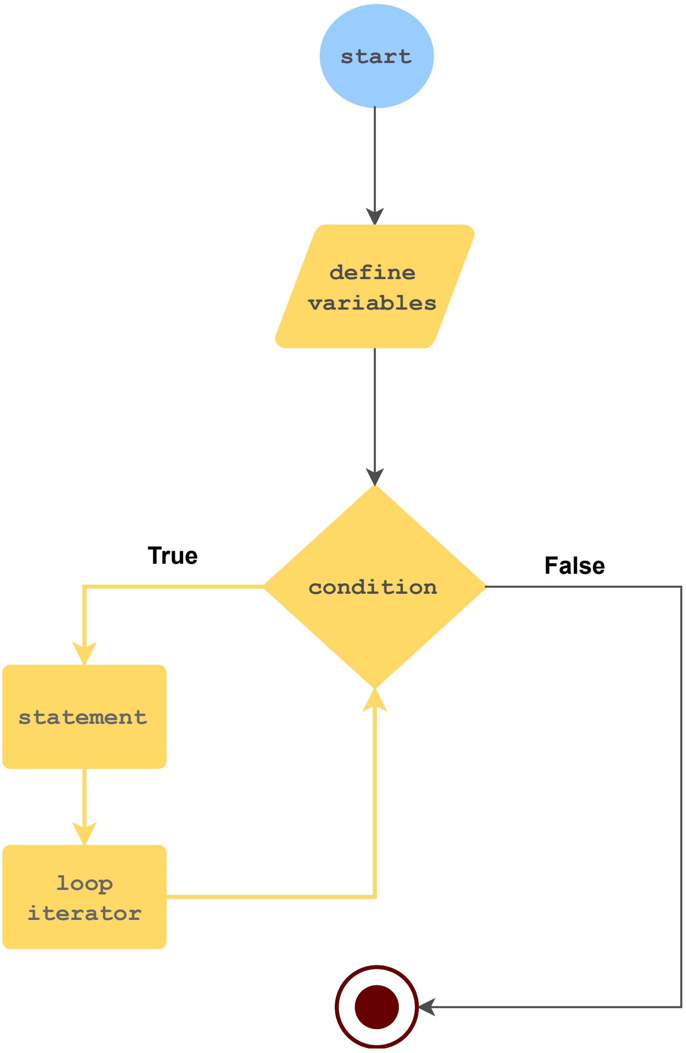
Definite Loop - For Loop

This lesson will teach about finite loop, i.e., for loop.

We'll cover the following

What Is a `for` Loop?

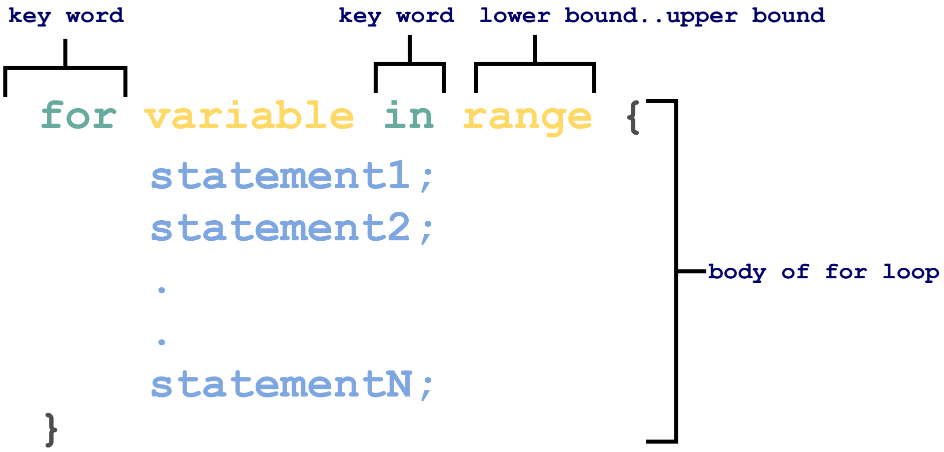
A `for` loop is a **definite loop**, meaning, the number of iterations is defined.



Syntax

The `for` loop is followed by a variable that iterates over a list of values.

