program Name** Please execute the program and check the output  
demo10.py

```
str1 = 'hello'  
str2 = ','  
str3 = 'world'  
  
print(str1[-1:])
```

**Output**

**Program Name** Please execute the program and check the output  
demo11.py

```
print(r"\nhello")
```

**Output**

**Program Name** Please execute the program and check the output  
demo12.py

```
print('new' 'line')
```

**Output**

**Program Name** Please execute the program and check the output  
demo13.py

```
str1="helloworld"
```

```
print(str1[::-1])
```

**Output**

**Program Name** Please execute the program and check the output  
demo14.py

```
example = "snow world"
```

```
example[3] = 's'
```

```
print(example)
```

**Output**

**Program Name** Please execute the program and check the output  
demo15.py

```
l = [1, 2, 3]
```

```
print(min(l))
```

```
print(max(l))
```

**Output**

**Program Name** Please execute the program and check the output  
demo16.py

```
q="what are you"  
print(min(q))  
print(max(q))
```

**Output**

**Program Name** Please execute the program and check the output  
demo17.py

```
example="hello"  
print(example.count("l"))
```

**Output**

**Program Name** Please execute the program and check the output  
demo18.py

```
example = "helle"  
print(example.find("e"))
```

**Output**

**Program Name** Please execute the program and check the output  
demo19.py

```
example = "helle"  
print(example.rfind("e"))
```

**Output**

**Program Name** Please execute the program and check the output  
demo20.py

```
example="helloworld"  
print(example[::-1])
```

**Output**

**Program Name** Please execute the program and check the output  
demo21.py

```
str1="restart"  
  
char = str1[0]  
str1 = str1.replace(char, '$')  
print(str1)
```

**Output**

**Program Name** Please execute the program and check the output  
demo22.py

```
str1="restart"
```

```
char = str1[0]  
str1 = str1.replace(char, '$')  
str2 = char + str1[1:]
```

```
print(str2)
```

**Output**

**Program Name** Please execute the program and check the output  
demo23.py

```
example="helloworld"  
print(example[::-1].startswith("d"))
```

**Output**

**Program Name** Please execute the program and check the output  
demo24.py

```
print("hello\\example\\test.txt")
```

**Output**



**Program Name** Please execute the program and check the output  
demo25.py  
`print("hello\\example\\test.txt")`  
**Output**

**Program Name** Please execute the program and check the output  
demo26.py  
`print("hello\"example\"test.txt")`  
**Output**

**Program Name** Please execute the program and check the output  
demo27.py  
`print("hello"\example"\test.txt")`  
**Output**

**Program Name** Please execute the program and check the output  
demo28.py  
`print("hello"+1+2+3)`  
**Output**

**Program Name** Please execute the program and check the output  
demo29.py

```
print("D", end = ' ')\nprint("C", end = ' ')\nprint("B", end = ' ')\nprint("A", end = ' ')
```

**Output**

**Program Name** Please execute the program and check the output  
demo30.py

```
print("Hello".replace("l", "e"))
```

**Output**

**Program Name** Please execute the program and check the output  
demo31.py

```
print("abc DEF".capitalize())
```

**Output**

**Program Name** Please execute the program and check the output  
demo32.py

```
print("abcdef".upper())
```

**Output**

**Program Name** Please execute the program and check the output  
demo33.py  
`print("ABCDEF".upper())`  
**Output**

**Program Name** Please execute the program and check the output  
demo34.py  
`print("ABCDEFGF".lower())`  
**Output**

**Program Name** Please execute the program and check the output  
demo35.py  
`print("abcdef".center())`  
**Output**

**Program Name** Please execute the program and check the output  
demo36.py  
`print("xyyzxyzxyy".count('yy'))`  
**Output**

**Program Name** Please execute the program and check the output  
demo37.py

```
print("xyyzxyzxyy".count('yy', 1))
```

**Output**

**Program Name** Please execute the program and check the output  
demo38.py

```
print("xyyzxyzxyy".count('yy', 4))
```

**Output**

**Program Name** Please execute the program and check the output  
demo39.py

```
print("xyyzxyzxyy".endswith("xyy"))
```

