C Programming Language: C Comments

- Comments can be used to explain code and to make it more readable. It can also be used to prevent execution when testing alternative code.
- Comments can be single-lined or multi-lined.

Single-line Comments

Single-line comments start with two forward slashes (//).

Remember: Any text between // and the end of the line is ignored by the compiler (will not be executed).

This example uses a single-line comment before a line of code:

```
// This is a comment
printf("Hello World!");
```

```
Hello World!
```

C Programming Language: C Comments

C Multi-line Comments
Multi-line comments start with /* and ends with */.

Remember: Any text between /* and */ will be ignored by the compiler

```
/* The code below will print the words Hello
World!
to the screen, and it is amazing */
printf("Hello World!");
```

```
Hello World!
```

Single or multi-line comments?

It is up to you which you want to use. Normally, we use `//` for short comments, and `/* */` for longer.

Good to know: Before version **C99** (released in 1999), you could only use multi-line comments in C.

C Programming Language: C Variables

Variables are containers for storing data values, like numbers and characters.

In C, there are different types of variables (defined with different keywords), for example:

- int - stores integers (whole numbers) without decimals, such as 123 or -123
- float - stores floating point numbers, with decimals, such as 19.99 or -19.99
- char - stores single characters, such as 'a' or 'B'. Characters are surrounded by single quotes

C Programming Language: C Variables

Declaring (Creating) Variables

To create a variable, specify the type and assign it a value:

```
type variableName = value;
```

Where type is one of C types (such as int), and variableName is the name of the variable (such as x or myName). The equal sign is used to assign a value to the variable.

Create a variable called myNum of type int and assign the value 15 to it:

```
int myNum = 15;
```

You can also declare a variable without assigning the value, and assign the value later:

```
// Declare a variable
int myNum;

// Assign a value to the variable
myNum = 15;
```

C Programming Language: C Variables

Output Variables

You learned from the output chapter that you can output values/print text with the printf() function.

In many other programming languages (like Python, Java, and C++), you would normally use a print function to display the value of a variable. However, this is not possible in C:

```
int myNum = 15;
printf(myNum);  // Nothing happens
```

To output variables in C, you must get familiar with something called "format specifiers"

C Programming Language: C Statements

Format Specifiers

- Format specifiers are used together with the printf() function to tell the compiler what type of data the variable is storing. It is basically a placeholder for the variable value.
- A format specifier starts with a percentage sign %, followed by a character.

For example, to output the value of an int variable, use the format specifier %d surrounded by double quotes (""), inside the printf() function:

```
int myNum = 15;
printf("%d", myNum);  // Outputs 15
```

C Programming Language: C Variables

To print other types, use %c for char and %f for float:

```
// Create variables
int myNum = 15;             // Integer (whole
number)
float myFloatNum = 5.99;    // Floating point
number
char myLetter = 'D';        // Character

// Print variables
printf("%d\n", myNum);
printf("%f\n", myFloatNum);
printf("%c\n", myLetter);
```

To combine both text and a variable, separate them with a comma inside the printf() function:

```
int myNum = 15;
printf("My favorite number is: %d", myNum);
```

C Programming Language: C Variables

To print different types in a single printf() function, you can use the following:

```
int myNum = 15;
char myLetter = 'D';
printf("My number is %d and my letter is
%c", myNum, myLetter);
```

Print Values Without Variables

You can also just print a value without storing it in a variable, as long as you use the correct format specifier:

```
printf("My favorite number is: %d", 15);
printf("My favorite letter is: %c", 'D');
```

C Programming Language: C Variables

Change Variable Values

If you assign a new value to an existing variable, it will overwrite the previous value:

```
int myNum = 15;  // myNum is 15
myNum = 10;  // Now myNum is 10
```

You can also assign the value of one variable to another:

```
int myNum = 15;

int myOtherNum = 23;

// Assign the value of myOtherNum (23) to
myNum
myNum = myOtherNum;

// myNum is now 23, instead of 15
printf("%d", myNum);
```

C Programming Language: C Variables

Add Variables Together

To add a variable to another variable, you can use the + operator:

```
int x = 5;
int y = 6;
int sum = x + y;
printf("%d", sum);
```

```
11
```