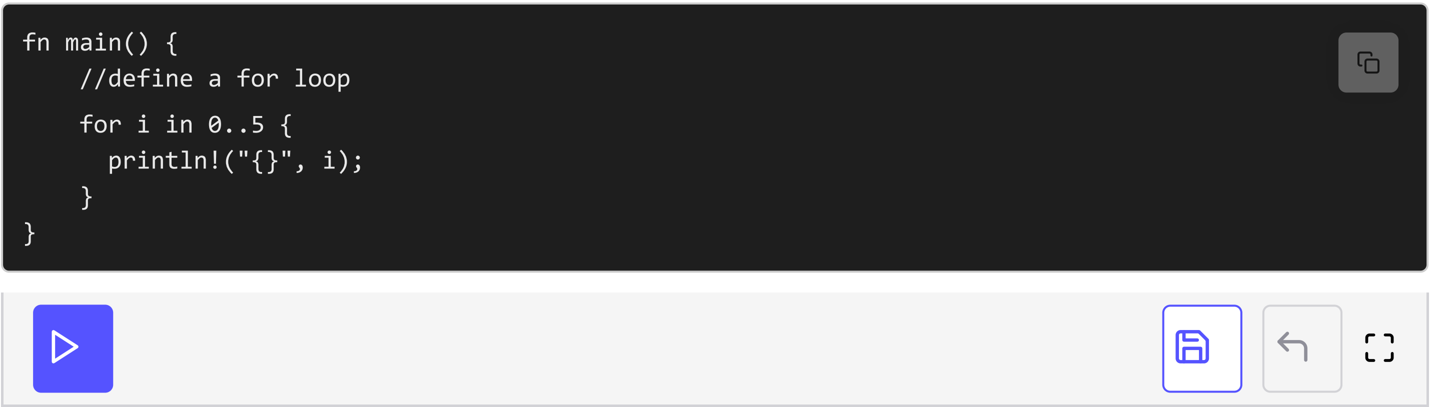
The general syntax is :



```
for variable in range {
      statement1;
      statement2;
      .
      .
      statementN;
}
```

Example

The following example uses a `for` loop that prints 5 numbers.



```
fn main() {
    //define a for loop
    for i in 0..5 {
      println!("{}", i);
    }
}
```

