```
for i in 'Hello':
  print(i)

```

Below Snippet code returns iterators for string, list and tuple

Double-click (or enter) to edit

```
'Python'.__iter__()
```

```
l1=[1,2,3]
l2=(1,2,3)
print (f'Iterator for list l1 :{l1.__iter__()} and Iterator for tuple l2:{l2.__iter__()}
```

Iterator object is created from string and next function is called till iterable is not exhausted and fina the end to indicate stop of value extraction

```
iterator='Hello'.__iter__()
print(iterator.__next__())
print(iterator.__next__())
print(iterator.__next__())
print(iterator.__next__())
print(iterator.__next__())
print(iterator.__next__())
```

Below code shows very basic illustration of for loop in python

```
iterator='Hello'.__iter__()
while True:
  try:
    print(iterator.__next__())
  except StopIteration:
    break


```

If we need again to iterate we need to again create the Iterator instance and call do the while loop

```
while True:
  try:
    print(iterator.__next__())
  except StopIteration:
    break
```

Double-click (or enter) to edit

Lets define how iterator and Iterables works together through a classes

```
class India:
  def __init__(self):
    self._states=['Himachal','Delhi','Gujarat','Goa']
    self._index=0
  def __iter__(self):
    return IndiaIt(self)
  def __len__(self):
    return len(self._states)

class IndiaIt:
  def __init__(self,India_obj):
    print(f'Entered Iterator')
    self. states obj=India obj
```

```
    self._index=0
  def __iter__(self):
      return self
  def __next__(self):
    if self._index>=len(self._states_obj):
      raise StopIteration
    else:
       _states=self._states_obj._states[self._index]
       print(f'Reading elements from list')
       self._index+=1
       return _states
```

```
India1=India()
for states in India1:
  print(states)
```

```
Entered Iterator
Reading elements from list
Himachal
Reading elements from list
Delhi
Reading elements from list
Gujarat
Reading elements from list
Goa
```

So this is custom class that can be implemented as iterable and Iterator.Also, IndiaIT class is Iterato Iterable (India class) return from iter is Iterator instance.Moreover, for loop entered Iterator first and Iterator instance to read element from collection* **bold text

Double-click (or enter) to edit

```
1
```