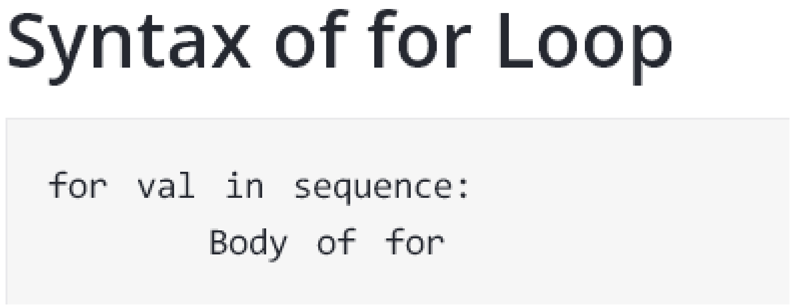
For loop

- For loops iterate over a given sequence.



Syntax of for Loop

```
for val in sequence:
        Body of for
```

- Here, val is the variable that takes the value of the item inside the sequence on each iteration.
- Loop continues until we reach the last item in the sequence. The body of for loop is separated from the rest of the code using indentation.

```python
# Program to find the sum of all numbers stored in a list

# List of numbers
numbers = [6, 5, 3, 8, 4, 2, 5, 4, 11]

# variable to store the sum
totalSum = 0

# iterate over the list
for val in numbers:
    totalSum = totalSum + val

print("The sum is", totalSum)
```

Range Function

- We can generate a sequence of numbers using range() function.
- range(10) will generate numbers from 0 to 9 (10 numbers).

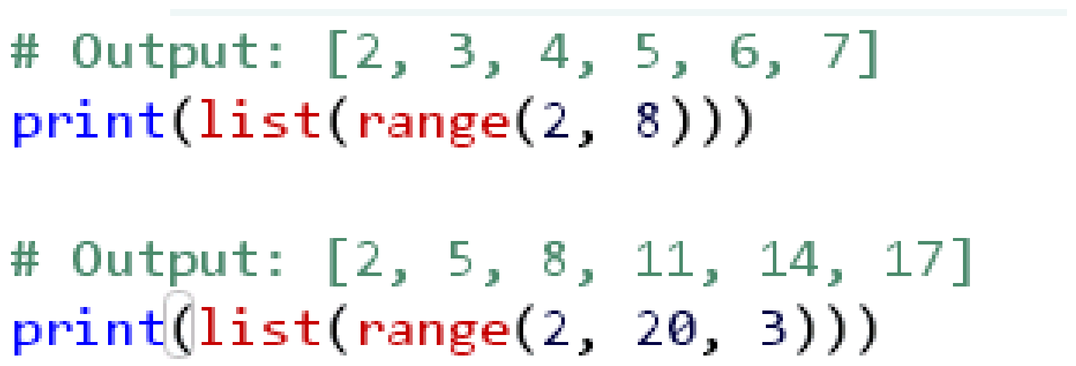
```
# Output: range(0, 10)
print(range(10))
```

- To force this function to output all the items, we can use the function list().