Explanation

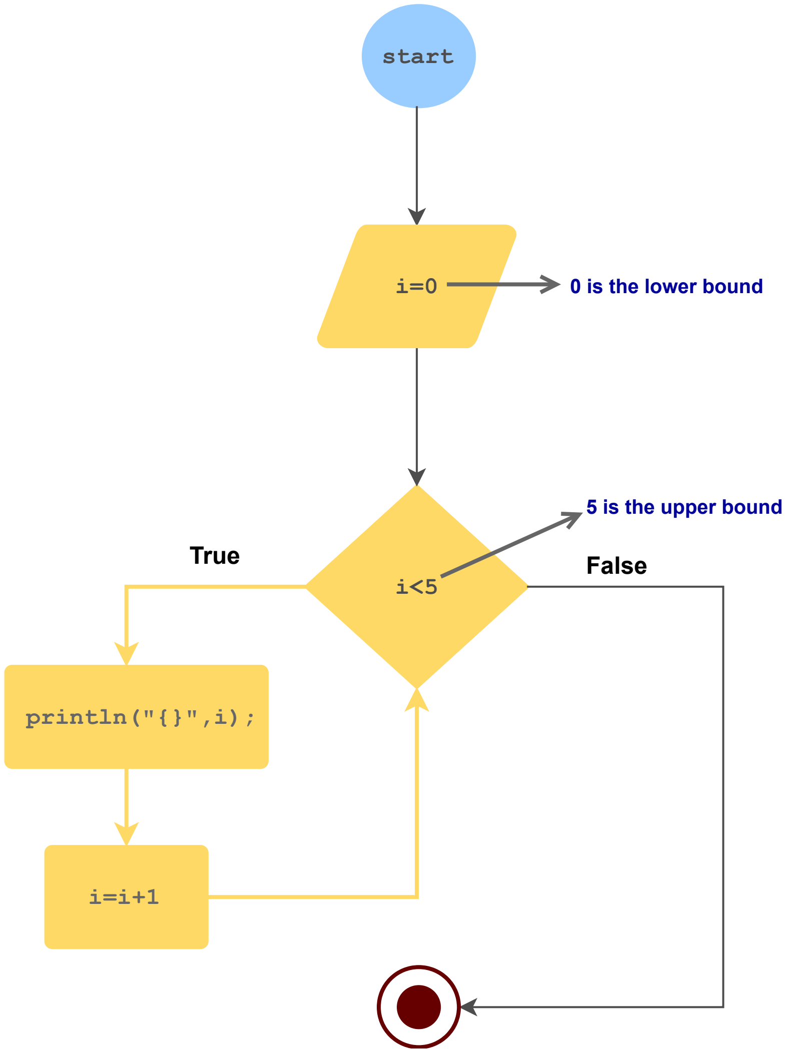
`for` loop definition

- On **line 3** a `for` loop is defined.
    - Variable `i` is an **iterator variable** that iterates over the range with the lower bound as 0 and the upper bound as 5. From here the body of the loop starts.

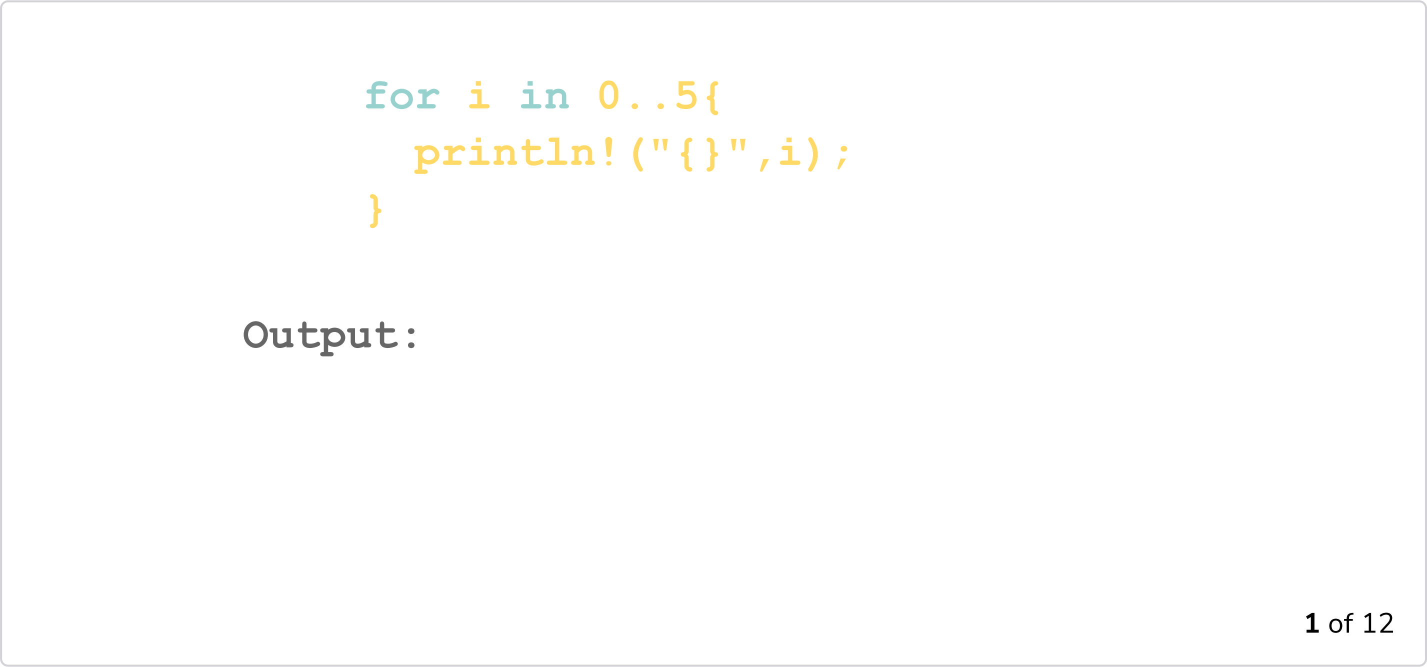
`for` loop body

- The body of the `for` loop is defined from **line 3 to line 5**
- In each iteration:
    - On **line 4**, the value of the variable `i` is printed.
    - The value of the variable `i` is incremented by 1.
- The iterator variable `i` traverses over the range until the upper bound is reached.

