C++: if / if-else / else-if (elif) — Complete Guide

1. Simple if

Execute a block only when a condition is true.

```
if (x > 0) {
    std::cout << "positive";
}
```

2. if-else

Choose between two alternatives depending on the condition.

```
if (x % 2 == 0) {
    std::cout << "even";
} else {
    std::cout << "odd";
}
```

3. else-if ladder (elif)

Test multiple mutually exclusive conditions in sequence.

```
if (score >= 90) {
    grade = 'A';
} else if (score >= 75) {
    grade = 'B';
} else if (score >= 60) {
    grade = 'C';
} else {
    grade = 'F';
}
```

4. Nested if

One if inside another. Use sparingly; prefer early returns or helper functions.

```
if (user) {
    if (user->isActive()) {
        user->sendNotification();
    }
}
```

5. if with initializer (C++17)

Declare and initialize a variable inside the if statement; its scope is the if/else blocks.

```
if (auto it = m.find(key); it != m.end()) {
    process(it->second);
} else {
    // not found
}
```

6. Guard clauses / early return

Reduce nesting by returning early when preconditions fail.

```
void process(int *ptr) {
    if (!ptr) return; // guard
    // main logic
}
```

7. Assignment vs comparison pitfall

Be careful: using '=' assigns. Prefer comparisons; use parentheses if assigning then comparing.

```
int a;
if (a = getValue()) { // assigns, then tests a
    // often a bug unless intentional
}
// safer when assigning for test:
if (int n = parse(); n > 0) {
    use(n);
}
```

8. Boolean expressions and short-circuiting with if

Combine conditions with && and ||. Order conditions so cheap/safe checks come first.

```
if (ptr && ptr->value > 10) {
    // safe: ptr tested before ptr->value
}
if (fastCheck() || expensiveOperation()) {
    // expensiveOperation only called if needed
}
```

9. Best practices and style

- Prefer small functions and guard clauses to reduce nesting. - Name complex conditions with descriptive booleans. - Keep each branch short and focused. - Avoid deeply nested ifs; refactor into functions.

10. Examples — combined

Realistic examples using many patterns together.

```
std::optional<int> findIndex(const std::vector<int>& v, int x) {
    for (size_t i = 0; i < v.size(); ++i) {
        if (v[i] == x) return (int)i;
    }
    return std::nullopt;
}

if (auto idx = findIndex(arr, target); idx) {
    std::cout << "found at " << *idx;
} else {
    std::cout << "not found";
}
```