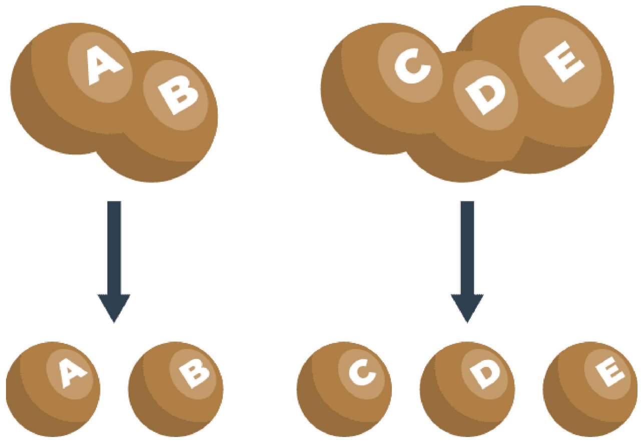
Function

- Repeated many times in your program.
- if a particular fragment of the code begins to appear in more than one place, consider the possibility of isolating it in the form of a function

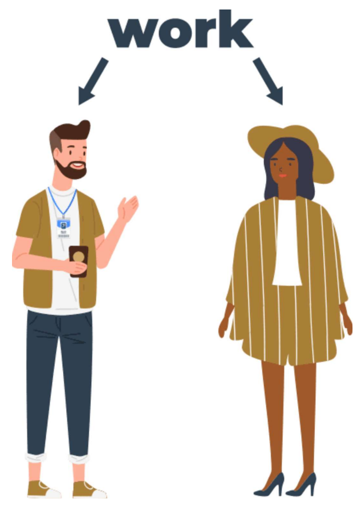


```
def function_name():
   function_body
```

```
print("Enter a value: ")
a = int(input())
print("Enter a value: ")
b = int(input())
print("Enter a value: ")
c = int(input())
```

Function

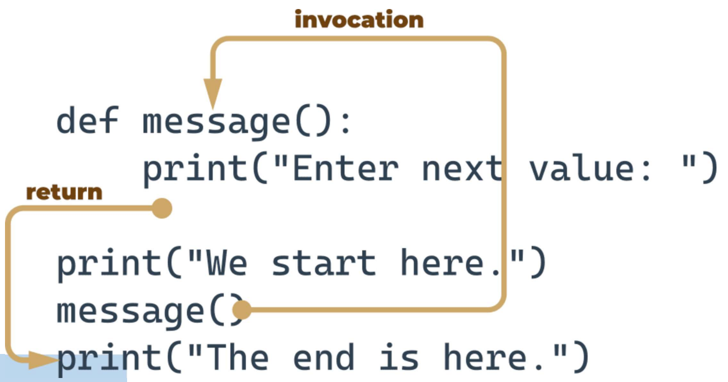
```python
def message():
    print("Enter value")
    c=input()
    print(c)
message()
```



```
= RESTART: C:/23CYBER/PF_LAB/Lab 9/Prac.py
Enter value
44
44
```

Function-How functions work

- when you invoke a function, Python remembers the place where it happened and jumps into the invoked function;
- the body of the function is then executed;
- reaching the end of the function forces Python to return to the place directly after the point of invocation.
- You mustn't invoke a function which is not known at the moment of invocation.



```
def message():
    print("Enter next value: ")

print("We start here.")
message()
print("The end is here.")
```

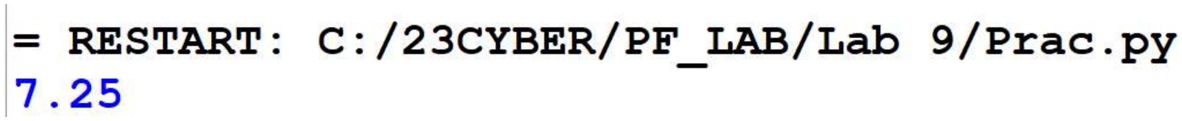
```
def hello(name): # defining a function
   print("Hello,", name) # body of the function

name = input("Enter your name: ")
hello(name) # calling the function
```