> **Note:** The lower bound is inclusive and the upper bound is exclusive in the range

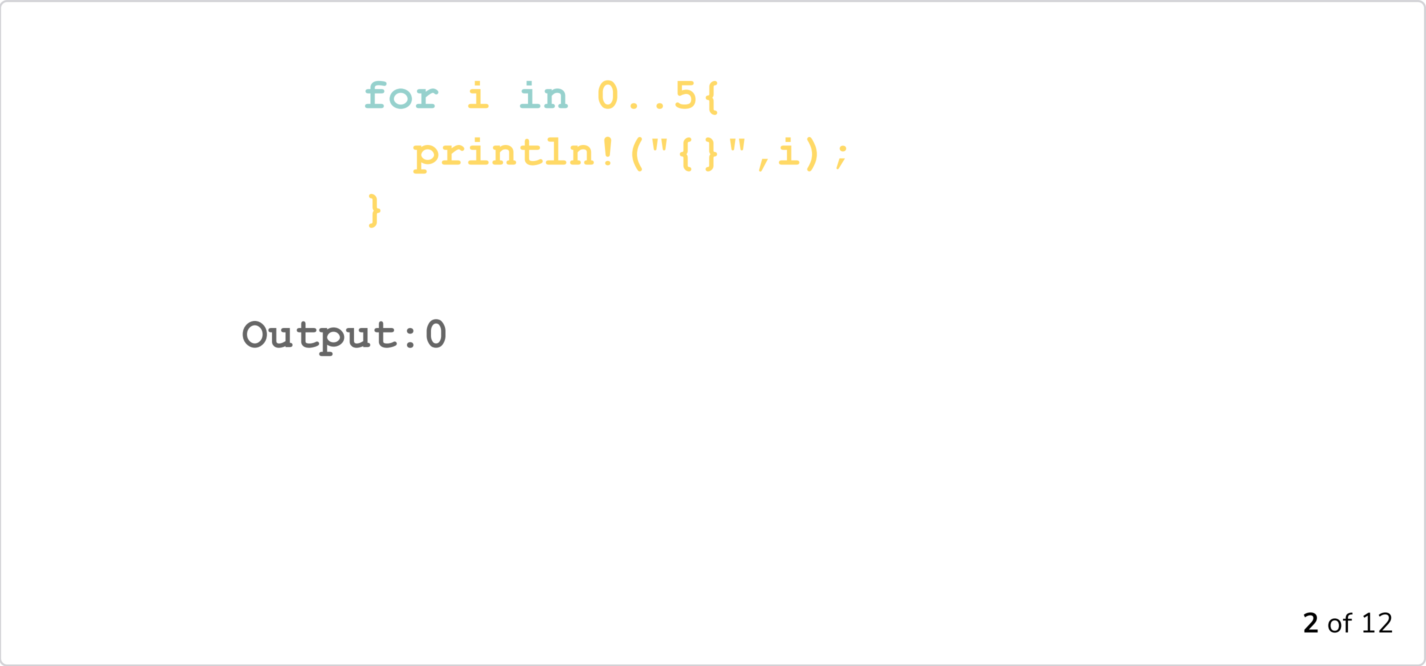


The following illustration explains this concept:



