String Examples

Input: Read string from user (keyboard)

```
string = input('Enter a string:')
```

Empty string: We can use ' ' to refer to an empty string. It is equivalent of the number 0.

```
string =' '
sting = " "
```

Length: To get the length of a string (how many characters), we can use the built-in function len

```
string ='CSE4IP'
print (len(string))
>>> 6
print (len("ABC "))
>>> 4 # space is counted as one character
```

Concatenation: We can use + operator combines several strings

```
s1 ='CSE4IP'
s2 =' Sem 1'
s3=' PG'
print (s1+s2)
>>> CSE4IP Sem 1
print (s1+s2+s3)
>>> CSE4IP Sem 1 PG
```

Repetition: We can use * operator combines to repeat strings several times

```
s1 ='CSE4IP '
print (s1 * 3)
>>> CSE4IP CSE4IP CSE4IP
print('-' * 10)
>>> ----------
```

Examples

in operator: The **in** operator is used to check if an item is a member of a given string.

```
string ='CSE4IP'
if 'P' in string:
    print('The string contains the letter P.')
```

not operator: We use both **not** and **in** operators to check if an item is **Not** a member of a given string.

```
s='Z' # s=input ("enter a letter: ")
string ='CSE4IP'
if s not in string:
    print('The string does not contains {} letter'
        .format(s))
```

We can re-write previous example (CSE4IP character combination) using **in** operator instead of using **Long** **or** and **if** conditional statement.

```
string=' '
for i in range (6):
    s = input('Enter a letter: ')
    if s in 'CSE4IP':
        string = string + s
print(string)
```

Indexing: We can use the square brackets **[]** to access to a string letter (character). In python, string starts with index 0. We can also use negative indexing to access to the last character. For example, CSE4IP letter indices are as follows:

index:	0	1	2	3	4	5	6
letters:	C	S	E	5	A	P	G

```
string = "Welcome to CSE4IP"
print (string[0])
>>> W
print (string[1])
>>> e
print (string[-1])
>>> P
print (string[6])
>>>
print (string[18])
>>> IndexError: string index out of range
```

Examples

A slicing operator can be used to access to a set of string letters. It acts like a combination of indexing and the **range()** function.

Slicing: a slicing operator : (colon) can be used to get a range of characters: **string_name** [starting location : ending location+1]. For example, CSE4IP letter indices are as follows:

index:	0	1	2	3	4	5
letters:	C	S	E	4	I	P

```
string = "Welcome to CSE4IP"
print (string[0:3])
>>> Wel
print (string[6:8])
>>> e
print (string[6:9])
>>> e t
print (string[0:17])
>>> Welcome to CSE4IP
print (string[0:18])
>>> Welcome to CSE4IP
print (string[0:24])
>>> Welcome to CSE4IP
print (string[13:])
>>> E4IP
print (string[-2:])
>>> IP
print (string[1:7:2]) #  from 1 to 6, by twos
>>> ecm
```

Examples

- Can we change a character in a string?
- String is immutable so we can not change the character of the created string.
- We can simply reassign different string to the same name.

Change string: Change the letter at index 1 into 'A': "Welcome to CSE4IP".

```
string = "Welcome to CSE4IP"
string[0]='A'
>>> TypeError: 'str' object does not support
        item assignment
```

Change character of string: If we want to change a character of string, we need to write a tricky method. We have to instead create a new string and reassign it to old string using slicing or loop function. For example, we could replace 'A' and change it with index 1 using slice operator.

```
string = "Welcome to CSE4IP"
s='A'+string[1:]
print (s)
>>> Aelcome to CSE4IP
s='A'+string[2:]
print (s)
>>> Alcome to CSE4IP
s2=string[0:6]+' x' + string[7:10]+ 'z'+string[10:]
print (s2)
>>> Welcome x toz CSE4IP
```

Examples

- Can we delete a character in a string?
- String is immutable so we can not delete the character of the created string.
- We can use `del()` to delete the entire string.

Delete character: We can not delete a character from a created string.

```
>>> string = "Welcome to CSE4IP"
>>> print (string)
Welcome to CSE4IP
>>> del string[0]
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: 'str' object doesn't support item deletion
```

Delete string: We can delete the entire string using `del()` function.

```
>>> string = "Welcome to CSE4IP"
>>> print (string)
Welcome to CSE4IP
>>> del string
>>> print (string)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: name 'string' is not defined
```

Examples

We can loop through a sting using for or while statements. We can also use in function.

for-loop: We can loop through a sting using for.

```
>>> string = "CSE4IP"
>>> for i in range(len(string)):
...     print (string[i],'! ',end='')
...
C ! S ! E ! 4 ! I ! P !
```

while-loop: We can loop through a sting using while statement.

```
>>> string = "CSE4IP"
>>> n=len(string)
>>> i=0
>>> while i<n:
...     print (string[i],'* ',end='')
...     i+=1
...
C * S * E * 4 * I * P *
```

in: We can loop through a sting using for statement and in.

```
>>> string = "CSE4IP Sem 1"
>>> for i in string:
...     print (i,' ', end='')
...
C  S  E  4  I  P     S  e  m     1
```

Examples

Python provides several methods for string that either return information about the current string or return a new string by modifying the current string. Here are some of the useful ones.

