Pass by Value

This lesson discusses how to pass arguments to the function by value.

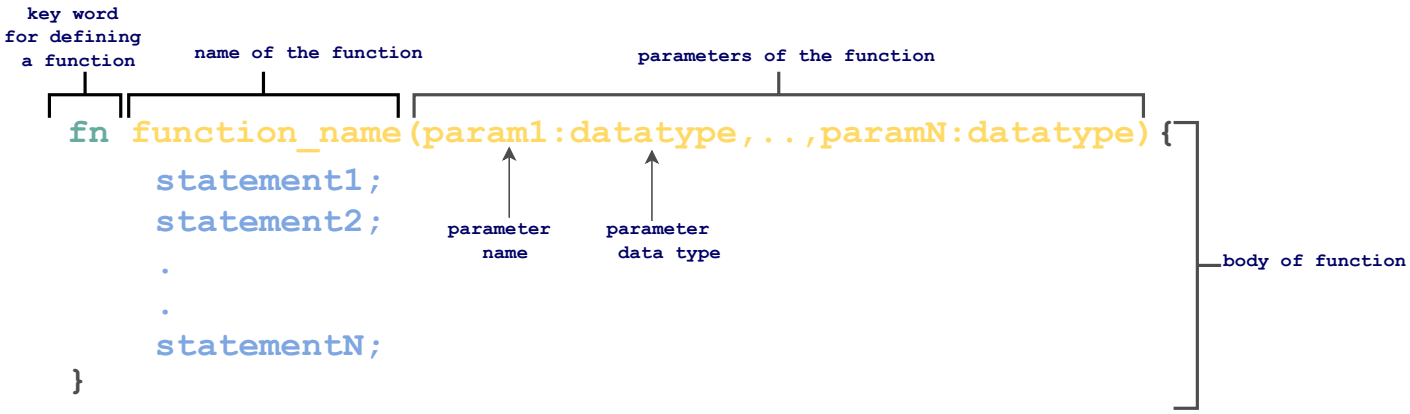
We'll cover the following

Arguments Pass by Value

The values from the calling function are copied to the parameters in the called function at the time the function is called. The called function can change the values of the parameter variables all it wants. This change will not be reflected in the variables passed as arguments in the calling function.

Syntax

The general syntax of passing arguments by value is:



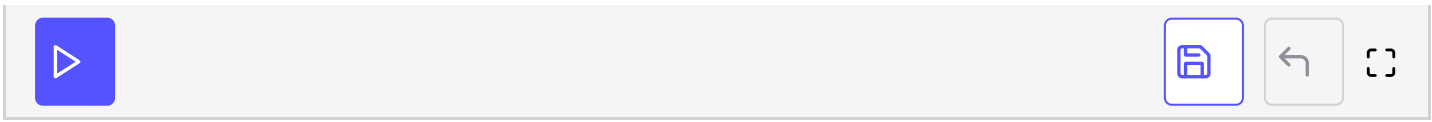
Defining a function with parameters passed by value

Example

Example

The following example makes a function `square()` that takes a number `n` as a parameter to the function and prints the square of the function within the function.

```
fn square(mut n:i32){
  n = n * n;
  println!("The value of n inside function : {}", n);
}
fn main() {
  let n = 4;
  println!("The value of n before function call : {}", n);
  println!("Invoke Function");
  square(n);
  println!("\nThe value of n after function call : {}", n);
}
```



Explanation

The above program is of two parts, the user defined function `square()` and the driver function `main()` where the function is being called.

User defined function

The function `square()` is defined from **line 1 to line 4**.

- On *line 2* `n` is multiplied with itself and the value is saved in `n`.
- The square of the argument thus calculated is displayed to the screen on *line 3*.

Driver function

The driver function `main()` is defined from **line 5 to line 11**.

- On *line 6*, a variable `n` is defined.
- On *line 9*, the function `square()` is invoked which takes `n` as an argument to the function.
- After the function call, the value of the `n` is printed.