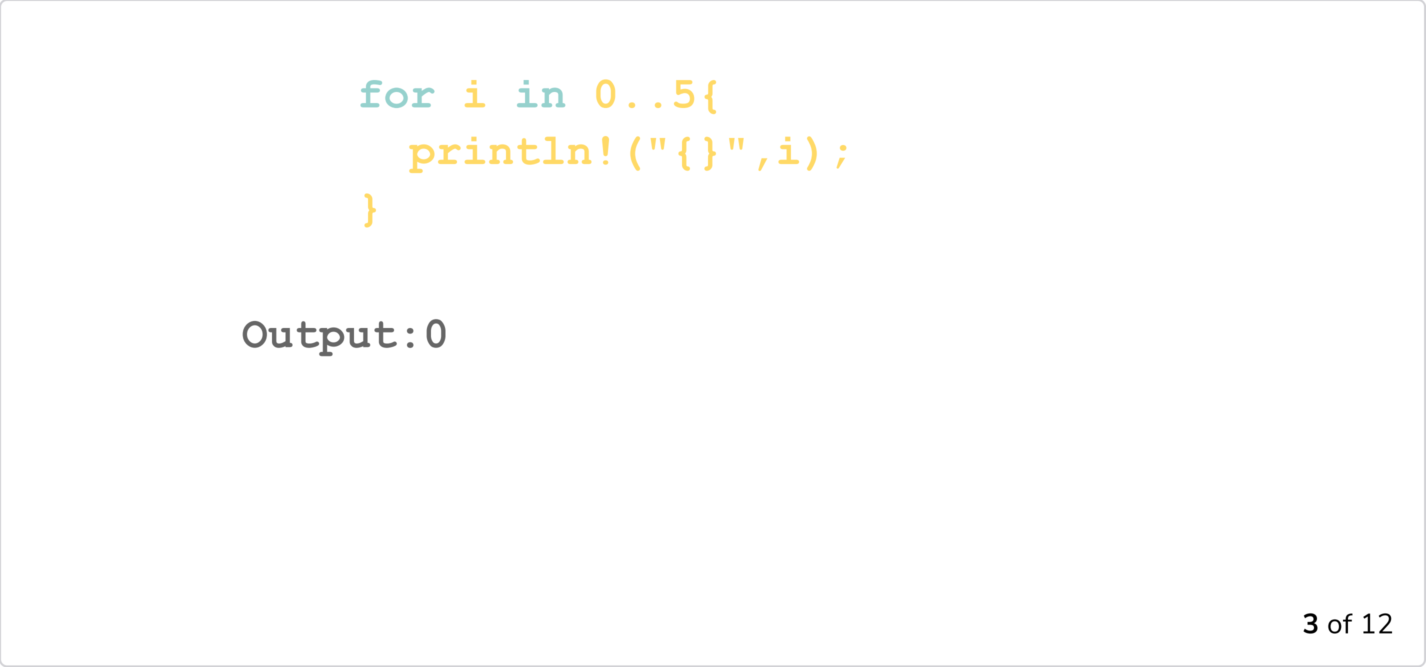
```
for i in 0..5{
  println!("{}",i);
}
```

Output:



