Function Parameter Styles in C++

Contents

When to Use Each Form

Parameter Style	When to Use	Return Needed for Changes to Persist?
`int num, float x`	For **small built-in types** (`int`, `float`, `char`) – copying is cheap.	Yes, if caller needs result
`const Type& obj`	For **large objects**, or classes – avoids copy and prevents changes.	Not applicable (read-only)
`Type obj`	If the function **needs a copy** (e.g., to modify independently). Changes made will **not** affect the original unless a modified copy is **returned**.	**Yes**
`Type& obj`	If the function needs to **modify the caller's object** directly. No need to return unless chaining or feedback is desired.	**No**

Note: When using `Type obj` (pass-by-value), changes made inside the function do not affect the original object unless the modified value is explicitly `returned` and stored in the caller.

1. Examples

1. Small Built-in Types

```
#include <iostream>
void printSum(int a, float b) {
    std::cout << "Sum: " << (a + b) << std::endl;
}
```

2. const Type& obj — No Modification

```
#include <iostream>
#include <string>
class Book {
public:
    std::string title;
    Book(std::string t) : title(t) {}
};

void displayBook(const Book& b) {
    std::cout << "Book: " << b.title << std::endl;
}
```

3. Type obj — Make a Copy

```
#include <iostream>
#include <string>

class Book {
public:
    std::string title;
    Book(std::string t) : title(t) {}
};

void backupBook(Book b) {
    b.title = "Backup of " + b.title;
    std::cout << "Inside function: " << b.title << std::endl;
}
```

4. Type& obj — Modify Caller

```
#include <iostream>
#include <string>

class Book {
public:
    std::string title;
    Book(std::string t) : title(t) {}
};

void renameBook(Book& b) {
    b.title = "Updated: " + b.title;
}
```

5. Main Function to Demonstrate All

```
int main() {
    printSum(3, 4.5f);

    Book original("C++ Primer");

    displayBook(original);

    backupBook(original);
    std::cout << "After backupBook: " << original.title << std::endl;

    renameBook(original);
    std::cout << "After renameBook: " << original.title << std::endl;

    return 0;
}
```