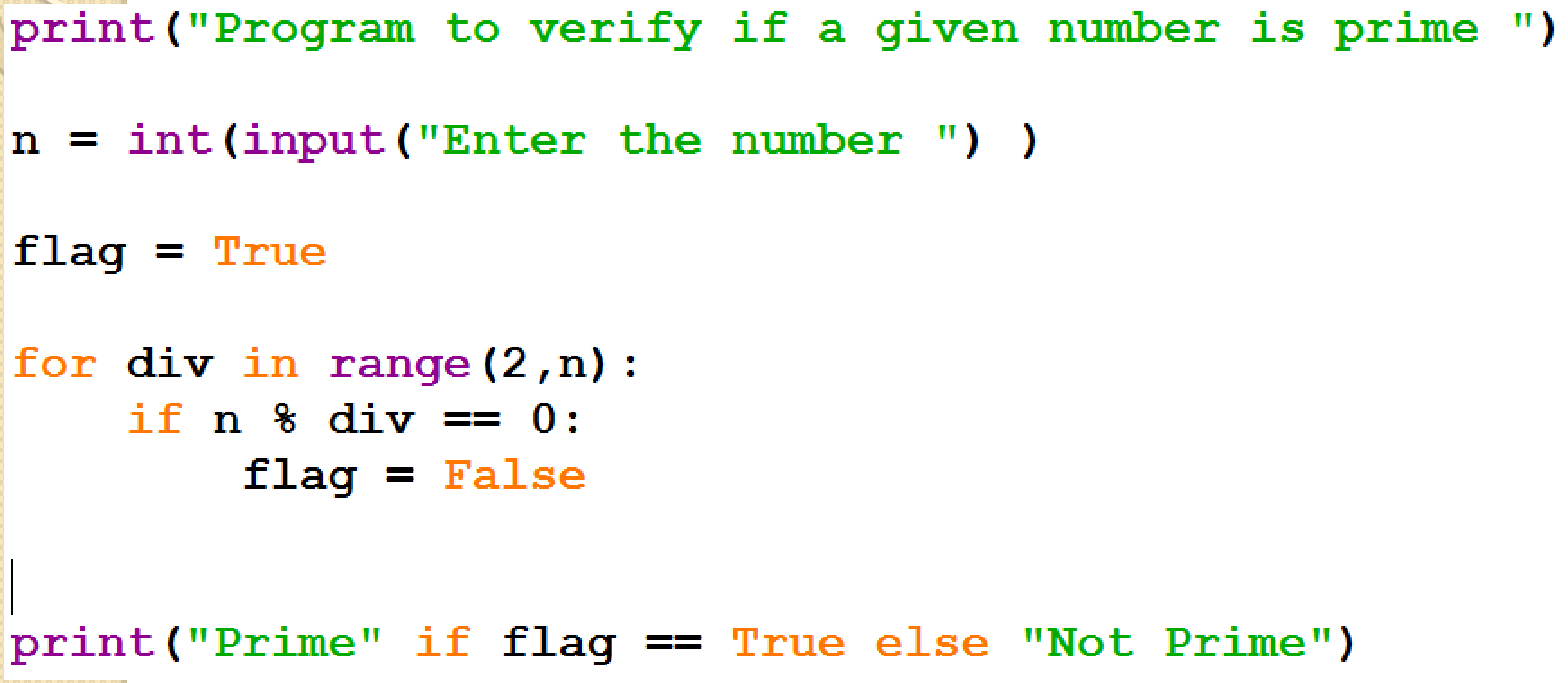
```
# Output: [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
print(list(range(10)))
```

Range function

- We can also define the start, stop and step size as range(start,stop,step size).
- step size defaults to 1 if not provided.