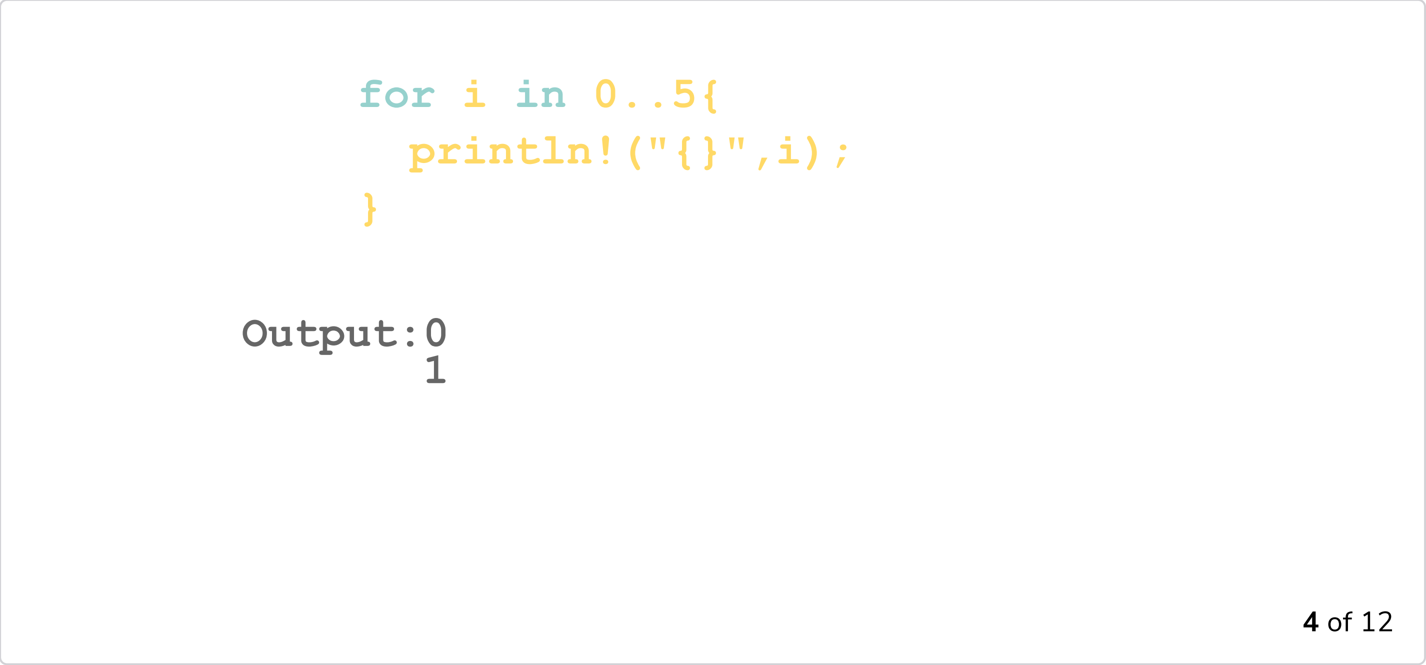
```
for i in 0..5{
  println!("{}",i);
}
```

```
Output:0
```



```
for i in 0..5{
  println!("{}",i);
}
```

```
Output:0
```



```
for i in 0..5{
  println!("{}",i);
}
```

```
Output:0
       1
```

```
for i in 0..5{
  println!("{}",i);
}
```

```
Output:0
       1
```

```
for i in 0..5{
  println!("{}",i);
}
```

```
Output:0
       1
       2
```

```
for i in 0..5{
  println!("{}",i);
}
```

```
Output:0
       1
       2
```

```
for i in 0..5{
  println!("{}",i);
}
```

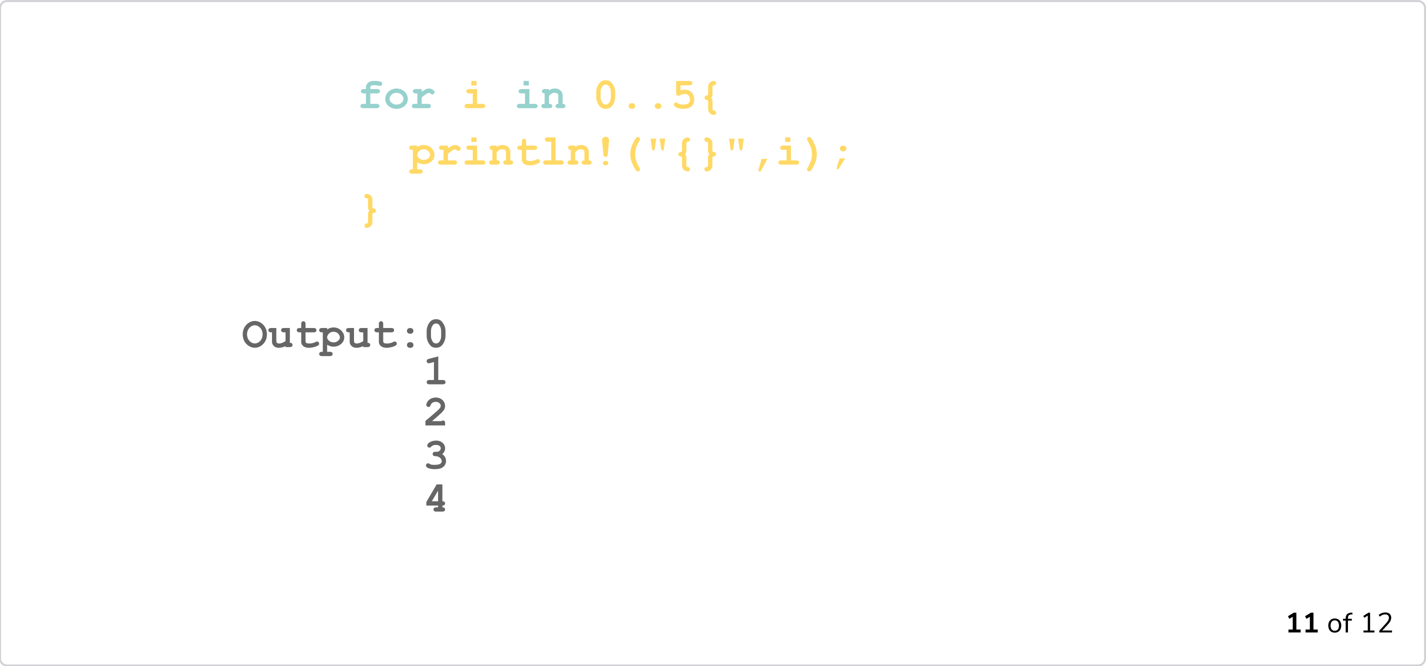
```
Output:0
       1
       2
       3
```

```
for i in 0..5{
  println!("{}",i);
}
```