Function-Build in Function

- **abs()** Returns the absolute value of a number

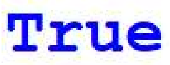
```
x=abs(-7.25)
print(x)
```



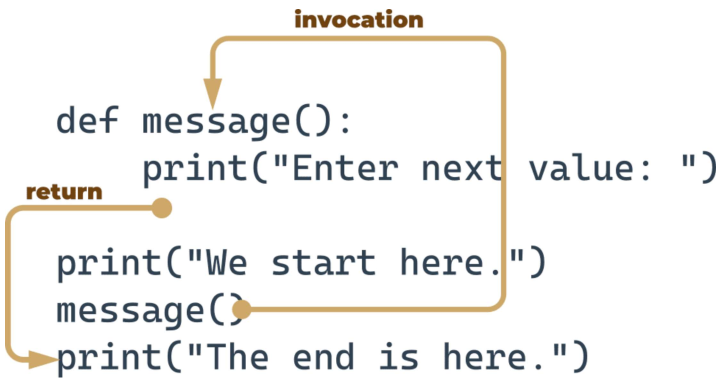
```
= RESTART: C:/23CYBER/PF_LAB/Lab 9/Prac.py
7.25
```

- **all()** Returns True if all items in an iterable object are true

```
mylist = [True, True, True]
x = all(mylist)
```



```
True
```



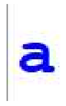
```
def message():
    print("Enter next value: ")

print("We start here.")
message()
print("The end is here.")
```

Function-Build in Function

- **chr()** Returns a character from the specified Unicode code.

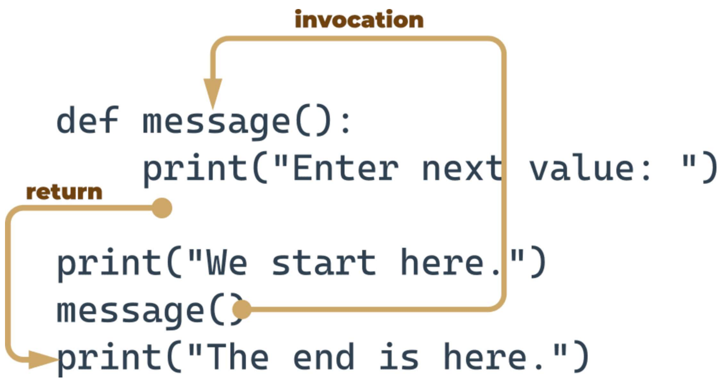
```
x = chr(97)
print(x)
```



```
a
```

- **max() and min**

```
List1=[1,23,11,23,22]
print(max(list1))
print(min(list1))
```



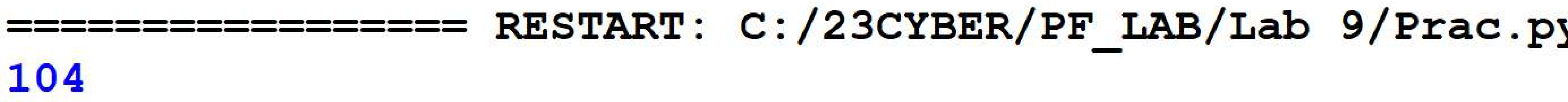
```
def message():
    print("Enter next value: ")

print("We start here.")
message()
print("The end is here.")
```

Function-Build in Function

- **ord()** Convert an integer representing the Unicode of the specified character

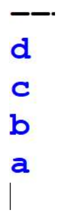
```
x = ord("h")
print(x)
```



```
=================== RESTART: C:/23CYBER/PF_LAB/Lab 9/Prac.py
104
```

- **reversed()** Returns a reversed iterator

```
alph = ["a", "b", "c", "d"]
ralph = reversed(alph)
for x in ralph:
  print(x)
```