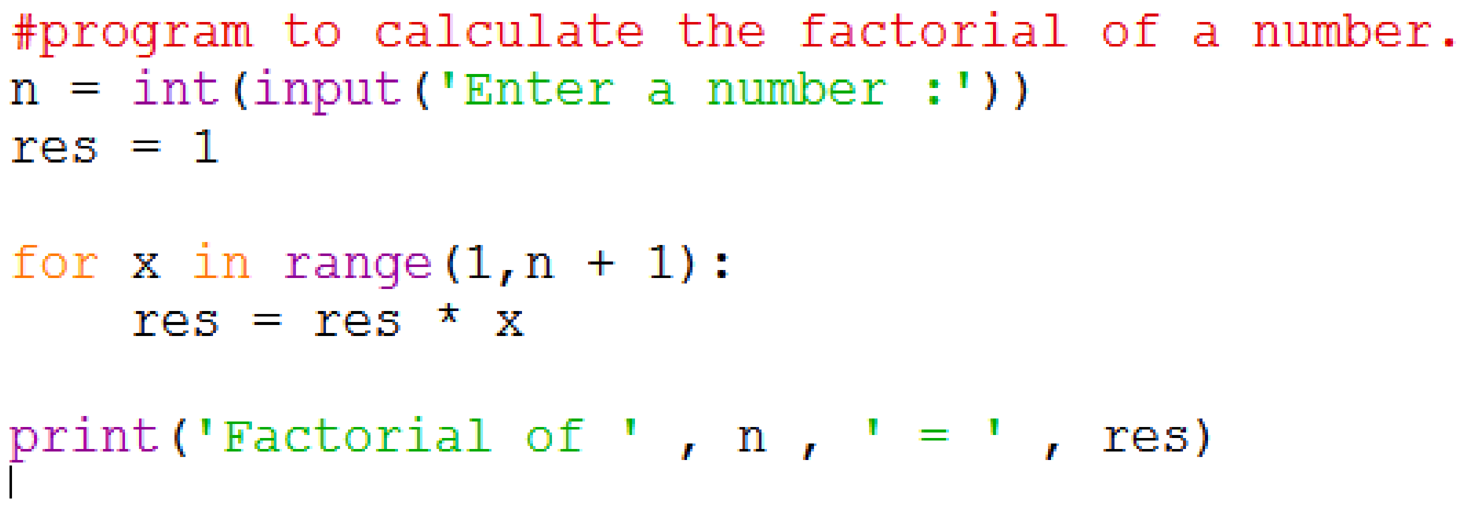
```
# Output: [2, 3, 4, 5, 6, 7]
print(list(range(2, 8)))

# Output: [2, 5, 8, 11, 14, 17]
print(list(range(2, 20, 3)))
```



```python
print("Program to verify if a given number is prime ")

n = int(input("Enter the number ") )

flag = True

for div in range(2,n):
    if n % div == 0:
        flag = False


print("Prime" if flag == True else "Not Prime")
```

Exercise

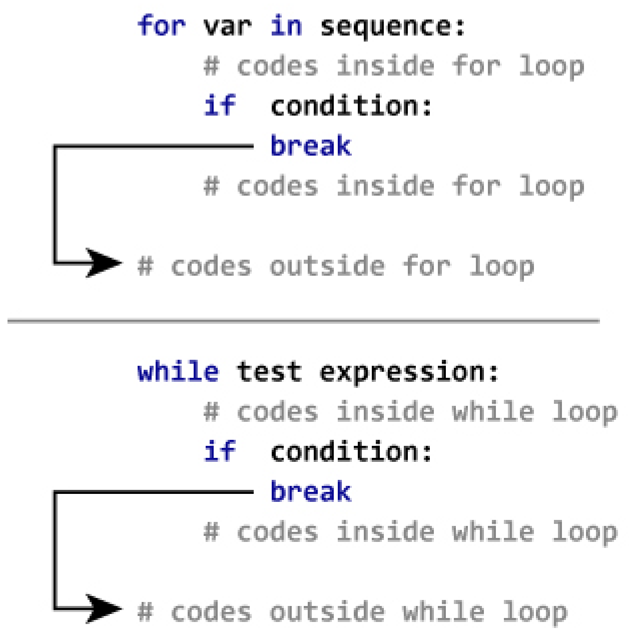
- Write a program to calculate the factorial of a given number.



```python
#program to calculate the factorial of a number.
n = int(input('Enter a number :'))
res = 1

for x in range(1,n + 1):
    res = res * x

print('Factorial of ' , n , ' = ' , res)
```

break and continue

- Loops iterate over a block of code until test expression is false, but sometimes we wish to terminate the current iteration or even the whole loop without checking test expression.
- **break statement**
- The break statement terminates the loop containing it. Control of the program flows to the statement immediately after the body of the loop.
- If break statement is inside a nested loop (loop inside another loop), break will terminate the innermost loop.



```
for var in sequence:
    # codes inside for loop
    if  condition:
        break
    # codes inside for loop

# codes outside for loop
```

```
while test expression:
    # codes inside while loop
    if  condition:
        break
    # codes inside while loop

# codes outside while loop
```

Exercise

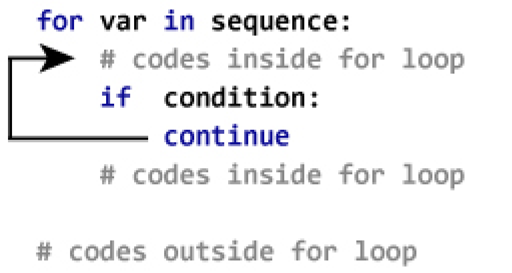
- Write a program to check if a given number is prime or not.