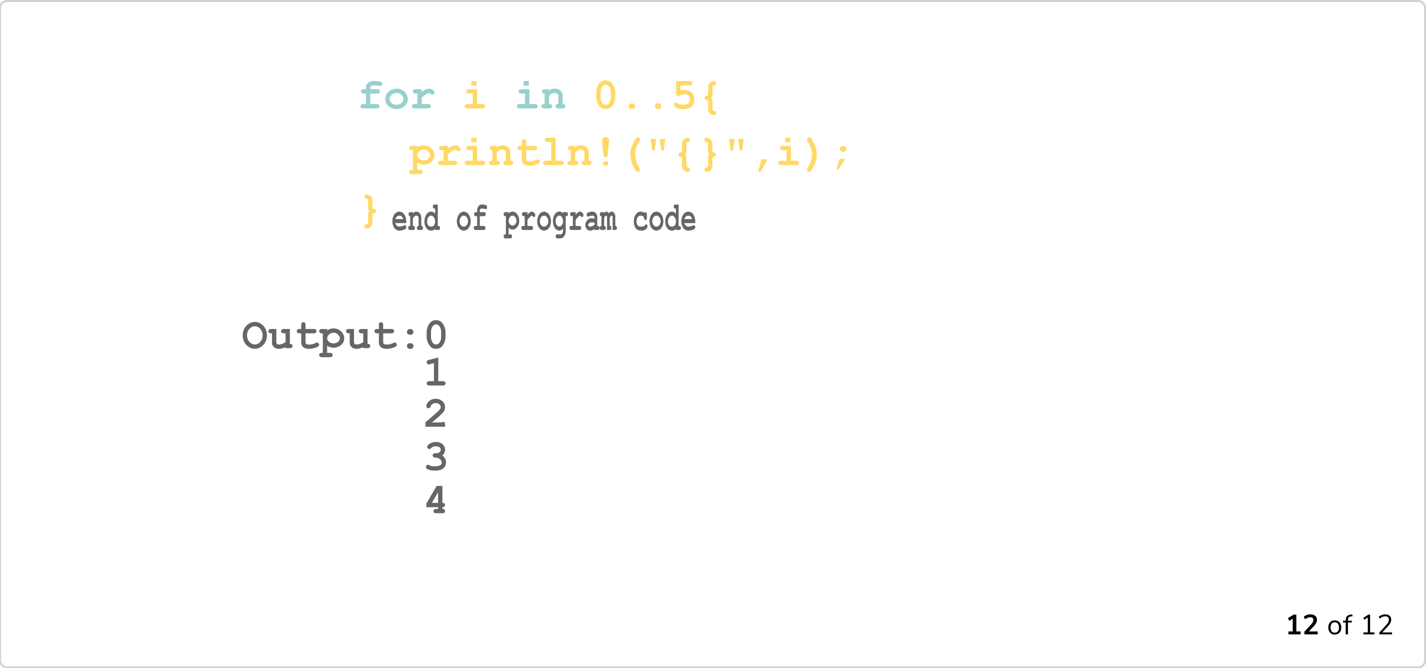
```
Output:0
       1
       2
       3
```

```
for i in 0..5{
  println!("{}",i);
}
```

```
Output:0
       1
       2
       3
       4
```



```
for i in 0..5{
  println!("{}",i);
}


Output:0
       1
       2
       3
       4
```