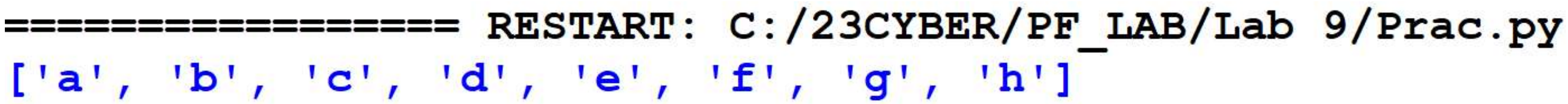


```
d
c
b
a
```

Function-Build in Function

- **sorted()** Returns a sorted list

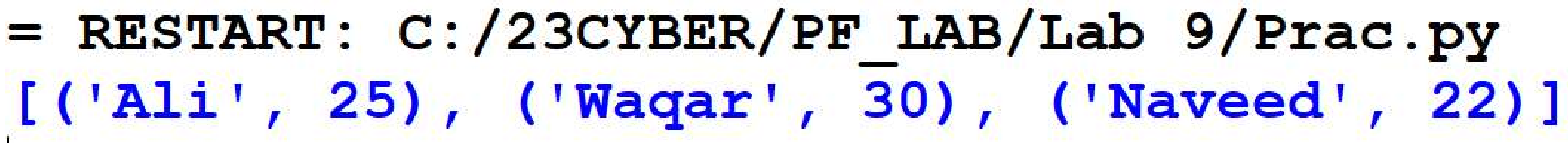
```
a = ("b", "g", "a", "d", "f", "c", "h", "e")
x = sorted(a)
print(x)
```



```
================== RESTART: C:/23CYBER/PF_LAB/Lab 9/Prac.py
['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h']
```

- **zip()** Returns an iterator, from two or more iterators

```
names = ['Ali', 'Waqar', 'Naveed']
ages = [25, 30, 22]
z=zip(names, ages)
list1= list(z)
print(list1)
```



```
= RESTART: C:/23CYBER/PF_LAB/Lab 9/Prac.py
[('Ali', 25), ('Waqar', 30), ('Naveed', 22)]
```

Replace Function

- The `replace()` method replaces a specified phrase with another specified phrase

syntax *string*.replace(*oldvalue, newvalue, count*)

```
txt = "one one was a race horse, two two was one too."

x = txt.replace("one", "three")

print(x)
```

```
three three was a race horse, two two was three too.
```

```
txt = "one one was a race horse, two two was one too."

x = txt.replace("one", "three", 2)

print(x)
```

```
three three was a race horse, two two was one too.
```

Replace Function

```python
original_string = "Remove spaces from this string."

new_string = original_string.replace(" ", "")

print("Original String:", original_string)
print("New String:", new_string)
```

```
Original String: Remove spaces from this string.
New String: Removespacesfromthisstring.
```

Task

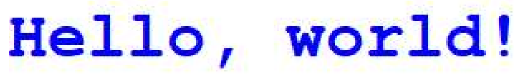
Write a Python program that allows the user to input a string, a word they want to replace in the string, and the new word they want to replace it with. The program should then perform the replacement and display the modified string.

strip() Method

- Removes leading (leftmost) and trailing (rightmost) whitespaces by default

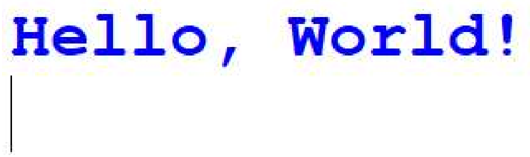
Syntax *string*.strip(*characters*)

```
original_string = "   Hello, world!   "
stripped_string = original_string.strip()
```



```
Hello, world!
```

```
text = "\nHello, World!\n"
stripped_text = text.strip("\n")
print(stripped_text)
```



```
Hello, World!
```

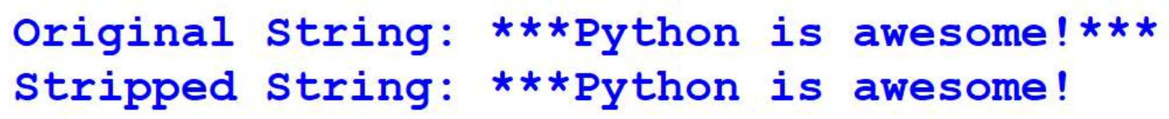
rstrip() Method

- The **rstrip()** method removes any **trailing characters (characters at the end a string)**, space is the default trailing character to remove.

