COL1000
Introduction to Programming

Priyanka Golia

Most (if not all) of the content is borrowed from Prof. Subodh Kumar's slides

To test if a given number is a prime number or not.

```
if number < 2:
   print(number, 'is not prime')
else: # number is >= 2
   if number == 2:
      print(number, 'is prime')
   else: # number is > 2
      if number % 2 == 0: # it’s even
         print(number, 'is not prime')
      else: # Neither 2 nor even
         if number == 3:
            print(number, 'is prime')
         else:
            if number % 3 == 0: # divisible by 3
               print(number, 'is not prime')
            else: # Odd, >3, Not divisible by 3
               if number == 5:
                  print(number, 'is prime')
               else:
                  if number % 5 == 0: # divisible by 5
                     print(number, 'is not prime')
                  else:
                     if number == 7:
                        print(number, 'is prime')
                     else:
                        if number % 7 == 0: # divisible by 7
                           print(number, 'is not prime')
                        else:
                           print(“C'mon! Stop already.")
```

To test if a given number is a prime number or not.

```
if number < 2:
    print(number, 'is not prime')
else: # number is >= 2
    if number == 2:
        print(number, 'is prime')
    else: # number is > 2
        if number % 2 == 0: # it’s even
            print(number, 'is not prime')
        else: # Neither 2 nor even
            if number == 3:
                print(number, 'is prime')
            else:
                if number % 3 == 0: # divisibl
                    print(number, 'is not prime
                else: # Odd, >3, Not divisible
                    if number == 5:
                        print(number, 'is prime'
                    else:
                        if number % 5 == 0: # di
                            print(number, 'is not
                        else:
                            if number == 7:
                                print(number, 'is
                            else:
                                if number % 7 == 0
                                    print(number, '
                                else:
                                    print(“C'mon! S
```

```
if number < 2:
    print(number, 'is not prime')
elif number == 2:
    print(number, 'is prime')
elif number % 2 == 0: # it’s even
    print(number, 'is not prime')
elif number == 3:
    print(number, 'is prime')
elif number % 3 == 0: # divisible by
3
    print(number, 'is not prime')
elif number == 5:
    print(number, 'is prime')
elif number % 5 == 0: # divisible by
5
    print(number, 'is not prime')
elif number == 7:
    print(number, 'is prime')
elif number % 7 == 0: # divisible by
7
    print(number, 'is not prime')
elif number == 11:
    print(number, 'is prime')
else: print("C'mon! Enough
already.")
```

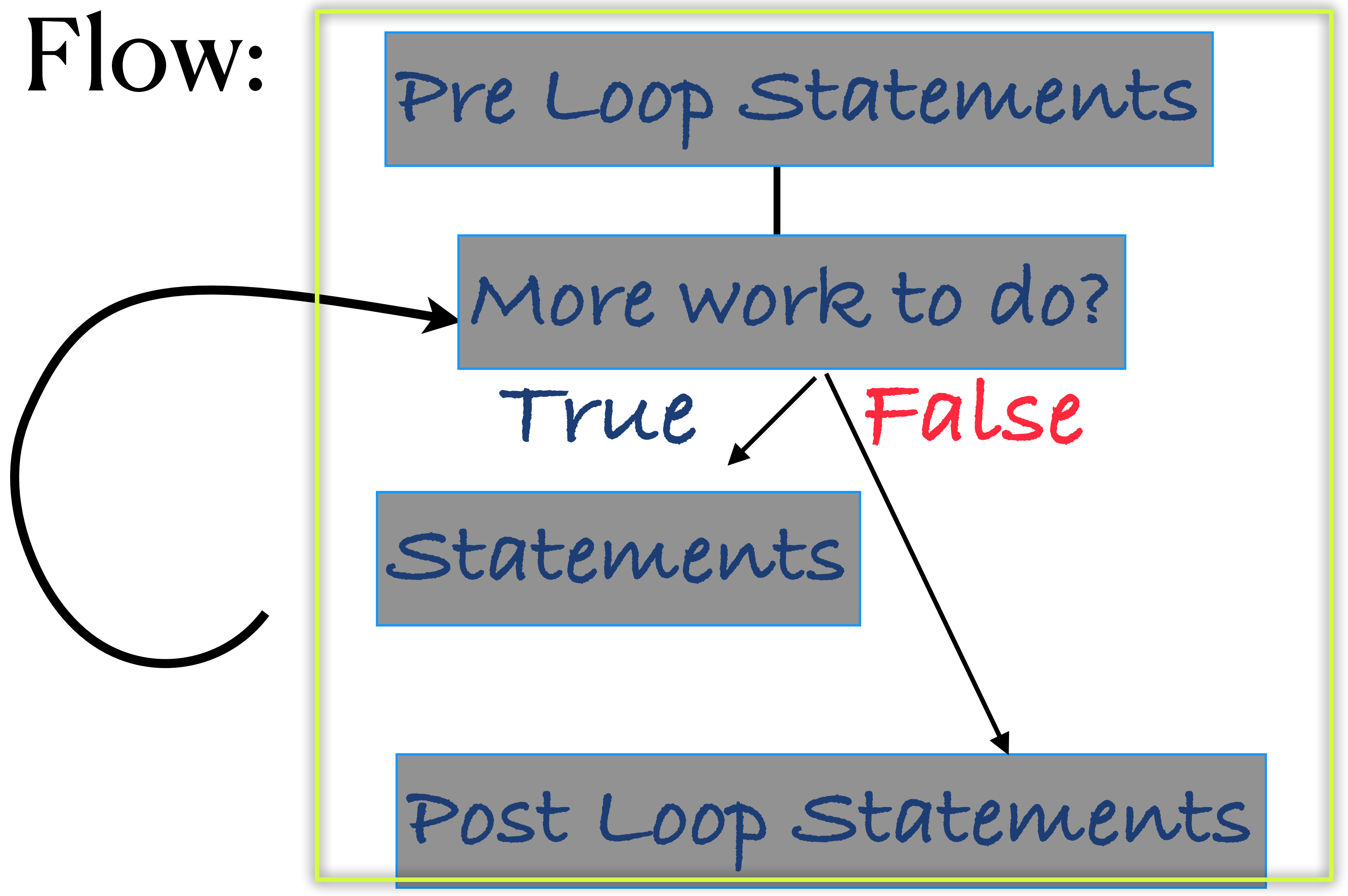
```
if number < 2:
   print(number, 'is not prime')
elif number == 2:
   print(number, 'is prime')
elif number % 2 == 0: # it’s even
   print(number, 'is not prime')
elif number == 3:
   print(number, 'is prime')
elif number % 3 == 0: # divisible by
3
   print(number, 'is not prime')
elif number == 5:
   print(number, 'is prime')
elif number % 5 == 0: # divisible by
5
   print(number, 'is not prime')
elif number == 7:
   print(number, 'is prime')
elif number % 7 == 0: # divisible by
7
   print(number, 'is not prime')
elif number == 11:
   print(number, 'is prime')
else: print("C'mon! Enough
already.")
```