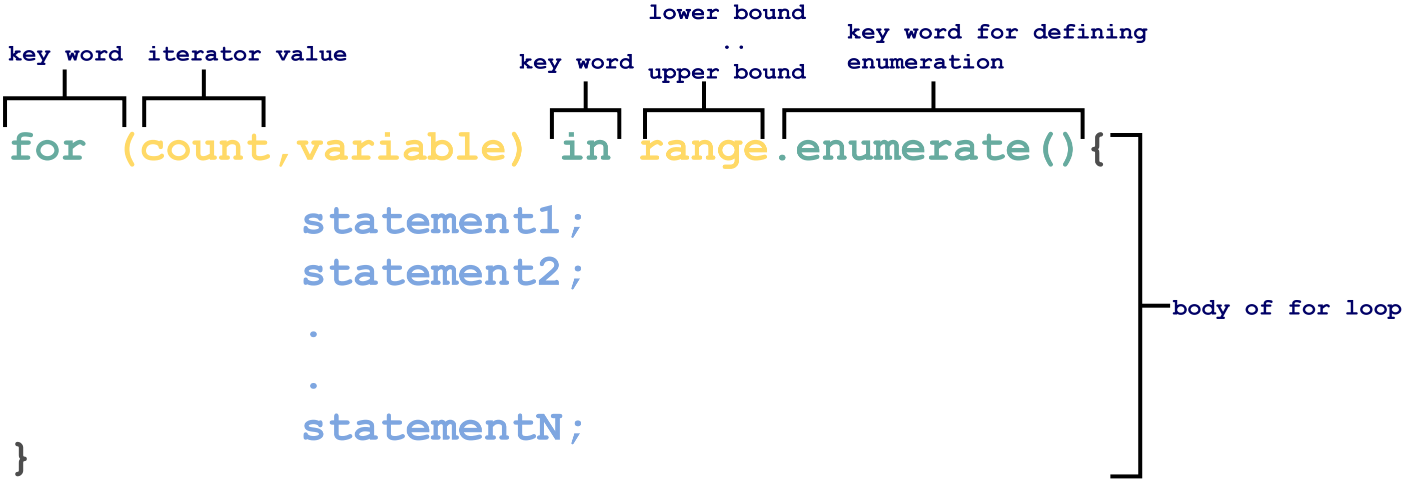
```
for i in 0..5{
  println!("{}",i);
} end of program code


Output:0
       1
       2
       3
       4
```

Enumerate

To count how many times the loop has already executed, use the `.enumerate()` function.

Syntax

The general syntax is :



```
for (count,variable) in range.enumerate(){
                statement1;
                statement2;
                .
                .
                statementN;
}
```

Example

The example below prints the frequency of iterations and the value of variable.

```
fn main() {
  for (count, variable) in (7..10).enumerate() {
      println!("count = {}, variable = {}", count, variable);
  }
}
```

Explanation

`enumerated for` loop definition

- On **line 2** an `enumerate` `for` **loop** is defined.
    - The variable `variable` iterates over the range with the lower bound as 7 and the upper bound as 10 and a variable `count` which shows how many times the loop is iterated. From here the body of the loop starts.

`enumerated for` loop body

- On **line 3**, the value of `count` and `variable` is printed and then incremented by 1.

Quiz

Test your understanding of `for` loop and enumerated `for` loop.

Quick Quiz on `for` Loop!

1

What is the output of the following code?

```
fn main() {
  for i in 0..5{
     if i % 4 == 0 {
        print!("{}", i);
     }
  }
}
```

2

What is the output of the following code?

```
fn main() {
  for (count, variable) in (7..10).enumerate() {
       if count * 2 == 4{
      println!("count = {}, variable = {}", count, variable);
       }
  }
}
```

Retake Quiz

Now that you have learned about for loop, let’s look at indefinite loops in the next lesson.