Syntax *string*.rstrip(*characters*)

```
original_string = "***Python is awesome!***"
stripped_string = original_string.rstrip("*")

print("Original String:", original_string)
print("Stripped String:", stripped_string)
```

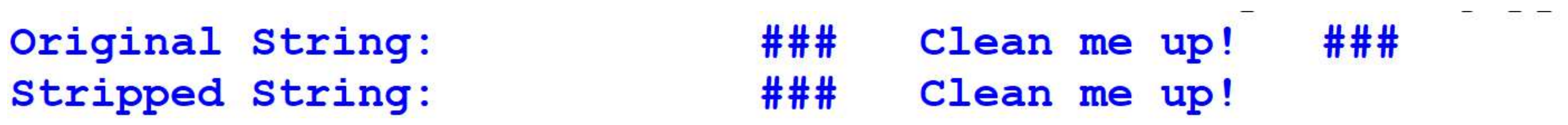


```
Original String: ***Python is awesome!***
Stripped String: ***Python is awesome!
```

rstrip() Method

```
original_string = "###   Clean me up!   ###"
stripped_string = original_string.rstrip("#")

print("Original String:", original
_string)
print("Stripped String:", stripped_string)
```



```
Original String:                ###   Clean me up!   ###
Stripped String:                ###   Clean me up!
```

Create a basic Python program that receives a user input string containing extra whitespaces at the end. Implement the rstrip() method to remove trailing whitespaces and then display the cleaned string.

lstrip() Method

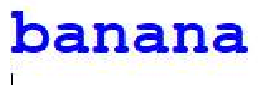
- In Python, the **lstrip()** method is used to remove **leading (leftmost) characters** or a specified **set of characters from a string**. It returns a new string with the leading characters removed.

Syntax *string*.lstrip(*characters*)

```
txt = ",,,,,ssaaww.....banana"

x = txt.lstrip(",.asw")

print(x)
```

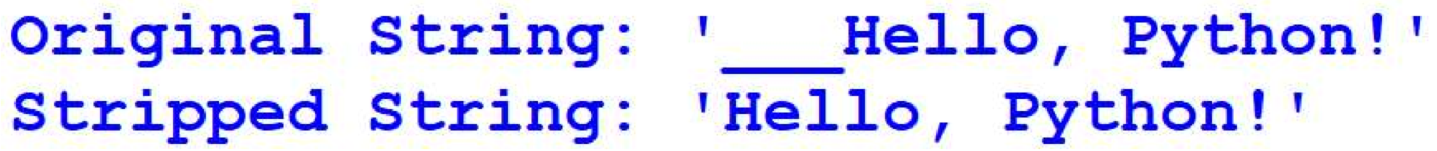


```
banana
```

lstrip() Method

```
original_string = "___Hello, Python!"
stripped_string = original_string.lstrip("_")

print("Original String:", repr(original_string))
print("Stripped String:", repr(stripped_string))
```



```
Original String: '___Hello, Python!'
Stripped String: 'Hello, Python!'
```

Write a Python program that takes user input for a sentence containing underscores. Utilize the lstrip() method to remove these leading characters and display the cleaned sentence.

Split Function

- Python, the **split() function** is a built-in method used to split a string into a **list of substrings** based on a specified delimiter.

Syntax string.split(separator, maxsplit)

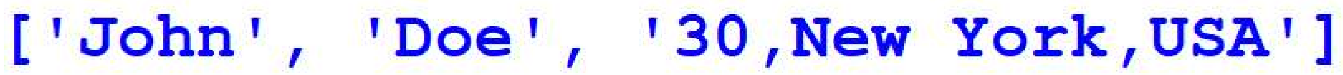
- **Separator:** The seperator based on which the string will be split. If not specified, whitespace characters (spaces, tabs, and newlines) are used by default.
- **maxsplit:** Specifies the maximum number of splits. **Default is -1**, meaning "**all occurrences.**

```
Splitting a string into words
sentence = "Hello world, how are you today?"
words = sentence.split()
print(words)
```

```
['Hello', 'world,', 'how', 'are', 'you', 'today?']
```