Keep going

```
if number < 2:
   print(number, 'is not prime')
elif number == 2:
   print(number, 'is prime')
elif number % 2 == 0: # it’s even
   print(number, 'is not prime')
elif number == 3:
   print(number, 'is prime')
elif number % 3 == 0: # divisible by
3
   print(number, 'is not prime')
elif number == 5:
   print(number, 'is prime')
elif number % 5 == 0: # divisible by
5
   print(number, 'is not prime')
elif number == 7:
   print(number, 'is prime')
elif number % 7 == 0: # divisible by
7
   print(number, 'is not prime')
elif number == 11:
   print(number, 'is prime')
else: print("C'mon! Enough
already.")
```

Keep going

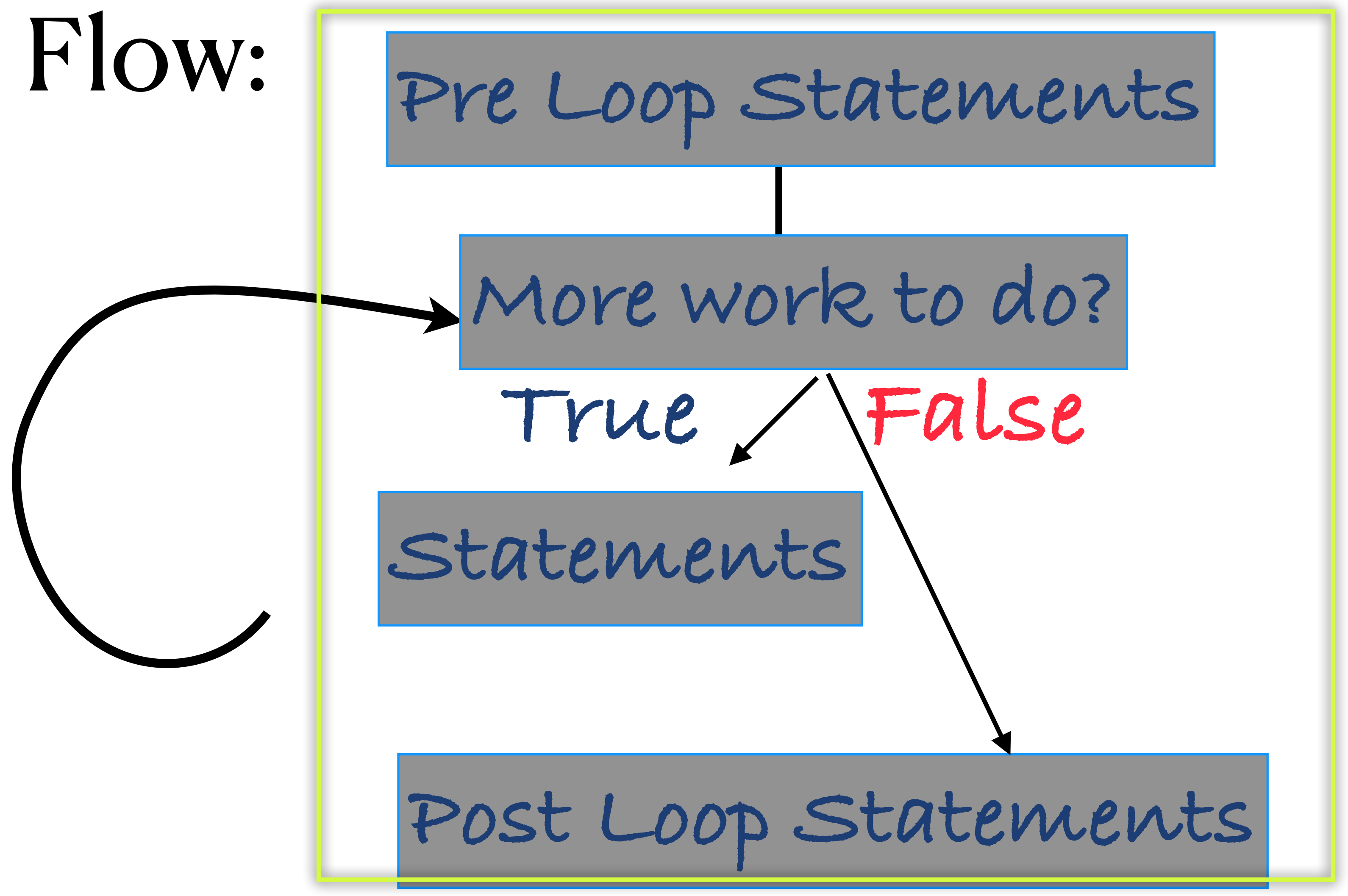
Control Flow:



```
if number < 2:
   print(number, 'is not prime')
elif number == 2:
   print(number, 'is prime')
elif number % 2 == 0: # it's even
   print(number, 'is not prime')
elif number == 3:
   print(number, 'is prime')
elif number % 3 == 0: # divisible by
3
   print(number, 'is not prime')
elif number == 5:
   print(number, 'is prime')
elif number % 5 == 0: # divisible by
5
   print(number, 'is not prime')
elif number == 7:
   print(number, 'is prime')
elif number % 7 == 0: # divisible by
7
   print(number, 'is not prime')
elif number == 11:
   print(number, 'is prime')
else: print("C'mon! Enough
already.")
```

Keep going

Control Flow:



We call this a "loop"

Do something Similar &
Keep going until "done"

Loops

For loops

- Example: *Sum positive integers upto n*
- The “for” construct

i is renamed to all values in the range, one after another

```
sum = 0
for i in range(1, n+1):
   sum = sum + i
print('Sum is', sum)
```

Need to “remember”: use variable

Statement after the loop statement

☞ Recall:

This Creates an object of the “range” type

By default, a single number denotes the end of the range

(This end is non-inclusive)

The range begins at 0 by default

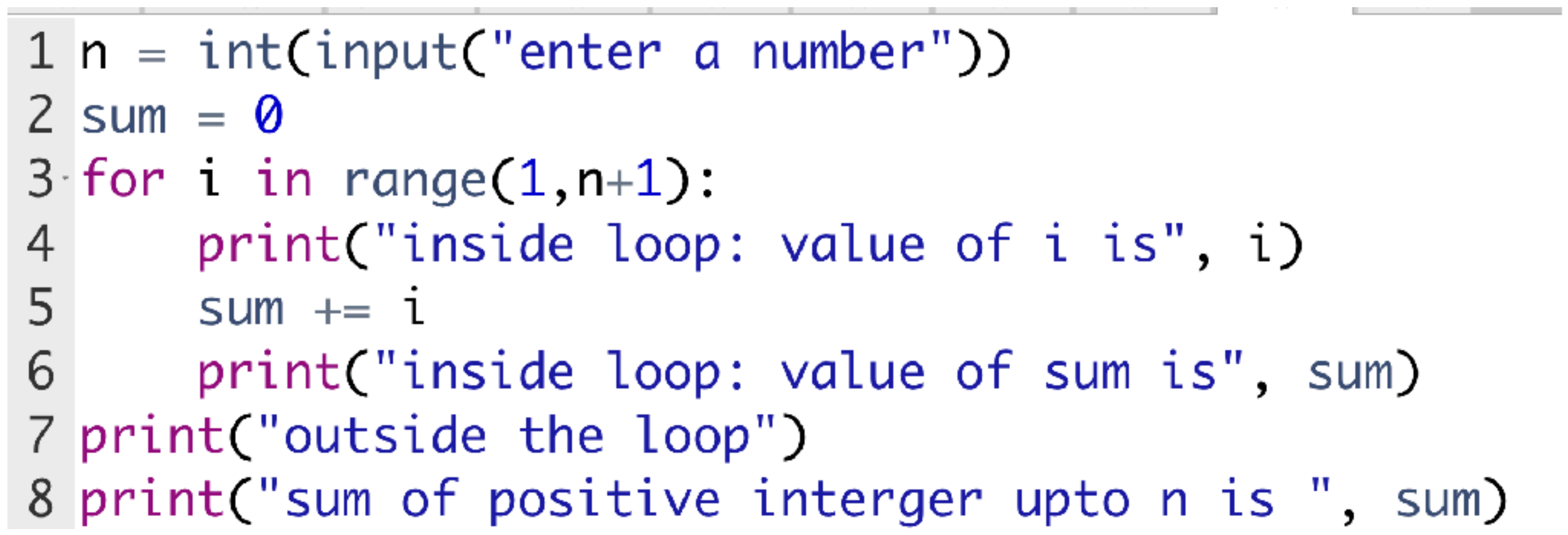
- Loops have three parts:

