

User Functions

- Functions are similar to **C**
- Need to specify the return type, identifier and arguments followed by the block of statements
- All parameters to user functions are passed by reference
- Unlike **C**, nested functions are allowed in **VEX**
- **void** data type functions which will not return anything

Structure of a Function

The diagram illustrates the components of a C++ function signature: `int my_function (int a, b; string c){`. Brackets above the code identify the parts: `int` is the Data type; `my_function` is the Identifier; `(int a, b; string c)` are the Arguments, with a note "(passed by reference &)" below them. A large bracket on the right groups the opening curly brace, the body `//do something`, and the closing curly brace as the Block. The body also contains `return 8`.

```
int my_function ( int a, b; string c){
    //do something
    return 8
}
```

Data type Identifier Arguments
(passed by reference &)

Block

Pre-processor Directives

- Separate program that runs before the compiler and processes directives
- It's not VEX or C, it's simply a **text processor**
- It **strips comments, reads include files** and **expands macros**
- The **include** keyword copies the content of the included file into the program
- The **define** directive creates a macro that will be replaced before compiling
- The **pragma** directives are for creating the user interface