**Output**

**Program Name** Please execute the program and check the output  
demo40.py

```
print("ab\tcd\tef")
```

**Output**

<b>Program Name</b>	Please execute the program and check the output demo41.py  <code>print("a\nb")</code>
<b>Output</b>	

<b>Program Name</b>	Please execute the program and check the output demo42.py  <code>print("abcdef".find("cd"))</code>
<b>Output</b>	

<b>Program Name</b>	Please execute the program and check the output demo43.py  <code>print("ccdcdcd".find("c"))</code>
<b>Output</b>	

**Program Name** Please execute the program and check the output  
demo44.py

```
print("Hello {0} and {1}".format('foo', 'bin'))
```

**Output**

**Program Name** Please execute the program and check the output  
demo45.py

```
print("Hello {1} and {0}".format('bin', 'foo'))
```

**Output**

**Program Name** Please execute the program and check the output  
demo46.py

```
print("Hello {} and {}".format('foo', 'bin'))
```

**Output**

**Program Name** Please execute the program and check the output  
demo47.py

```
print("Hello {name1} and {name2}".format('foo', 'bin'))
```

**Output**

**Program Name** Please execute the program and check the output  
demo48.py

```
print("Hello {name1} and  
{name2}".format(name1='foo',name2='bin'))
```

**Output**

**Program Name** Please execute the program and check the output  
demo49.py

```
print('The sum of {0} and {1} is {2}'.format(2, 10, 12))
```

**Output**

**Program Name** Please execute the program and check the output  
demo50.py

```
print('ab12'.isalnum())
```

**Output**

**Program Name** Please execute the program and check the output  
demo51.py

```
print('ab,12'.isalnum())
```

**Output**

**Program Name** Please execute the program and check the output  
demo52.py

```
print('ab'.isalpha())
```

**Output**

**Program Name** Please execute the program and check the output  
demo53.py

```
print('a B'.isalpha())
```

**Output**



**Program Name** Please execute the program and check the output  
demo54.py  
`print('0xa'.isdigit())`  
**Output**

**Program Name** Please execute the program and check the output  
demo55.py  
`print('').isdigit())`  
**Output**

**Program Name** Please execute the program and check the output  
demo56.py  
`print('my_string'.isidentifier())`  
**Output**

**Program Name** Please execute the program and check the output  
demo57.py

```
print('$my_string'.isidentifier())
```

**Output**

**Program Name** Please execute the program and check the output  
demo58.py

```
print('abc'.islower())
```

**Output**

**Program Name** Please execute the program and check the output  
demo59.py

```
print('a@ 1'.islower())
```

**Output**

**Program Name** Please execute the program and check the output  
demo60.py

```
print('11'.isnumeric())
```

**Output**

**Program Name** Please execute the program and check the output  
demo61.py

```
print('HelloWorld'.istitle())
```

**Output**

**Program Name** Please execute the program and check the output  
demo62.py

```
print('Hello World'.istitle())
```

**Output**

**Program Name** Please execute the program and check the output  
demo63.py

```
s1="" hello
    hi
    how are you""
print(s1)
```

**Output**

**Program Name** Please execute the program and check the output  
demo64.py

```
s1="" hello
    hi
how are you""
print(s1.strip())
```

**Output**

**Program Name** Please execute the program and check the output  
demo65.py

```
print('abcdef'.partition('cd'))
```

**Output**

**Program Name** Please execute the program and check the output  
demo66.py

```
print('abcdefcdgh'.partition('cd'))
```

**Output**

**Program Name** Please execute the program and check the output  
demo67.py

```
print('abcd'.partition('cd'))
```

**Output**

**Program Name** Please execute the program and check the output  
demo68.py

```
print('abcdef12'.replace('cd', '12'))
```