Before Loop pattern Condition

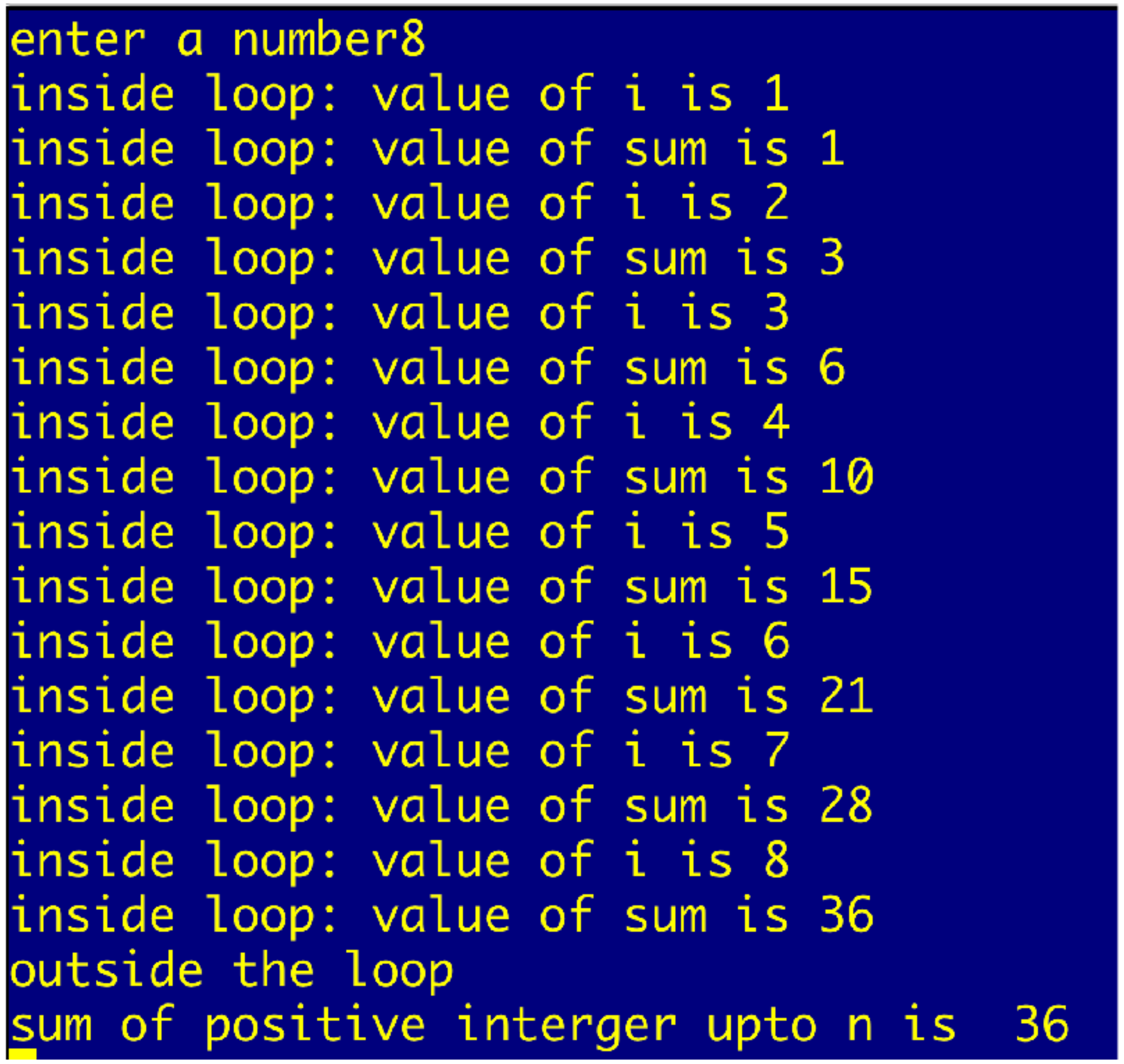
➡ Initialization; Progress; Termination

Generally, there is a loop control variable (that drives the pattern and termination)

For loops

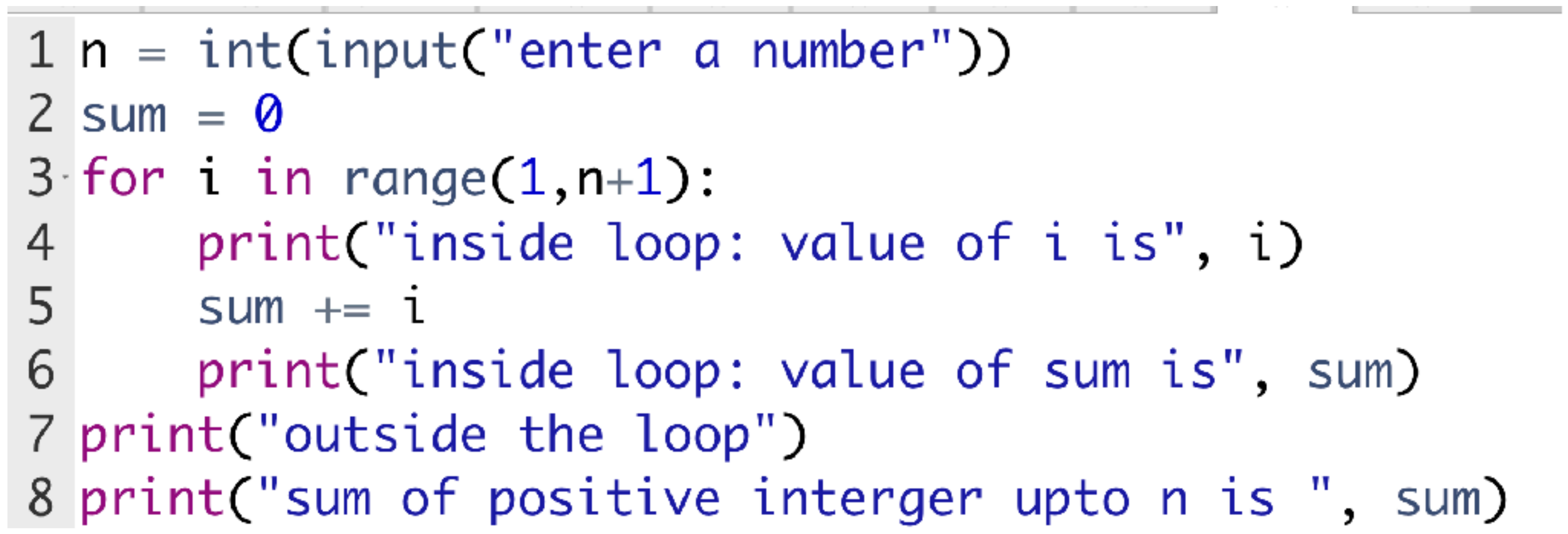


```python
n = int(input("enter a number"))
sum = 0
for i in range(1,n+1):
    print("inside loop: value of i is", i)
    sum += i
    print("inside loop: value of sum is", sum)
print("outside the loop")
print("sum of positive interger upto n is ", sum)
```

