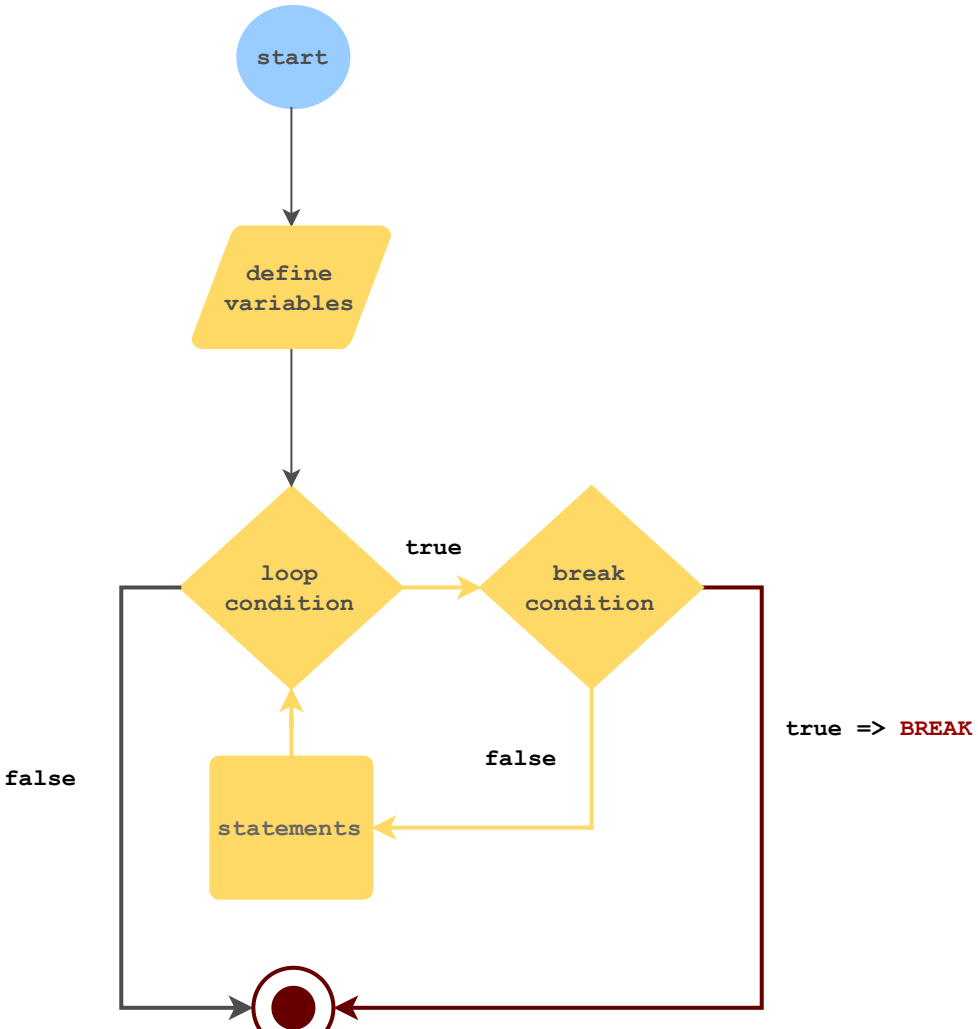
Break Statement

This lesson describes the break statement.

We'll cover the following

What Is a `break` Statement?

The `break` statement terminates the loop. It is generally placed inside a conditional statement so that the loop terminates if the associated condition is true.



The following illustration explains the concept of a break statement.

```
for i in 0..10{
    println!("{}",i);
    if i==2{
      break;
    }
  }
```

Output:

1 of 10

```
for i in 0..10{
    println!("{}",i);
    if i==2{
      break;
    }
  }
```

Output: 0

2 of 10

```
for i in 0..10{
    println!("{}",i);
    if i==2{
      break;
    }
  }
```

Output: 0

3 of 10

```
for i in 0..10{
    println!("{}",i);
    if i==2{
      break;
    }
  }
```

1

Output: 0

4 of 10

```
for i in 0..10{
    println!("{}",i);
    if i==2{
      break;
    }
  }
```

1

Output: 0
1

5 of 10

```
for i in 0..10{
    println!("{}",i);
    if i==2{False
      break;
    }
  }
```

1

Output: 0
1

6 of 10

```
for i in 0..10{
    println!("{}",i);
    if i==2{
      break;
    }
  }
```

2

Output: 0
1

7 of 10

```
for i in 0..10{
    println!("{}",i);
    if i==2{
      break;
    }
  }
```

2

Output: 0
1
2

8 of 10

```
for i in 0..10{
    println!("{}",i);
    if i==2{
      break;
    }
  }
```

2

Output: 0
1
2

```
for i in 0..10{
    println!("{}",i);
    if i==2{
      break;
    }
  }
```

```
Output: 0
        1
        2
```


Break statement is valid in case of `while`, `for` and `loop`.

Using With a `for` Loop

Below is an example of break expression, using a `for` loop.

- The range defined in the `for` loop is from 0 to 10.
- Within the `for` loop :
  - The value of `i` is printed
  - When the value of `i` is equal to 5, the loop terminates

```
fn main() {
  // define a for loop
  for i in 0..10 {
    println!("i:{}", i);
    if i == 5 {
      break;
    }
  }
}
```

Using With a `while` Loop

Below is an example of break expression, using a `while` loop

Below is an example of break expression, using a `while` loop.

- A mutable variable `i` is defined
- A boolean variable `found` is defined
- Within the `while` loop body :
  - The value of `i` is printed
  - When the value of `i` is equal to 5, the loop terminates

```
fn main() {
  let mut i = 1;
  let found = false;
  // define a while loop
  while !found {
    println!("i:{}", i);
    if i == 5 {
      break;
    }
    i = i + 1;
  }
}
```



Using With a `loop`

Below is an example of break expression, using a `loop`.

- A mutable variable `i` is defined
- Within the `loop` body:
  - The value of `i` is printed
  - When the value of `i` is equal to 4, the loop terminates

> The infinite loop is turned into a “manageable” loop.

```
fn main() {
  let mut i = 1;
  // define a loop
  loop{
    println!("i:{}", i);
    if i == 5 {
      break;
    }
    i = i + 1;
  }
}
```

Quiz

Test your understanding of how `break` statement works in Rust.

Quick Quiz on break Statement!

Q

How many times does the print statement in the loop run?

```
fn main() {
  for i in 0..10 {
    println!("i:{}", i);
    if i == 5 {
      break;
    }
  }
}
```

Retake Quiz

Now that you have learned about break statements, let's learn about continue statements in the next lesson.