> **Note:** The value of `n` **is not changed**

The following illustration shows how program execution proceeds in the above code:

```
fn square(mut n:i32){
  n=n*n;
  println!("The value of n inside function : {}",n);
}
fn main() {
  let n=4;
  println!("The value of n before function call : {}",n);
  println!("Invoke Function");
  square(n);
  println!("\nThe value of n after function call : {}",n);
}
```

Output:


```
fn square(mut n:i32){
  n=n*n;
  println!("The value of n inside function : {}",n);
}
fn main() {
  let n=4;
  println!("The value of n before function call : {}",n);
  println!("Invoke Function");
  square(n);
  println!("\nThe value of n after function call : {}",n);
}
```

Output:

```
fn square(mut n:i32){
  n=n*n;
  println!("The value of n inside function : {}",n);
}
fn main() {
  let n=4;
  println!("The value of n before function call : {}",n);
  println!("Invoke Function");
  square(n);
  println!("\nThe value of n after function call : {}",n);
}
```

```
Output:The value of n before function call : 4
```



```
fn square(mut n:i32){
  n=n*n;
  println!("The value of n inside function : {}",n);
}
fn main() {
  let n=4;
  println!("The value of n before function call : {}",n);
  println!("Invoke Function");
  square(n);
  println!("\nThe value of n after function call : {}",n);
}
```

```
Output:The value of n before function call : 4
       Invoke Function
```

```
fn square(mut n:i32){
  n=n*n;
  println!("The value of n inside function : {}",n);
}
fn main() {
  let n=4;
  println!("The value of n before function call : {}",n);
  println!("Invoke Function");
  square(n);
  println!("\nThe value of n after function call : {}",n);
}
```

```
Output:The value of n before function call : 4
       Invoke Function
```


n=4

```
fn square(mut n:i32){
  n=n*n;
  println!("The value of n inside function : {}",n);
}
fn main() {
  let n=4;
  println!("The value of n before function call : {}",n);
  println!("Invoke Function");
  square(n);
  println!("\nThe value of n after function call : {}",n);
}
```

```
Output:The value of n before function call : 4
       Invoke Function
```

n=4

```
fn square(mut n:i32){
  n=n*n;
  println!("The value of n inside function : {}",n);
}
fn main() {
  let n=4;
  println!("The value of n before function call : {}",n);
  println!("Invoke Function");
  square(n);
  println!("\nThe value of n after function call : {}",n);
}
```

```
Output:The value of n before function call : 4
       Invoke Function
```


n=4

```
fn square(mut n:i32){
  n=n*n;
  println!("The value of n inside function : {}",n);
}
fn main() {
  let n=4;
  println!("The value of n before function call : {}",n);
  println!("Invoke Function");
  square(n);
  println!("\nThe value of n after function call : {}",n);
}
```

```
Output:The value of n before function call : 4
       Invoke Function
       The value of n inside function:16
```

```
fn square(mut n:i32){
  n=n*n;
  println!("The value of n inside function : {}",n);
}end of function
fn main() {
  let n=4;
  println!("The value of n before function call : {}",n);
  println!("Invoke Function");
  square(n);
  println!("\nThe value of n after function call : {}",n);
}
```

```
Output:The value of n before function call : 4
       Invoke Function
       The value of n inside function:16
```



```
fn square(mut n:i32){
  n=n*n;
  println!("The value of n inside function : {}",n);
}
fn main() {
  let n=4;
  println!("The value of n before function call : {}",n);
  println!("Invoke Function");
  square(n);
  println!("\nThe value of n after function call : {}",n);
}
```

```
Output:The value of n before function call : 4
       Invoke Function
       The value of n inside function:16
       The value of n after function call : 4
```

```
fn square(mut n:i32){
  n=n*n;
  println!("The value of n inside function : {}",n);
}
fn main() {
  let n=4;
  println!("The value of n before function call : {}",n);
  println!("Invoke Function");
  square(n);
  println!("\nThe value of n after function call : {}",n);
} end of program code
```

```
Output:The value of n before function call : 4
       Invoke Function
       The value of n inside function:16
       The value of n after function call : 4
```


Quiz

Test your understanding of passing arguments by value as a function parameter.

Quick Quiz on Pass by Value!

Q

What is the output of the following code?

```
fn swap( x:i32, y:i32 ) {
  let temp = y;
  let y = x;
  let x = temp;
  println!("x : {}, y : {}", x , y);
}
fn main() {
  let x = 10;
  let y = 9;
  swap( x, y );
```

```
  println!("x : {}, y : {}", x, y);
}
```

Retake Quiz

Now that you have learned pass by value, explore pass by reference in the next lesson.