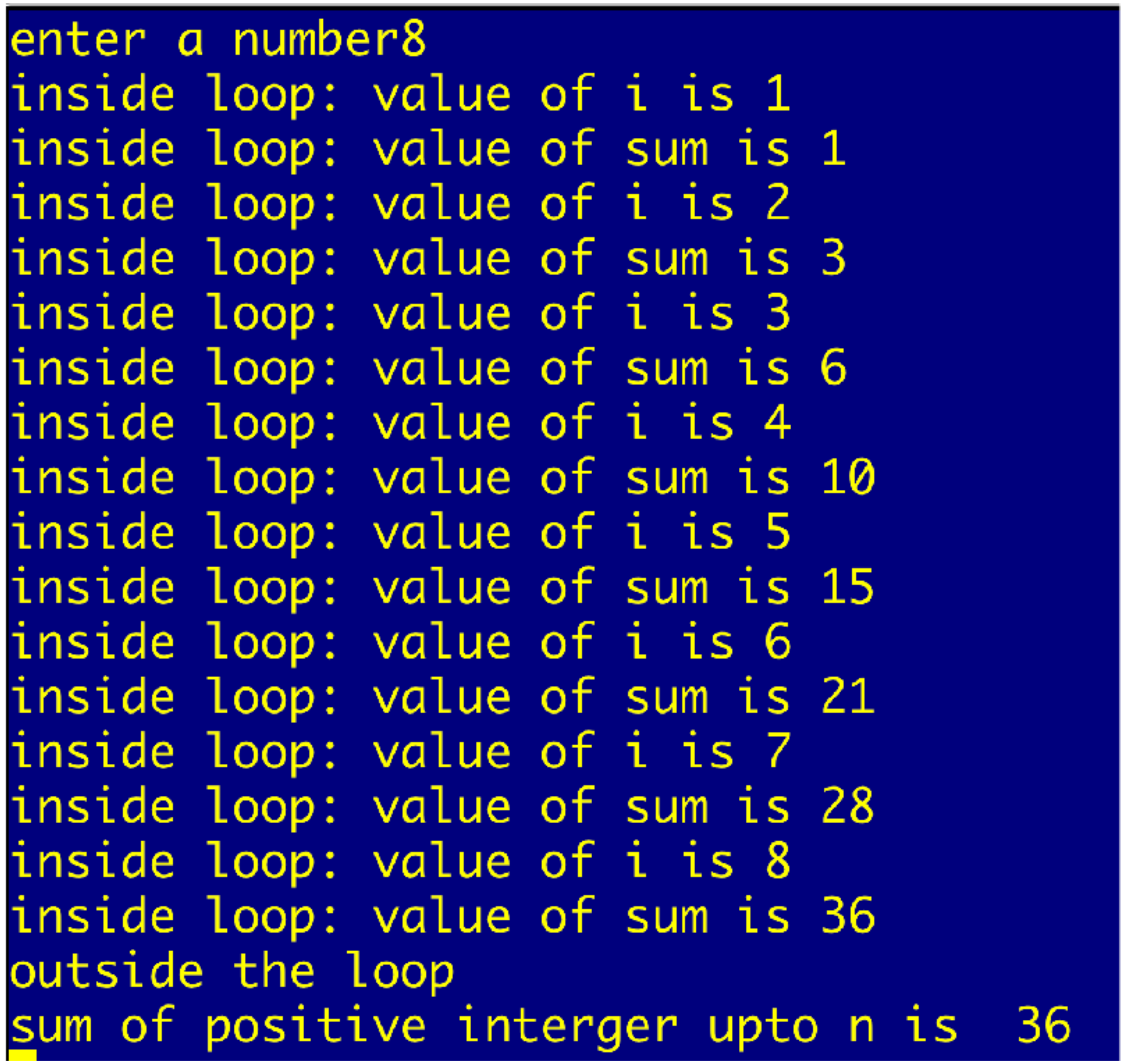


```
enter a number8
inside loop: value of i is 1
inside loop: value of sum is 1
inside loop: value of i is 2
inside loop: value of sum is 3
inside loop: value of i is 3
inside loop: value of sum is 6
inside loop: value of i is 4
inside loop: value of sum is 10
inside loop: value of i is 5
inside loop: value of sum is 15
inside loop: value of i is 6
inside loop: value of sum is 21
inside loop: value of i is 7
inside loop: value of sum is 28
inside loop: value of i is 8
inside loop: value of sum is 36
outside the loop
sum of positive interger upto n is  36
```

For loops



```python
n = int(input("enter a number"))
sum = 0
for i in range(1,n+1):
    print("inside loop: value of i is", i)
    sum += i
    print("inside loop: value of sum is", sum)
print("outside the loop")
print("sum of positive interger upto n is ", sum)
```



