**Implementation of Iterator in class**

**Example:**

class MyIterator:

def \_\_init\_\_(self, start, end):

self.current = start

self.end = end

def \_\_iter\_\_(self):

return self

def \_\_next\_\_(self):

if self.current < self.end:

result = self.current

self.current += 1

return result

else:

raise StopIteration

# Example usage:

iterator = MyIterator(1, 5)

for value in iterator:

print(value)

#####The **\_\_iter\_\_()** method returns the iterator object itself, and the **\_\_next\_\_()** method returns the next value from the iterator.

**EXPLANATION**

\_\_init\_\_ **method:**

* + The constructor method \_\_init\_\_ initializes the iterator with a starting value (start) and an ending value (end).
  + It sets the initial current value (self.current) to the starting value and the end value (self.end) to the ending value.

1. \_\_iter\_\_ **method:**
   * The \_\_iter\_\_ method is required for an object to be considered an iterable. It returns the iterator object itself. In this case, it returns self.
2. \_\_next\_\_ **method:**
   * The \_\_next\_\_ method is called to get the next value from the iterator.
   * It checks if the current value is less than the ending value.
   * If true, it sets the result to the current value, increments the current value, and returns the result.
   * If false, it raises the StopIteration exception to signal the end of iteration.

def \_\_next\_\_(self):

if self.current < self.end:

result = self.current

self.current += 1

return result

else:

raise StopIteration

**EXPLANATION**

if self.current < self.end:**:**

* + This is a conditional statement that checks if the current value (self.current) is less than the ending value (self.end).

1. result = self.current**:**
   * If the condition is true (i.e., if self.current is less than self.end), the current value is assigned to the variable result. This is the value that will be returned by the \_\_next\_\_ method.
2. self.current += 1**:**
   * After assigning the current value to result, the self.current value is incremented by 1. This is done to prepare for the next iteration, ensuring that the next value will be returned in the next call to \_\_next\_\_.