C Programming

Functions

Functions are also known as 'Methods' in some Programming languages.

Functions are a way to reuse certain code segments. This is highly efficient and avoids having to re-write the same pieces of code throughout a program.

All programming languages have built-in functions and also allow the developer to create their own functions.

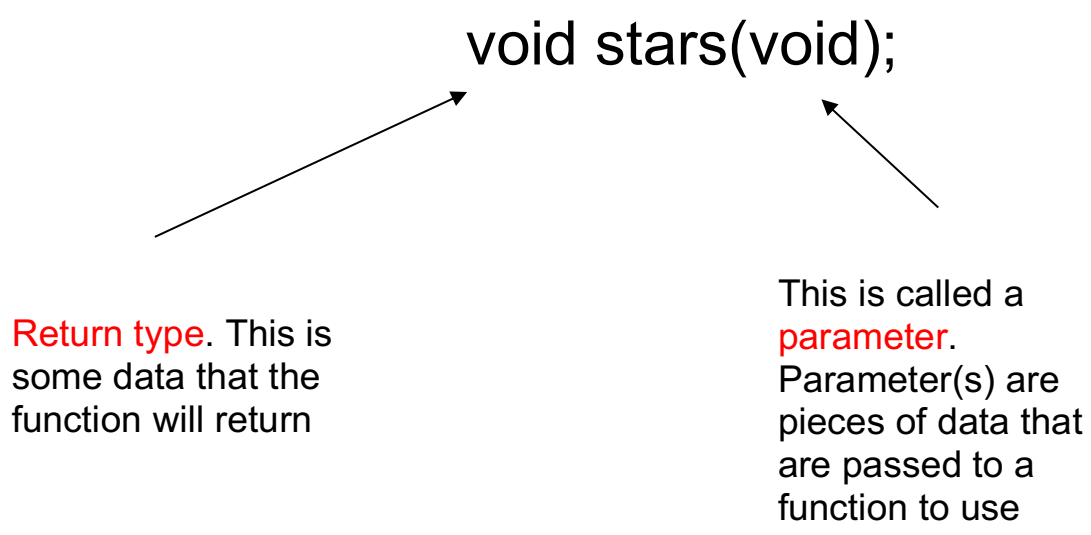
Functions have a name and a set of regular braces following. e.g.,

main()
sizeof()
printf()
scanf()

When you create your own function, the name of the function must follow the same rules for naming variables, i.e., cannot use a reserved word in C.

Every function that you create in a program in C, must have a **Function Signature** (aka Function Prototype) written before the main(). This is used to give the compiler all required info about the function before you use it in the program. For example, a function signature may look like the following:

```
void stars(void);
```



Let's take a look at our first function ..

```c
/*
Program that uses Functions
*/

#include <stdio.h>

//#define SIZE 5

// Function signature or Function prototype
void stars(void);

int main()
{
	printf("Before function call\n\n");

	// Execute our function stars()
	stars();

	printf("\n\nAfter function call");

	return 0;

} // end main()


// Function stars() used to display a set of asterix
//
void stars(void)
{
	printf("*****");

} // end stars()
```

Repl 12.1: https://replit.com/@michaelTUDublin/121-Simple-function

Parameters (aka Arguments)

A parameter is a piece of data that is passed to a function when it is called. This data can be any simple data type, e.g., an int, float, etc., and can also be a data structure, e.g., an array

There are no limits to the number of parameters you can pass to a function. You can also mix the different parameter types, e.g., passing an int and an array

Let's modify the above program so that it passes a single parameter …

```
/*
Program that uses Functions
*/
#include <stdio.h>

//#define SIZE 5

// Function signature or Function prototype
void stars(int);

int main()
{
   int no_stars = 0;


   printf("How many stars to display?\n\n");
   scanf("%d", & no_stars);

   // Execute our function stars()
   stars(no_stars);

   printf("\n\nAfter function call");

   return 0;

} // end main()

// Function stars() used to display a set of asterix
//
void stars(int num)
{
```

```
    int i;

    for(i = 0; i < num; i++)
    {
        printf("*");

    } // end for

} // end stars()
```

Repl 12.2: https://replit.com/@michaelTUDublin/122-Single-parameter

Let's modify the above code so that more than 1 parameter is used …

```
/*
Program that uses Functions
*/
#include <stdio.h>

//#define SIZE 5

// Function signature or Function prototype
void stars(int, char);

int main()
{
    int no_chars = 0;
    char my_char;


    printf("How many characters to display?\n\n");
    scanf("%d", & no_chars);

    // Clears the input buffer
    while(getchar() != '\n');

    printf("Which character to display?\n\n");
    scanf("%c", & my_char);
```

```
    // Execute our function stars()
    stars(no_chars, my_char);

    printf("\n\nAfter function call");

    return 0;
} // end main()


// Function stars() used to display a set of asterix
//
void stars(int num, char ch)
{
    int i;

    for(i = 0; i < num; i++)
    {
        printf("%c", ch);

    } // end for

} // end stars()
```

Repl 12.3: https://replit.com/@michaelTUDublin/123-Multiple-parameters