جامعة المستقبل
AL MUSTAQBAL UNIVERSITY

قسم الامن السيبراني
Department of Cyber Security

Subject:

Structured Programming

Class:

1st stage

Lecturer:

Dr. Abdulkadhem A. Abdulkadhem

Lecture: (5)

Passing Parameters to Function by Value and by Reference

Passing Parameters by Value and by Reference in C++

1. Passing Parameters by Value

When passing parameters **by value**, a copy of the variable is sent to the function. Any modification inside the function does **not** affect the original variable.

Example 1: Passing an Integer by Value

```
#include <iostream>
using namespace std;

void square(int num) { // 'num' is a copy of the original value
    num = num * num;
    cout << "Value inside function: " << num << endl;
}

int main() {
    int value = 5;
    square(value);
    cout << "Original value after function call: " << value << endl;
    // Remains unchanged
    return 0;
}
```

Output:

```
Value inside function: 25
Original value after function call: 5
```

Note: The `value` in `main()` remains unchanged because modifications occur only on the copy.

Example 2: Passing Two Variables by Value to Calculate Sum

```
#include <iostream>
using namespace std;

int add(int a, int b) { // Copies of 'a' and 'b' are passed
    return a + b;
}

int main() {
    int x = 10, y = 20;
    cout << "Sum: " << add(x, y) << endl;
    return 0;
}
```

Note: The original values of `x` and `y` remain unchanged after calling the function.

2. Passing Parameters by Reference

When passing parameters **by reference**, the function receives the actual memory address of the variable. Any modification inside the function **will affect** the original variable.

Example 1: Passing an Integer by Reference

```
#include <iostream>
using namespace std;

void squareByReference(int &num) { // 'num' refers to the original variable
    num = num * num;
}

int main() {
    int value = 5;
    squareByReference(value);
    cout << "Original value after function call: " << value << endl; // Value
is modified
    return 0;
}
```

Output:

```
Original value after function call: 25
```

Note: The value in `main()` is modified because `num` is a reference to `value`.

Example 2: Swapping Two Variables Using Pass by Reference

```
#include <iostream>
using namespace std;

void swapValues(int &a, int &b) {
    int temp = a;
    a = b;
    b = temp;
}

int main() {
    int x = 10, y = 20;
    cout << "Before swapping: x = " << x << ", y = " << y << endl;
    swapValues(x, y);
    cout << "After swapping: x = " << x << ", y = " << y << endl;
    return 0;
}
```

Output:

```
Before swapping: x = 10, y = 20
After swapping: x = 20, y = 10
```

Note: The values of `x` and `y` are swapped because they were passed by reference.

3. Comparison: Pass by Value vs. Pass by Reference

Feature	Pass by Value	Pass by Reference
Modifies Original Value?	□ No	□ Yes
Performance	Slower for large data structures (copies are created)	Faster (no copy, direct memory access)
Data Safety	Safer (no unintended modifications)	Risky (original data can be modified)

4. When to Use Each?

- **Use pass by value** when the function **does not** need to modify the original variable (e.g., calculations that don't change input values).
- **Use pass by reference** when the function **must** modify the original variable (e.g., swapping values, updating data).