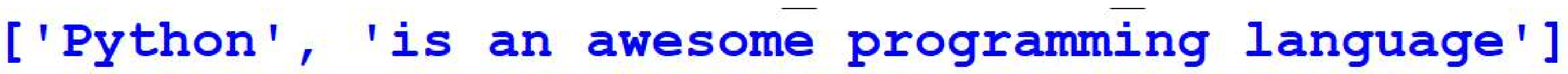
Split Function

```
Splitting a CSV (Comma-Separated Values) string with a specific separator and
limiting the number of splits
csv_data = "John,Doe,30,New York,USA"
fields = csv_data.split(',', 2)
print(fields)
```



```
['John', 'Doe', '30,New York,USA']
```

```
sentence = "Python is an awesome programming language"
words = sentence.split(" ", 1)
print(words)
```



```
['Python', 'is an awesome programming language']
```

Replace Function

- The `replace()` method replaces a specified phrase with another specified phrase

syntax *string*.replace(*oldvalue, newvalue, count*)

```
txt = "one one was a race horse, two two was one too."

x = txt.replace("one", "three")

print(x)
```

```
three three was a race horse, two two was three too.
```

```
txt = "one one was a race horse, two two was one too."

x = txt.replace("one", "three", 2)

print(x)
```

```
three three was a race horse, two two was one too.
```

Replace Function

```
original_string = "Remove spaces from this string."

new_string = original_string.replace(" ", "")

print("Original String:", original_string)
print("New String:", new_string)
```

```
Original String: Remove spaces from this string.
New String: Removespacesfromthisstring.
```

Task

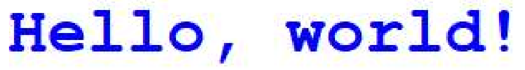
Write a Python program that allows the user to input a string, a word they want to replace in the string, and the new word they want to replace it with. The program should then perform the replacement and display the modified string.

strip() Method

- Removes leading (leftmost) and trailing (rightmost) whitespaces by default

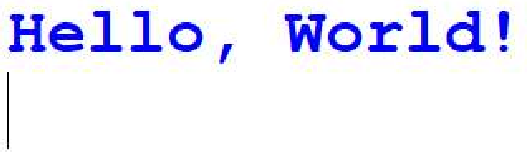
Syntax *string*.strip(*characters*)

```
original_string = "   Hello, world!   "
stripped_string = original_string.strip()
```



```
Hello, world!
```

```
text = "\nHello, World!\n"
stripped_text = text.strip("\n")
print(stripped_text)
```



```
Hello, World!
```

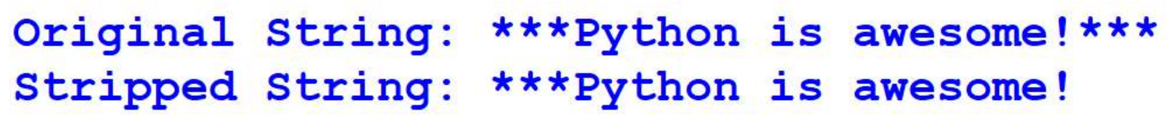
rstrip() Method

- The **rstrip()** method removes any **trailing characters (characters at the end a string)**, space is the default trailing character to remove.

Syntax *string*.rstrip(*characters*)

```
original_string = "***Python is awesome!***"
stripped_string = original_string.rstrip("*")

print("Original String:", original_string)
print("Stripped String:", stripped_string)
```

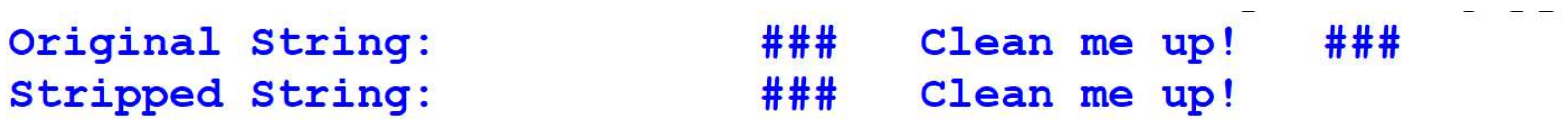


```
Original String: ***Python is awesome!***
Stripped String: ***Python is awesome!
```

rstrip() Method

```
original_string = "###   Clean me up!   ###"
stripped_string = original_string.rstrip("#")

print("Original String:", original
_string)
print("Stripped String:", stripped_string)
```



```
Original String:          ###   Clean me up!   ###
Stripped String:          ###   Clean me up!
```