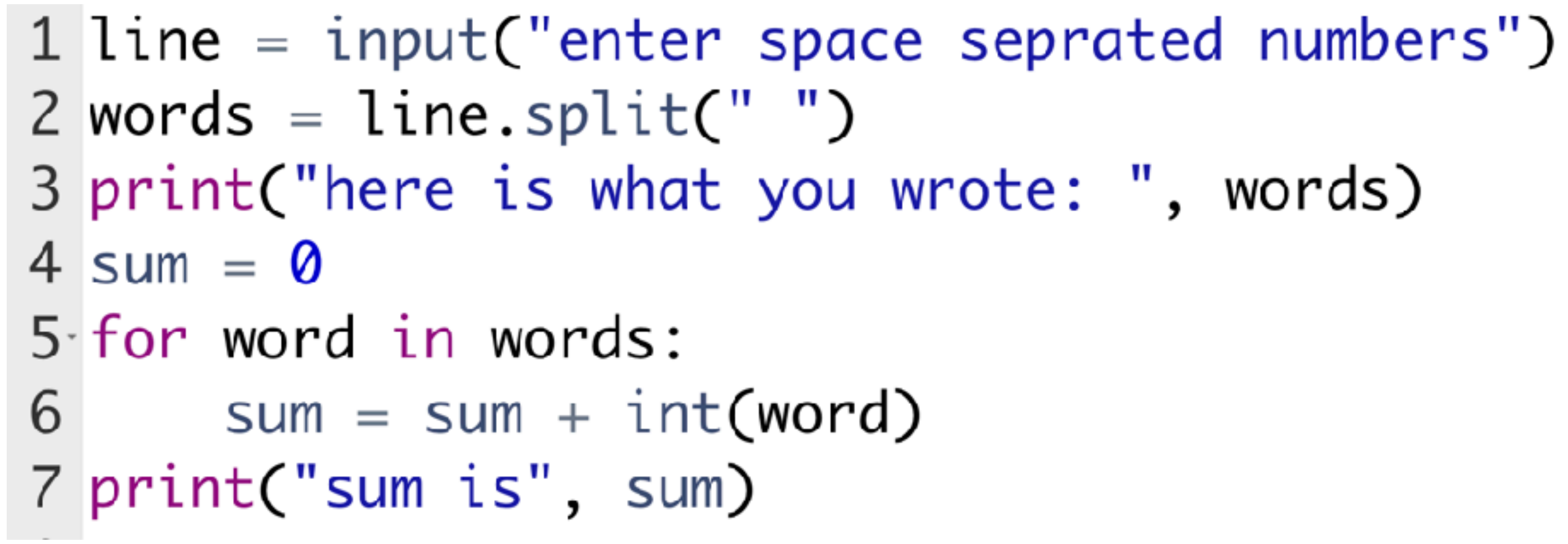
```
enter a number8
inside loop: value of i is 1
inside loop: value of sum is 1
inside loop: value of i is 2
inside loop: value of sum is 3
inside loop: value of i is 3
inside loop: value of sum is 6
inside loop: value of i is 4
inside loop: value of sum is 10
inside loop: value of i is 5
inside loop: value of sum is 15
inside loop: value of i is 6
inside loop: value of sum is 21
inside loop: value of i is 7
inside loop: value of sum is 28
inside loop: value of i is 8
inside loop: value of sum is 36
outside the loop
sum of positive interger upto n is  36
```

The most important aspect of loop design:
How each individual iteration moves the ball closer to the goal: how it modifies the partial solution from the previous iteration to provide to the next iteration

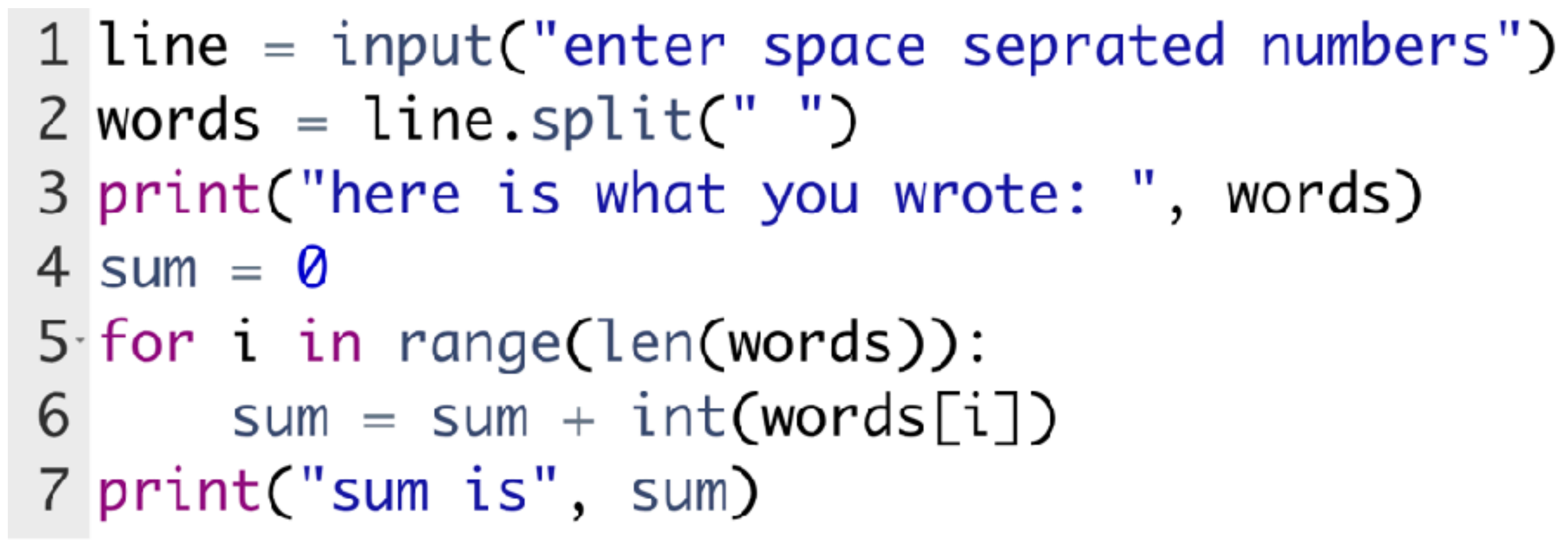
Look at the value of sum

For loops

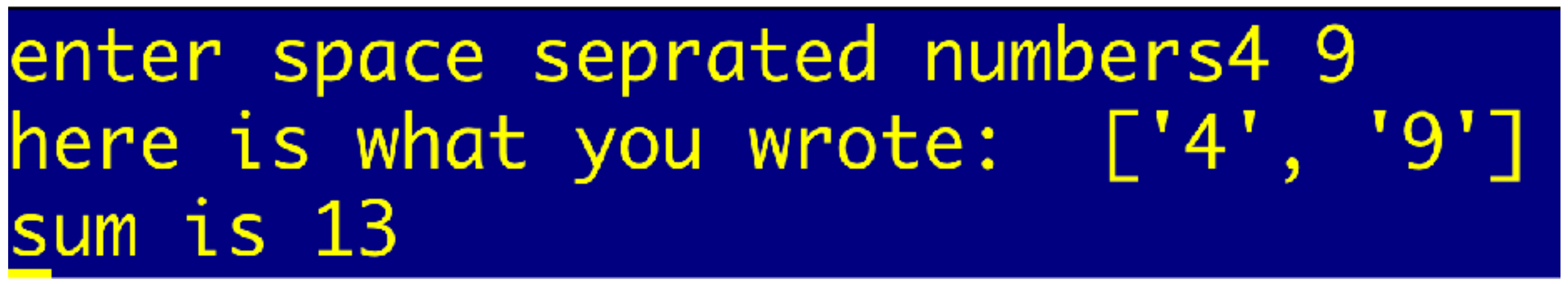
Sum of integers provided by the user



```python
line = input("enter space seprated numbers")
words = line.split(" ")
print("here is what you wrote: ", words)
sum = 0
for word in words:
    sum = sum + int(word)
print("sum is", sum)
```



