Functions in C++

Dr Ian Cornelius

Hello

Hello (1)

Learning Outcomes

1. Understand how to use functions in C++
2. Demonstrate the ability to use functions in C++

Functions

Functions (1)

- **Recap**:
  - Functions are a block of reusable code that is used to perform a single action
  - They provide an aspect of modularity to your code and ensure a high-degree of code reuse
  - Help you break a complex problem into smaller portions of code
    - making the script easier to understand

Functions (2)

Creating a Function

- Creating a function in C++ is different to Python
- Functions in C++ begin with a return type, followed by the function name and a set of brackets (`()`)
- The `returnType` will be the data type of the data being returned from the function
  - it can also be set to `void` if no data is being returned

```
returnType functionName () {
  ...
}
```

Functions (3)

Using a Function

- Functions can be called using their function name, followed by a set of brackets
  - this is often known as the **function caller**

```
void hello() {
  std::cout << "Hello 5062CEM!" << std::endl;
}
```

```
hello() -> Hello 5062CEM!
```

Functions (4)

Returning Values from a Function i

- Functions can also return data from inside it using the return statement
- Useful if you have performed some operations inside a function and need to use the output in the main body of your code

```
std::string hello() {
  return "Hello 5062CEM!";
}
```

```
hello() -> Hello 5062CEM!
```

Functions (5)

Returning Values from a Function ii

- The returned value from a function can be stored in a variable

```
#include <iostream>
std::string hello() {
  return "Hello 5062CEM!";
}
int main() {
  std::string result = hello();
  std::cout << "result -> " << result << std::endl;
  return 0;
}
```

```
result -> Hello 5062CEM!
```

Parameters and Arguments

Parameters and Arguments (1)

- **Recap**:
  - Data can be passed through to a function, and these are known as either *parameters* or *arguments*
  - Parameter and argument can be used for the same thing
    - simply it is data that is passed into a function
  - But they do have a slightly different meaning:
    - **parameter** is the variable listed inside the brackets in the function definition
    - **argument** is the value that is sent to the function
- Parameters are specified after the declaration of the function name and inside the set of round brackets (`()`)
  - you are able to add as many parameters as you want, separating them with a comma (`,`)

```
returnType functionName (parameter1, parameter2, ...) {
  ...
}
```

Parameters and Arguments (2)

Example: Parameters and Arguments

- Create a function called `hello`
  - accepts a single parameter, `name` of type string
  - returns a `string`
- The Function is called with a string provided inside the brackets
  - i.e. `hello("Ian Cornelius")`
- The function returns a greeting welcoming the person to the module

```
std::string hello(std::string name) {
  return "Hello " + name + ", and welcome to 5062CEM!";
}
```

```
hello("Ian Cornelius") -> Hello Ian, and welcome to 5062CEM!
hello("Terry Richards") -> Hello Terry, and welcome to 5062CEM!
hello("Daniel Goldsmith") -> Hello Daniel, and welcome to 5062CEM!
```

Parameters and Arguments (3)

- When calling a function, it must be called with the correct number of arguments
  - if you have a function with three arguments, then you have to call the function with three arguments

```
std::string hello(std::string name, std::string code) {
  return "Hello " + name + " and welcome to " + code + "!";
}
```

```
hello("Ian Cornelius", "5062CEM") -> Hello Ian and welcome to 5062CEM!
hello("Terry Richards", "5069CEM") -> Hello Terry and welcome to 5069CEM!
hello("Daniel Goldsmith") -> error: too few arguments to function ‘std::string hello(std::string, std::string)’
```

Parameters and Arguments (4)

Default Values

- A function can be called without an argument if a default value has been assigned to the parameter
  - the default value will only be evaluated once

```
std::string hello(std::string name, std::string code="5062CEM") {
  return "Hello " + name + " and welcome to " + code + "!";
}
```

```
hello("Ian", "5062CEM") -> Hello Ian and welcome to 5062CEM!
hello("Terry", "5069CEM") -> Hello Terry and welcome to 5069CEM!
hello("Daniel") -> Hello Daniel and welcome to 5062CEM!
```

Goodbye

Goodbye (1)

Questions and Support

- Questions? Post them on the **Community Page** on Aula
- Additional Support? Visit the Module Support Page
- Contact Details:
  - Dr Ian Cornelius, ab6459@coventry.ac.uk