- If a number is divisible by any number between 2 and n-1, its not prime, otherwise prime

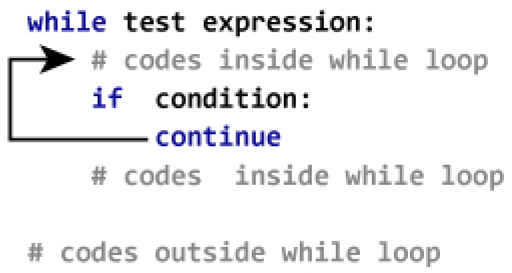
```
print("Program to verify if a given number is prime ")

n = int(input("Enter the number ") )

flag = True

for div in range(2,n):
    if n % div == 0:
        flag = False
        break

print("div = " ,div)
print("Prime" if flag == True else "Not Prime")
```

continue statement

- The continue statement is used to skip the rest of the code inside a loop for the current iteration only. Loop does not terminate but continues on with the next iteration.



```
for var in sequence:
    # codes inside for loop
    if  condition:
        continue
    # codes inside for loop

# codes outside for loop
```



```
while test expression:
    # codes inside while loop
    if  condition:
        continue
    # codes  inside while loop

# codes outside while loop
```

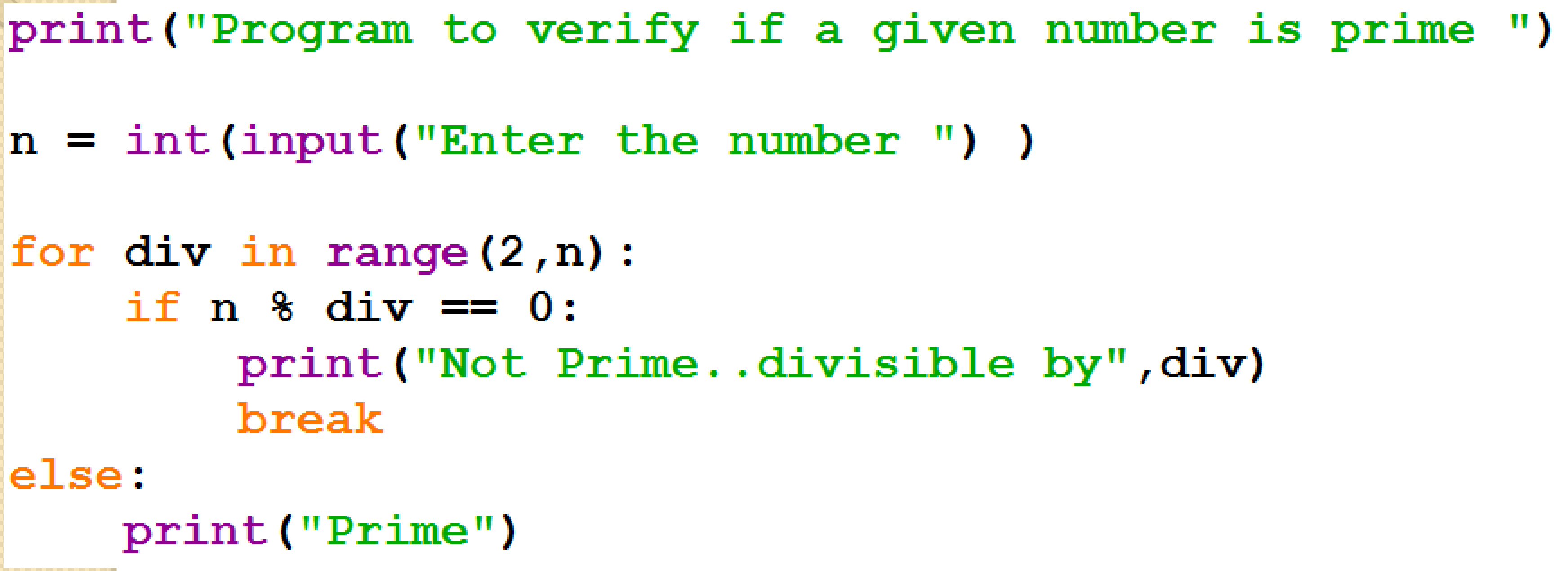
```
for val in range(20):
    if val % 3 == 0:
        continue
    print(val)

print("The end")
```