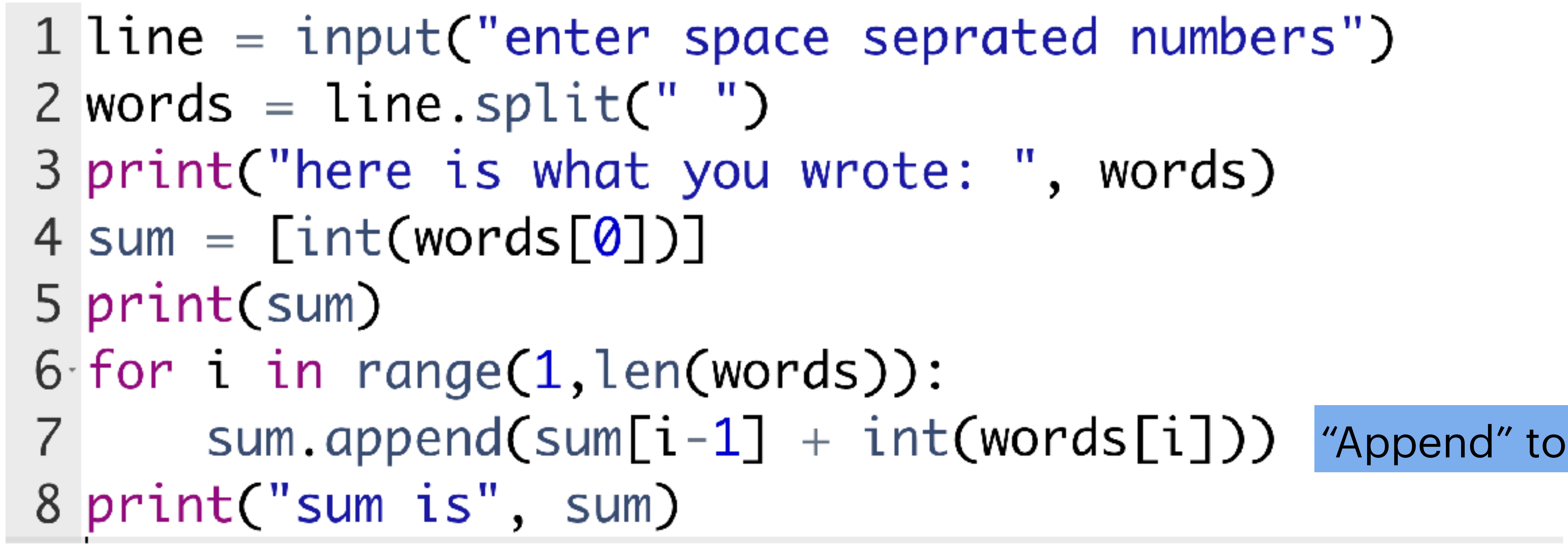
```python
line = input("enter space seprated numbers")
words = line.split(" ")
print("here is what you wrote: ", words)
sum = 0
for i in range(len(words)):
    sum = sum + int(words[i])
print("sum is", sum)
```



```
enter space seprated numbers4 9
here is what you wrote:  ['4', '9']
sum is 13
```

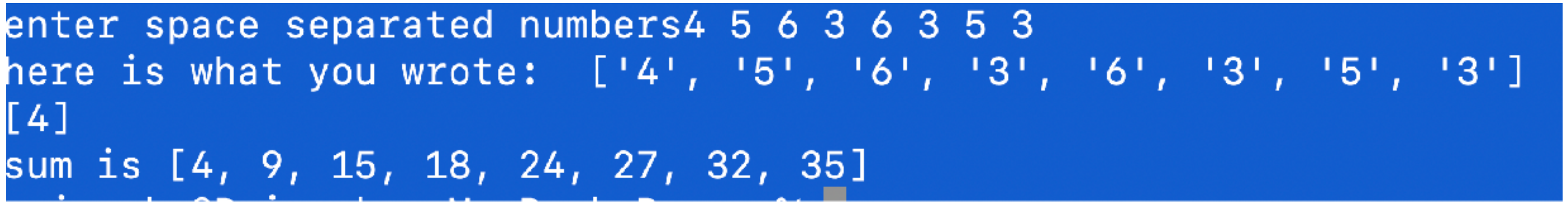
For loops

Provide all partial sums



```python
line = input("enter space seprated numbers")
words = line.split(" ")
print("here is what you wrote: ", words)
sum = [int(words[0])]
print(sum)
for i in range(1,len(words)):
    sum.append(sum[i-1] + int(words[i]))
print("sum is", sum)
```

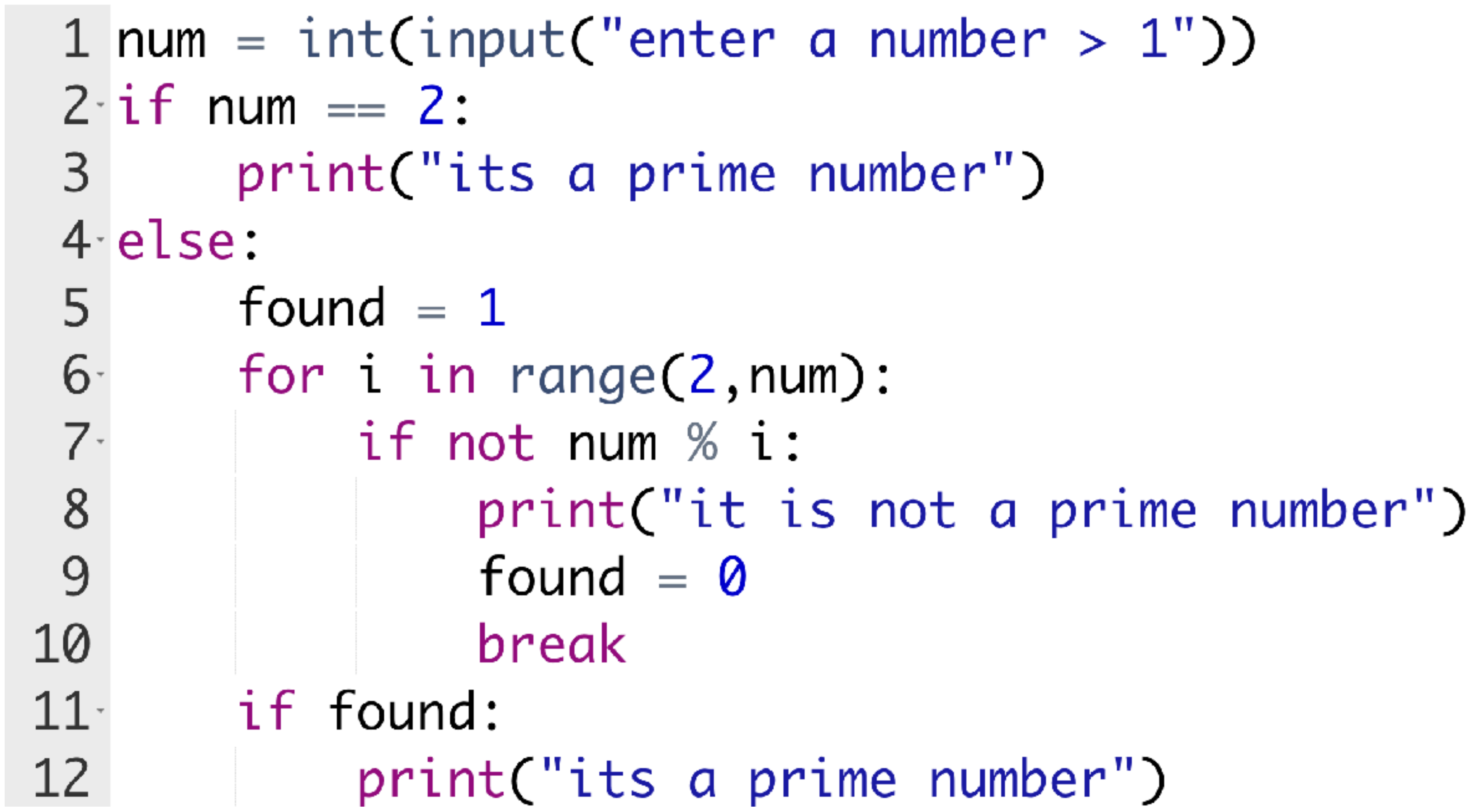
"Append" to add objects to the list.



```
enter space separated numbers4 5 6 3 6 3 5 3
here is what you wrote:  ['4', '5', '6', '3', '6', '3', '5', '3']
[4]
sum is [4, 9, 15, 18, 24, 27, 32, 35]
```

