

# Function With Multiple Return Values

This lesson discusses how we can return multiple values from a function.

## We'll cover the following

- Returning Multiple Values
  - Syntax
- Example
  - Explanation
    - User defined function
    - Driver function

## Returning Multiple Values #

In system programming languages like C++ and C, it is only possible to return a single value or a pointer to an array from a function. However, Rust allows you to return multiple values using a **tuple**.

## Syntax #

The **function definition** for returning multiple values:

```
fn function_name(param1:datatype, ...,paramN:datatype) -> ( datatype1,datatype2,...,datatypeN )
```

The diagram illustrates the syntax of a Rust function definition for returning a tuple. It includes the following annotations:

- key word for defining a function**: points to `fn`
- name of the function**: points to `function_name`
- parameters of the function**: points to the parameter list `(param1:datatype, ...,paramN:datatype)`
- indicates that the function is returning a value**: points to the return type arrow `->`
- define the data type of the returning values**: points to the tuple `( datatype1,datatype2,...,datatypeN )`

Defining a function with returning a tuple

The way to **return tuple** from a function is to just write the tuple:

( value1,value2,..valueN )

values to return  
from a function

Defining a function with returning a tuple

## Example #

The following example makes a function `calculate_area_perimeter()` that takes a `x` and `y` ( length and width of a rectangle) as a parameter to the function and returns a tuple `(area, perimeter)`.

```
// driver function
fn main() {
    let length = 4;
    let width = 3;
    println!("Rectangle lenth:{}", length);
    println!("Rectangle width:{", width);
    let (area, perimeter) = calculate_area_perimeter(length, width);
    println!("Area: {}, Perimeter: {}", area, perimeter);
}

// calculate area and perimeter
fn calculate_area_perimeter(x: i32, y: i32) -> (i32, i32) {
    // calculate the area and perimeter of rectangle
    let area = x * y;
    let perimeter = 2 * (x + y);
    // return the area and perimeter of rectangle
    (area, perimeter)
}
```



## Explanation #

The above program comprises two functions, the user defined function `calculate_area_perimeter()` and the driver function `main()` where the function is being called.

### User defined function #

The function `calculate_area_perimeter()` is defined from **line 11 to line 17**.

- On *line 13*, the area of the rectangle is calculated by multiplying parameters `x` and `y` and the result is saved in `area`.
- On *line 14*, the perimeter of the rectangle is calculated by adding parameters `x` and `y` and then multiplying the result with 2 and then, the final result is saved in `perimeter`.
- On *line 16*, a tuple `(area, perimeter)` is returned.

#### Driver function #

The driver function `main()` is defined from **line 2 to line 9**

- On *line 3*, a variable `length` is initialized with the value `4`.
- On *line 4*, a variable `width` is initialized with the value `3`.
- On *line 5 and 6*, the value of `length` and `width` is displayed respectively.
- On *line 7*, the function `calculate_area_perimeter()` is invoked which takes `length` and `width` as an argument to the function and return value of the function is saved in a tuple.

---

Learn about passing arrays to a function in the next lesson.