C++ Basics

1. C++ Syntax

C++ Program Structure: A typical C++ program consists of preprocessor directives, the `main` function, and statements.
cpp

```cpp
#include <iostream>

int main() {
    std::cout << "Hello, World!" << std::endl;
    return 0;
}
```

-
- **Headers**: `#include <iostream>` includes the input-output stream header for handling console input and output.
- **`main` Function**: The entry point of every C++ program. The program execution begins here.
- **Statements**: Each statement ends with a semicolon (`;`).

2. Variables

- **Variables** store data that can be used and manipulated in a program.

Declaration and Initialization:
cpp

```cpp
int age = 25;              // Integer variable
double price = 19.99;      // Floating-point variable
char grade = 'A';          // Character variable
std::string name = "John"; // String variable (requires
<string> header)
bool isStudent = true;     // Boolean variable
```

-
- **Naming Conventions**: Variable names should start with a letter or underscore, followed by letters, digits, or underscores.

3. Data Types

- **Primitive Data Types**:
  - `int`: Integer type (e.g., `int x = 10;`)
  - `float` and `double`: Floating-point types for real numbers (e.g., `float pi = 3.14f;`)
  - `char`: Single character (e.g., `char letter = 'A';`)
  - `bool`: Boolean type (e.g., `bool flag = true;`)
- **Modifiers**:
  - `short`, `long`, `unsigned`, and `signed` can modify data types.
  - Example: `unsigned int` ensures only positive integers.

4. Loops

- **Loops** allow repeated execution of a block of code.
- **Types of Loops**:

`for` Loop: Used when the number of iterations is known.
cpp

```
for (int i = 0; i < 5; i++) {
    std::cout << i << std::endl;
}
```

  -

`while` Loop: Used when the number of iterations is not known.
cpp

```
int i = 0;
while (i < 5) {
    std::cout << i << std::endl;
    i++;
}
```

  -

`do-while` Loop: Similar to the `while` loop but guarantees at least one execution.
cpp

```
int i = 0;
do {
```

```
    std::cout << i << std::endl;
    i++;
} while (i < 5);
```

-

5. Functions

- **Functions** are blocks of code that perform a specific task and can be reused.

Function Declaration and Definition:
cpp

```
// Declaration
int add(int a, int b);

// Definition
int add(int a, int b) {
    return a + b;
}
```

-

Calling a Function:
cpp

```
int result = add(3, 4);
std::cout << "Result: " << result << std::endl;
```

-
- **Return Types**: A function can return a value using the `return` keyword. The type of the returned value must match the function's return type.
- **Parameters**: Functions can accept input parameters to perform tasks based on the provided data.

Summary

- **Syntax**: The rules and structure for writing valid C++ code, with a focus on preprocessor directives, the `main` function, and semicolon-terminated statements.
- **Variables**: Containers for storing data, defined by data types, and initialized with values.

- **Data Types**: Primitive types include `int`, `float`, `double`, `char`, and `bool`. Modifiers like `short`, `long`, and `unsigned` alter the range and behavior of these types.
- **Loops**: Structures for repeating code execution, including `for`, `while`, and `do-while` loops.
- **Functions**: Blocks of reusable code that can perform tasks and return values, with the ability to accept parameters.