For loops

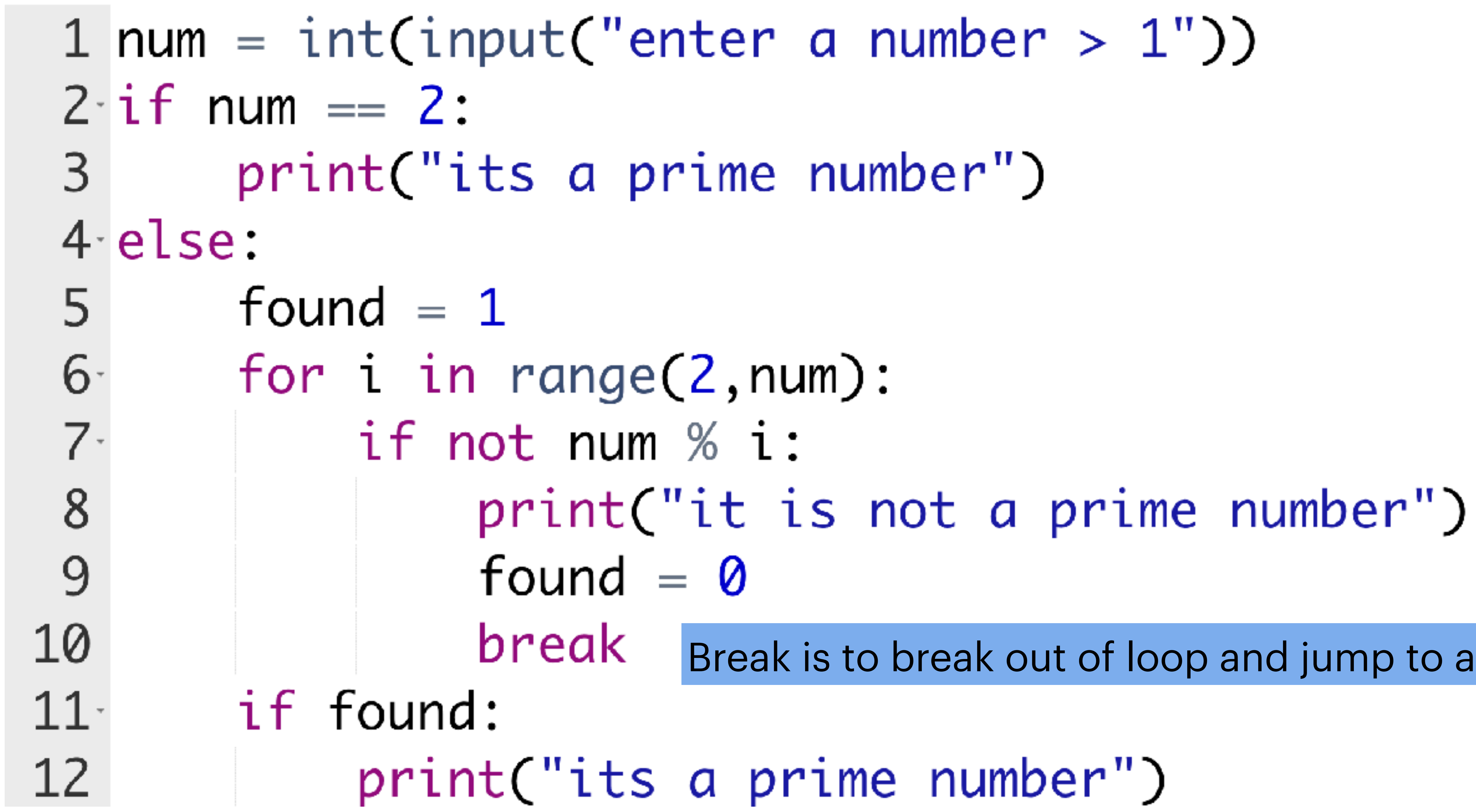
Test if a given number n is a prime number, n > 1



```python
num = int(input("enter a number > 1"))
if num == 2:
    print("its a prime number")
else:
    found = 1
    for i in range(2,num):
        if not num % i:
            print("it is not a prime number")
            found = 0
            break
    if found:
        print("its a prime number")
```

