Lab11:

Program No1: Create a countdown iterator.

```
class Countdown:
    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
try:
    start_value = int(input("Enter the starting value for the countdown: "))
except ValueError:
    print("Invalid input. Please enter a valid integer.")
    exit()
countdown = Countdown(start_value)
print("Countdown:")
for number in countdown:
    print(number)
```

…~…

Program No2: Create an iterator to iterate over a string.

I’ve created a special code for iteration over a string that will skip vowels in the given string.

```
class VowelSkipper:
    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):
            raise StopIteration
        else:
            current_char = self.input_string[self.index]
            self.index += 1
            while current_char.lower() in ['a', 'e', 'i', 'o', 'u']:
                if self.index >= len(self.input_string):
                    raise StopIteration
                else:
                    current_char = self.input_string[self.index]
                    self.index += 1
            return current_char
user_input = input("Enter a string: ")
iterator = VowelSkipper(user_input)
modified_string = ''.join(char for char in iterator)
print("Modified String:", modified_string)
```

…~…

Program No3: Create an iterator that iterates over the power of two(square) upto the given range of number.

```
class PowerIterator:
   def __init__(self, Num_range):
      self.Num_range = Num_range
      self.current_number = 0

   def __iter__(self):
      return self

   def __next__(self):
      if self.current_number <= self.Num_range:
         result = self.current_number ** 2
         self.current_number += 1
         return result
      else:
         raise StopIteration

try:
   Num_range = int(input("Enter the maximum number for the iterator: "))
except ValueError:
   print("Invalid input. Please enter a valid integer.")
   exit()
```

```
power_iterator = PowerIterator(Num_range)


print("Powers of Two Applied to Numbers:")
for result in power_iterator:
    print(result)
```

...~...

Program No4: Implement a custom iterator in pyhton that iterate over the given range of number and display out all the prime numbers.

```
class PrimeNumber:
    def __init__(self, Num_range):
        self.Num_range = Num_range
        self.current_number = 2


    def __iter__(self):
        return self


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


    def __next__(self):
        while self.current_number <= self.Num_range:
            if self.is_prime(self.current_number):
                result = self.current_number
                self.current_number += 1
                return result
            else:
                self.current_number += 1
        raise StopIteration


try:
    Num_range = int(input("Enter the maximum number for the iterator: "))
except ValueError:
    print("Invalid input. Please enter a valid integer.")
    exit()
prime_iterator = PrimeNumber(Num_range)


print("Prime Numbers:")
for prime_number in prime_iterator:
    print(prime_number)
```

...The End...