**Output**

**Program Name** Please execute the program and check the output  
demo69.py

```
print('abef'.replace('cd', '12'))
```

**Output**

**Program Name** Please execute the program and check the output  
demo70.py

```
print('abcdefcdghcd'.split('cd'))
```

**Output**

**Program Name** Please execute the program and check the output  
demo71.py  
`print('Ab!2'.swapcase())`  
**Output**

**Program Name** Please execute the program and check the output  
demo72.py  
`print('ab cd ef'.title())`  
**Output**

**Program Name** Please execute the program and check the output  
demo73.py  
`print('ab cd-ef'.title())`  
**Output**

### 1. Accessing string characters

- ✓ We can access string characters by using,
  - Indexing
  - Slicing

#### 1.1. Indexing in string

- ✓ Indexing means a position of string's characters where it stores.
- ✓ We need to use square brackets `[]` to access the string index.
- ✓ String indexing result is string type.

#### Python support two types of indexes

- ✓ Positive index
- ✓ Negative index

#### 1.2. Positive index

- ✓ The position of string characters can be positive index from **left to right** direction (we can say forward direction).
- ✓ In this way, the starting position is **0**(zero)

#### 1.3. Negative index

- ✓ The position of string characters can be negative index from **right to left** direction (we can say backward direction).
- ✓ In this way, the starting position is **-1**(minus one)
  - Please don't think too much like **-0** (minus zero), there is no minus zero boss.

### Internal representation

**Program Name**     Printing text message  
demo74.py

```
wish = "Hello World"  
print(wish)
```

**output**  
Hello World

### Diagram representation

-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1
H	e	l	l	o		W	o	r	l	d
0	1	2	3	4	5	6	7	8	9	10

**Program Name**     Accessing string index by using integer values.  
demo75.py

```
wish = "Hello World"  
  
print(wish[0])  
print(wish[1])
```

**output**  
H  
e



**Program Name**      Accessing string index by using negative values.  
demo76.py

```
wish = "Hello World"
```

```
print(wish[-1])  
print(wish[-2])
```

**output**

```
d  
l
```

### 1.4. Slicing in string

- ✓ A substring of a string is called as a slice.
- ✓ A slice can represent a part of string from string or a piece of string.
- ✓ String slicing result is string type.
- ✓ We need to use square brackets `[]` in slicing.

**Syntax 1:**

```
nameofthestring [start: stop]
```

- ✓ **start**
  - It indicates the index where slice can start.
  - Default value is 0
- ✓ **stop**
  - It indicates the index where slice can end.
  - Default value is max allowed index of list i.e. length of the string

### Syntax2:

nameofthestring [start : stop :step]

- ✓ **start**
  - It indicates the index where slice can start.
  - Default value is 0
- ✓ **stop**
  - It indicates the index where slice can end.
  - Default value is max allowed index of list i.e. length of the string
- ✓ **Step size**
  - Increment value.
  - Default value is 1

### 2. String several cases in slicing

#### Make a note

- ✓ If we are not specifying begin index, then it will consider from beginning of the string.
- ✓ If we are not specifying end index, then it will consider up to end of the string.
- ✓ The default value for step is 1

```
wish = "Hello World"
```

- ✓ `wish[:]`      => accessing from 0<sup>th</sup> to last
- ✓ `wish[::]`      => accessing from 0<sup>th</sup> to last
- ✓ `wish[2::]`      => accessing from 2<sup>nd</sup> position to till last.
- ✓ `wish[4::]`      => accessing from 4<sup>th</sup> position to till last.
- ✓ `wish[:8:]`      => accessing from 0<sup>th</sup> to 7<sup>th</sup> in steps of 1
- ✓ `wish[0:9:1]`   => accessing string from 0<sup>th</sup> to 8<sup>th</sup> means (9-1) element.

**Program**      slice operator several use cases  
**Name**          demo77.py

```
wish = "Hello World"
```

```
print(wish[:])  
print(wish[::])  
print(wish[2::])  
print(wish[4::])  
print(wish[:8:])  
print(wish[0:9:1])
```

**Output**

```
Hello World  
Hello World  
llo World  
o World  
Hello Wo  
Hello Wor
```