codecademy

Functions

Functions in C

A function is a block of reusable logic that may have a defined set of input and output.

Built-In Function in C

The C programming language comes with built-in standard library functions, such as:

- `printf()`
- `rand()`

```
#include <stdio.h>

int main() {
  // printf is a standard library function
  printf("Hello built-in functions!");
}
```

Calling Functions

In C, a function is called by stating the function name followed by parentheses. One or more argument values can be placed in the parentheses if the function requires any input values.

```
int incrementBy(int number1, int number2)
{
  return number1 + number2;
}

int main() {
  // The value of myNumber is retrieved by
  // calling the function incrementBy()
with
  // the arguments 5 and 2
  int myNumber = incrementBy(5, 2);
}
```

Storing A Return Value

A function `return` value, or function output, can be stored in a variable to be used for future calculations.

```
int incrementBy(int number1, int number2)
{
  return number1 + number2;
}

int main() {
  // myNumber will hold the return value
  // of increment by, which is 7
  int myNumber = incrementBy(5,2);
}
```

Function Signature

A user-defined function is defined using a function signature. This signature specifies the return type and the function name followed by parameters inside parentheses.

```
// A function signature includes the
// return type, function name, and
// parameter(s) in the parentheses
int incrementBy(int number1, number2) {
  return number1 + number2;
}
```

Return Type `void`

A function that returns no value must use the keyword `void` as the return type within the function signature.

```
// void is used since the function
// printNumnber does not return any value
void printNumber(int number) {
  printf("Your number is %d\n", number);
}
```

Function Return Value

A user-defined function can return a value with the `return` keyword followed by the value to be returned. The type of the returned value must match the return type specified in the function signature.

```
// the return keyword returns the
// value following the keyword
int getOne() {
  return 1;
}
```

Function Parameters

In C, a user-defined function can specify input using parameters. Parameters are comma-separated variable definitions within the function signature parentheses.

```
// number1 and number2 are paramters
// for the incrementBy function
void incrementBy(int number1, int number2)
{
  return number1 + number2;
}
```

Function Protypes

A function prototype specifies an interface with the required return type and parameter types to help the compiler ensure a function is called properly. A function prototype also helps separate the function declaration from its implementation.

```
// function prototpe
int increment(int);

// function implmentation
int increment(int number) {
  return number += 1;
}
```