Lower: `lower()` returns a new string by changing the current one letters into lowercase.

```
>>> string = "CSE4IP SEM 1"
>>> s=string.lower()
>>> print (s)
cse4ip sem 1
```

Upper: `upper()` returns a new string by changing the current one letters into uppercase.

```
>>> string = "cse4ip Sem 2"
>>> s=string.upper()
>>> print (s)
CSE4IP SEM 2
```

Count: `count(x)` counts the number of occurrences of x in a given string

```
>>> string = "WELCOME to CSE4IP"
>>> s=string.count('E')
>>> print ("E counted {} time".format(s))
E counted 3 time
```

Examples

Replace: `replace(x,y)` returns a string with every occurrence of x replaced by y

```
>>> string = "WELCOME to CSE4IP"
>>> s=string.replace('E','X')
>>> print (s)
WXLCOMX to CSX4IP
```

Index: `index(x)` returns the location of the first occurrence of x

```
>>> string = "WELCOME to CSE4IP"
>>> s=string.index('L')
>>> print (s)
2
```

isalpha: `isalpha()` returns `True` if every character of the string is a letter.

```
>>> string = "WELCOME to CSE4IP"
>>> s=string.isalpha()
>>> print (s)
False

>>> string = "WELCOME"
>>> s=string.isalpha()
>>> print (s)
True
```

Join: the `join()` function takes all items of an iterable and joins them into one string

```
>>> string = "CSE4IP"
>>> s=','.join(string) # join by comma
>>> print (s)
C,S,E,4,I,P
```

Examples

strip: the strip() function removes spaces at the beginning and at the end of the string

```
>>> string = " CSE4IP "
>>> s=string.strip()
>>> print (s)
CSE4IP
```

Split: the split() function splits the string into list of strings or letters

```
>>> string = "C S E 4 I P"
>>> s=string.split( )
>>> print (s)
['C', 'S', 'E', '4', 'I', 'P']
>>> string = "WELCOME to CSE4IP"
>>> print (string.split( ))
['WELCOME', 'to', 'CSE4IP']
>>> print (string.split('to'))
['WELCOME ', ' CSE4IP']
>>> string = "WELCOME, to CSE4IP"
>>> print (string.split(','))
['WELCOME', ' to CSE4IP']
```

Find: the find() returns the index of first occurrence of the specified letter

```
>>> string = "WELCOME to CSE4IP"
>>> f=string.find('P')
>>> print (f)
16
>>> print (string.find('o'))
9
>>> print (string.find('T'))
-1
```

Examples

Example: Write a Python program that asks the user for a string and then display the location of each 'b' in the provided string.

```
string = input('Enter strings: ')
for i in range(len(string)):
    if string[i]=='b':
        print(i)
```

Example: Write a python program that asks the user for a string and then creates a new string which doubles each character of the provided string. For example, if the string is Hi, the output should be HHii.

```
>>> # string = input('Enter strings: ')
>>> string= 'CSE4IP'
>>> new_s =' '
>>> for i in string:
...     new_s = new_s + i*2
...
>>> print(new_s)
 CCSSEE44IIPP
```

Example: Write a Python program that takes string which contains a decimal number and then print out the decimal part only. For example, if we give 4.21711, the program should print out .21711.

```
>>> #string = input('Enter your decimal number: ')
>>> string="4.21711"
>>> s1=string[string.index('.'): ]
>>> print(s1)
.21711
>>> s2=    string.find('.')
>>> print (string[s2:])
.21711
```

Examples

Example: Ask the user to enter several numbers on one line separated by space. Split the line up into tokens (sequences of characters separated by white-space characters). Print the total of all numbers.

```
>>> #input("Enter numbers on one line:")
>>> line = "1 2 3 4 5"
>>> print("line:", line)
line: 1 2 3 4 5
>>> tokens = line.split()
>>> print("tokens:", tokens)
tokens: ['1', '2', '3', '4', '5']
>>> # Add the numbers up (after convert each
>>> # token into a number
>>> total = 0
>>> for s in tokens:
...     total = total + float(s)
...
>>> print("total:", total)
total: 15.0
```