for loop with else

- A for loop can have an optional else block as well. The else part is executed if the items in the sequence used in for loop exhausts.
- break statement can be used to stop a for loop. In such case, the else part is ignored.
- Hence, a for loop's else part runs if no break occurs.

```
#program to demonstrate for with else
digits = [0, 1, 5]

for i in digits:
    print(i)
else:
    print("No items left.")
```

for loop with else



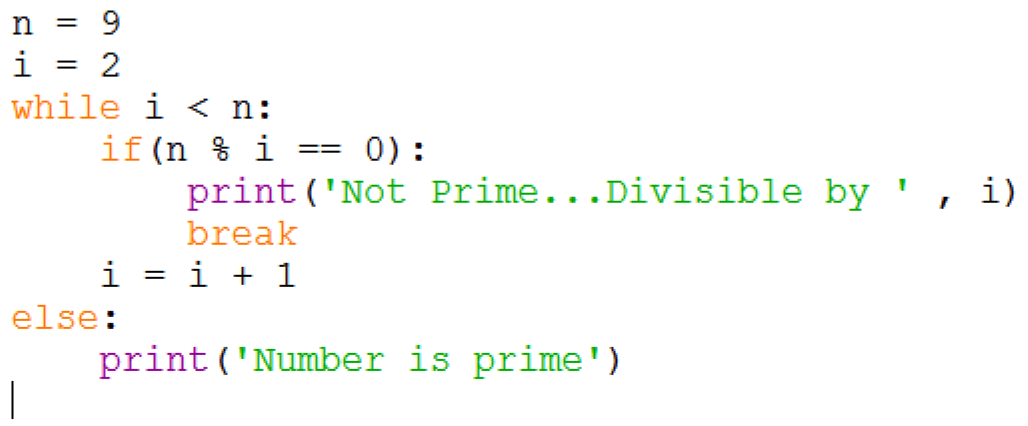
```
print("Program to verify if a given number is prime ")

n = int(input("Enter the number ") )

for div in range(2,n):
    if n % div == 0:
        print("Not Prime..divisible by",div)
        break
else:
    print("Prime")
```

Nested Loop

```
# Python program to display all the prime numbers within an interval

lower = int(input("Enter lower range: "))
upper = int(input("Enter upper range: "))

print("Prime numbers between",lower,"and",upper,"are:")

for num in range(lower,upper + 1):
   # prime numbers are greater than 1
   if num > 1:
       for i in range(2,num):
           if (num % i) == 0:
               break
       else:
           print(num)
```

while loop with else

- Same as that of for loop, we can have an optional else block with while loop as well.



```python
n = 9
i = 2
while i < n:
    if(n % i == 0):
        print('Not Prime...Divisible by ' , i)
        break
    i = i + 1
else:
    print('Number is prime')
```

Pass statement

- Suppose we have a loop or a function that is not implemented yet, but we want to implement it in the future.
- They cannot have an empty body.
- We use the pass statement to construct a body that does nothing.

```
# pass is just a placeholder for
# functionality to be added later.
sequence = {'p', 'a', 's', 's'}
for val in sequence:
    pass
```