Create a basic Python program that receives a user input string containing extra whitespaces at the end. Implement the rstrip() method to remove trailing whitespaces and then display the cleaned string.

lstrip() Method

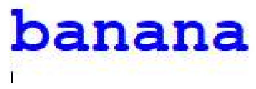
- In Python, the **lstrip()** method is used to remove **leading (leftmost) characters** or a specified **set of characters from a string**. It returns a new string with the leading characters removed.

Syntax `string.lstrip(characters)`

```
txt = ",,,,,ssaaww.....banana"

x = txt.lstrip(",.asw")

print(x)
```

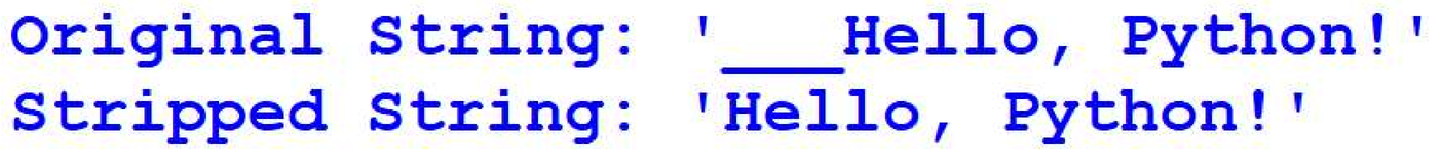


```
banana
```

lstrip() Method

```
original_string = "___Hello, Python!"
stripped_string = original_string.lstrip("_")

print("Original String:", repr(original_string))
print("Stripped String:", repr(stripped_string))
```



```
Original String: '___Hello, Python!'
Stripped String: 'Hello, Python!'
```

Task

Write a Python program that takes user input for a sentence containing underscores. Utilize the lstrip() method to remove these leading characters and display the cleaned sentence.

Split Function

- Python, the **split() function** is a built-in method used to split a string into a **list of substrings** based on a specified delimiter.

  Syntax string.split(separator, maxsplit)

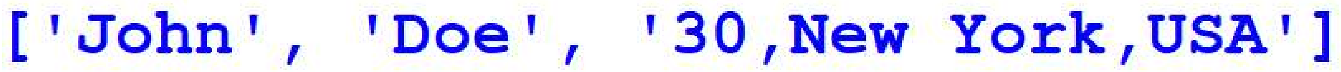
- **Separator:** The seperator based on which the string will be split. If not specified, whitespace characters (spaces, tabs, and newlines) are used by default.
- **maxsplit:** Specifies the maximum number of splits. **Default is -1**, meaning "**all occurrences.**

```
Splitting a string into words
sentence = "Hello world, how are you today?"
words = sentence.split()
print(words)
```

```
['Hello', 'world,', 'how', 'are', 'you', 'today?']
```

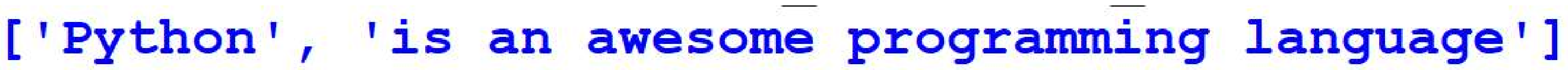
Split Function

```
Splitting a CSV (Comma-Separated Values) string with a specific separator and
limiting the number of splits
csv_data = "John,Doe,30,New York,USA"
fields = csv_data.split(',', 2)
print(fields)
```



```
['John', 'Doe', '30,New York,USA']
```

```
sentence = "Python is an awesome programming language"
words = sentence.split(" ", 1)
print(words)
```



```
['Python', 'is an awesome programming language']
```