For loops

Test if a given number n is a prime number, n > 1

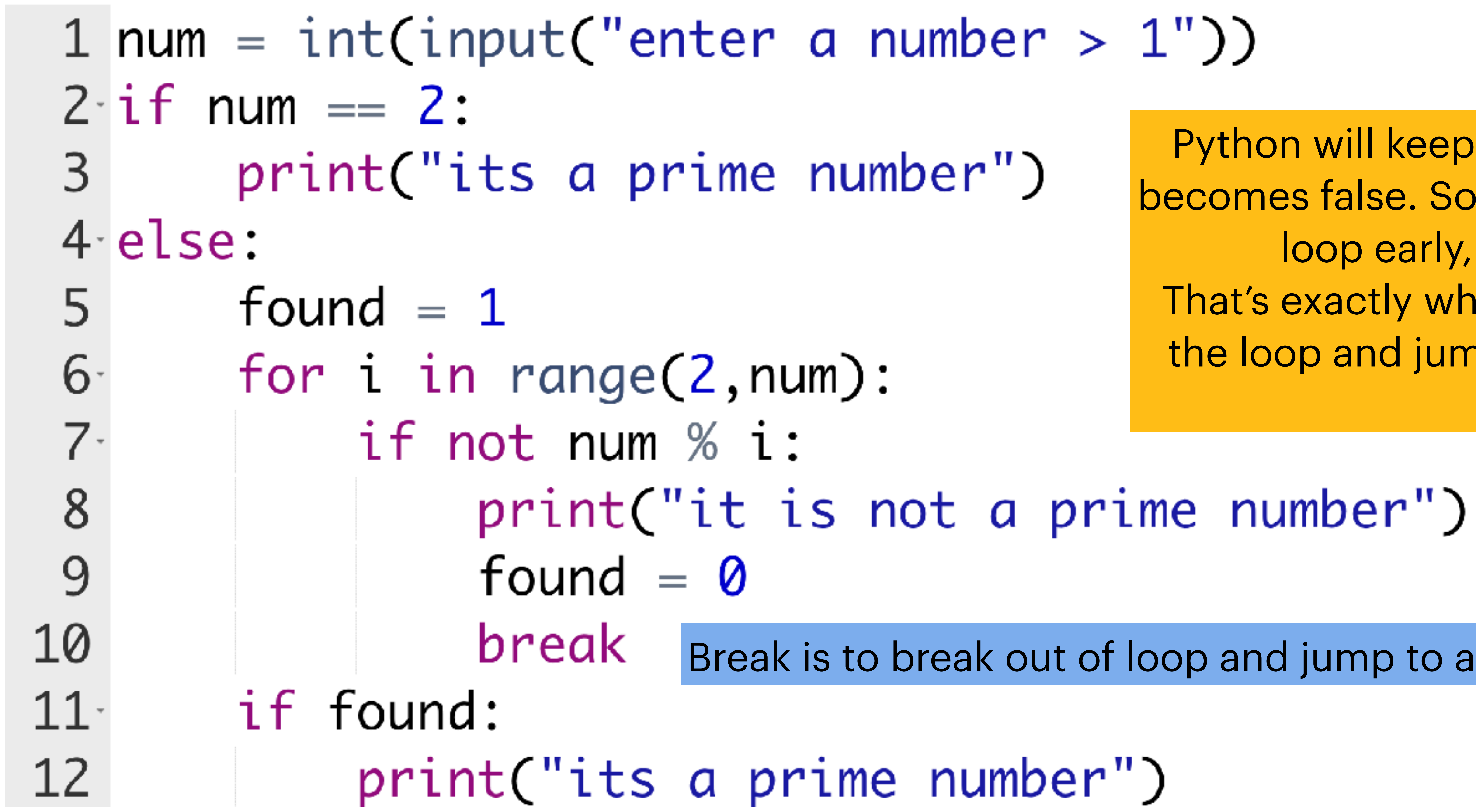


```python
num = int(input("enter a number > 1"))
if num == 2:
    print("its a prime number")
else:
    found = 1
    for i in range(2,num):
        if not num % i:
            print("it is not a prime number")
            found = 0
            break
    if found:
        print("its a prime number")
```

Break is to break out of loop and jump to after the loop statements

For loops

Test if a given number n is a prime number, n > 1



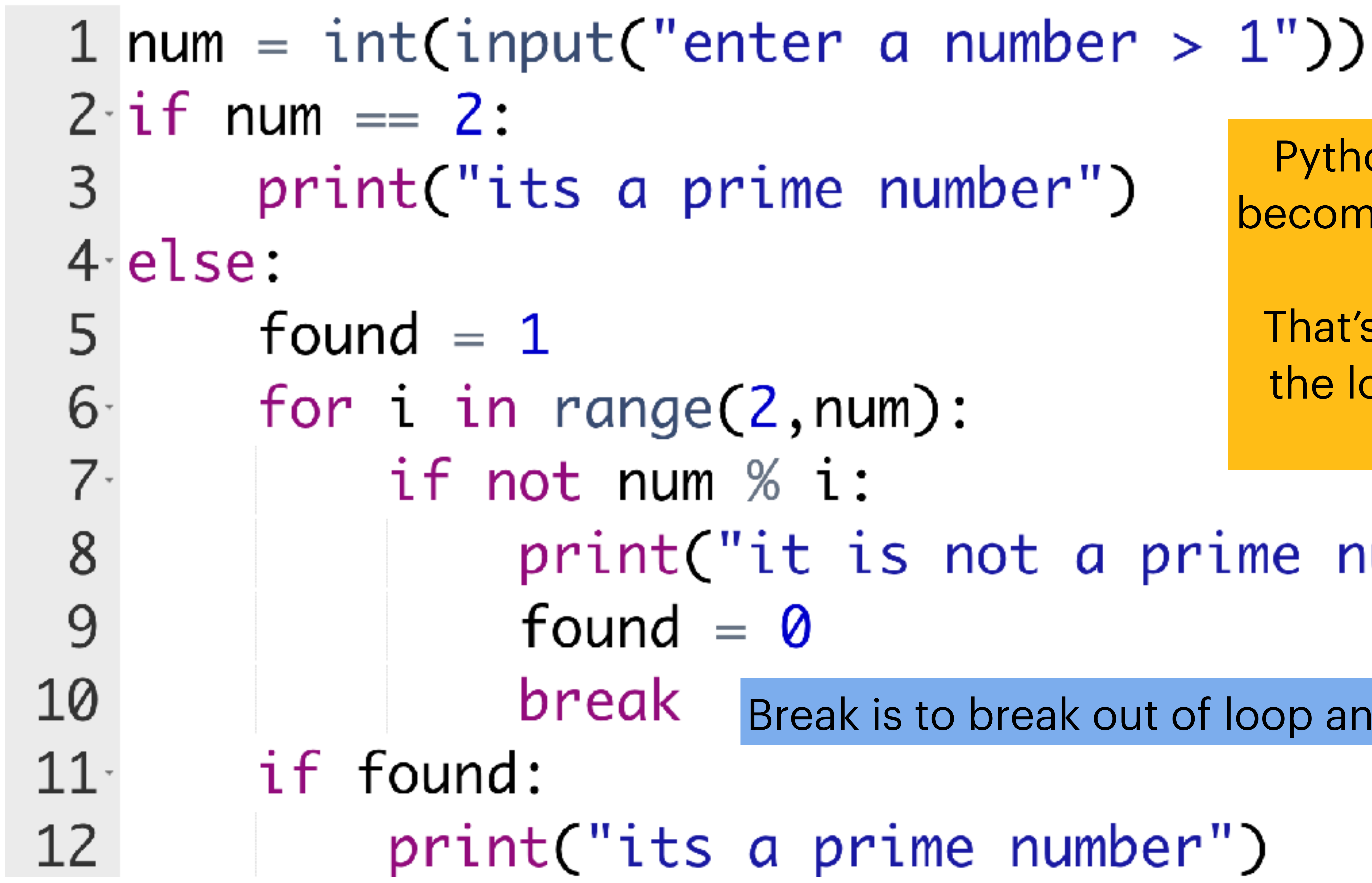
```python
num = int(input("enter a number > 1"))
if num == 2:
    print("its a prime number")
else:
    found = 1
    for i in range(2,num):
        if not num % i:
            print("it is not a prime number")
            found = 0
            break
    if found:
        print("its a prime number")
```

Python will keep executing the loop until the condition becomes false. Sometimes, however, you want to stop the loop early, even if the condition is still true.
That's exactly what `break` does — It immediately stops the loop and jumps to the first statement after the loop

Break is to break out of loop and jump to after the loop statements

For loops

Test if a given number n is a prime number, n > 1



```python
num = int(input("enter a number > 1"))
if num == 2:
    print("its a prime number")
else:
    found = 1
    for i in range(2,num):
        if not num % i:
            print("it is not a prime number")
            found = 0
            break
    if found:
        print("its a prime number")
```

Python will keep executing the loop until the condition becomes false. Sometimes, however, you want to stop the loop early, even if the condition is still true.
That's exactly what `break` does — It immediately stops the loop and jumps to the first statement after the loop

Break is to break out of loop and jump to after the loop statements

Loops and Branches can be nested to any degree