Name	:	Muhammad Hassan Ashraf
Class	:	BSCS 2nd A.
Roll no	:	2023132
Assignment	:	03
Submitted to	:	Sir Jamal Abdul ahad

1. **Write a countdown iterator.**

PROGRAM:

```
class CountdownIterator:
    def _init_(self, start):
        self.current = start

    def _iter_(self):
        return self

    def _next_(self):
        if self.current <= 0:
            raise StopIteration
        else:
            self.current -= 1
            return self.current + 1
```

2. **Create an iterator to iterate over a string in python.**

PROGRAM:

```
class StringIterator:
    def _init_(self, input_string):
        self.input_string = input_string
        self.index = 0

    def _iter_(self):
        return self

    def _next_(self):
        if self.index < len(self.input_string):
            result = self.input_string[self.index]
            self.index += 1
            return result
        else:
            raise StopIteration

# Example usage:
string_iterator = StringIterator("Hello, World!")

for char in string_iterator:
    print(char)
```

3. Create an iterator that iterates over power of 2 for a given range.

PROGRAM:

```
def powers_of_two_iterator(start, end):
    current_power = 0
    current_value = 2 ** current_power

    while current_value <= end:
        if current_value >= start:
            yield current_value
        current_power += 1
        current_value = 2 ** current_power

# Example usage:
start_value = 1
end_value = 64

for power_of_two in powers_of_two_iterator(start_value, end_value):
    print(power_of_two)
```

4. Implement a custom iterator that calculates prime numbers upto a given range.

PROGRAM:

```
class PrimeNumbersIterator:
```

```
def _init_(self, end):
    self.end = end
    self.current = 2

def _iter_(self):
    return self

def is_prime(self, num):
    for i in range(2, int(num**0.5) + 1):
        if num % i == 0:
            return False
    return True

def _next_(self):
    while self.current <= self.end:
        if self.is_prime(self.current):
            result = self.current
            self.current += 1
            return result
        else:
            self.current += 1

    raise StopIteration
```

```
# Example usage:
prime_iterator = PrimeNumbersIterator(30)

for prime_number in prime_iterator:
    print(prime_number)
```
