Programming in C++

Control Statements and Decision Making in C++

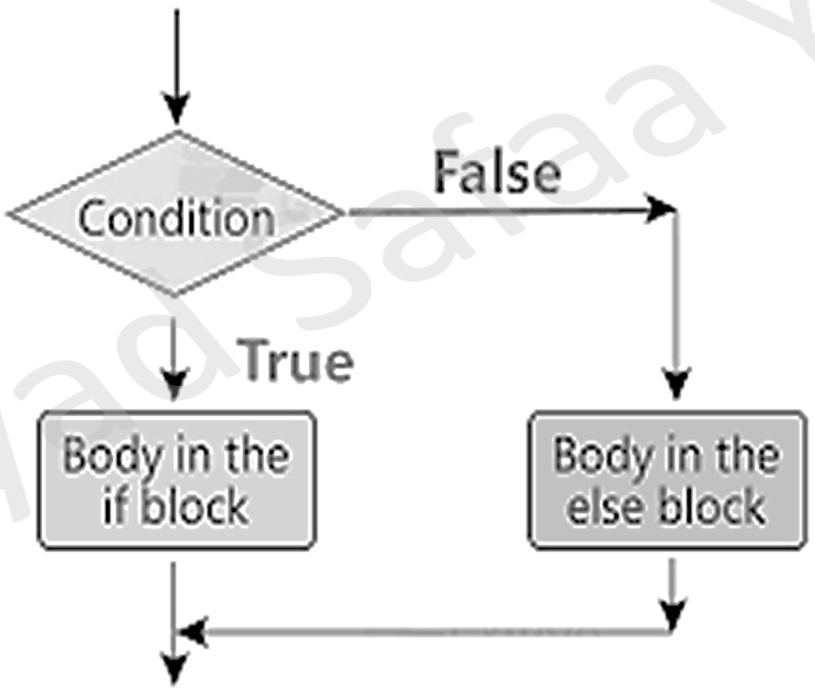
Dr. Zeyad Safaa Younus Saffawi

Control Statements and Decision Making

- **Control statements** allow the program to **make decisions** and **choose different paths** based on certain conditions.
- They **control** the **flow of execution** in a program.
- **Decision-making structures** are fundamental tools that allow a program to execute specific code based on **specific conditions**.
- **The core idea is simple**: **IF** a **condition** is **true**, **execute** this **set of instructions**, and **ELSE** execute a **different set** of instructions.

General Form of a Decision-Making Structure

- The typical **decision-making structure** found in most programming languages, including C++, consists of the following parts:



The if Statement

- The **if** statement is used to **check a condition**.
- An **if** statement consists of a **Boolean expression** followed by one or more statements.
- If the condition is **true**, a specific block of code will be executed.
- **Syntax:**

```
if ( condition (boolean_expression) )
  {
      statement(s) will be executed if the boolean expression is true
  }
```

- If you have only one statement after if, you don't need **{ }** braces, but if you have more than one statement, you must use **{ }** to group them.

if Statement

Example:

```
#include <iostream>
using namespace std;
int main ()
    {
     int a;
     cin >> a;
     if  ( a < 20 )              // check the boolean condition
         cout << "a is less than 20;" << endl;
     cout << "value of a is : " << a << endl;
     return 0;
     }
```

Multiple **if Statement**

Multiple if Statement:

Example: Compare two numbers.

```
#include <iostream>
using namespace std;
int main()
   {
       int a, b;
        cout << "Enter two numbers: ";
        cin >> a >> b;
        if (a = = b)
           cout << " a is equal b\n";
        if (a ! = b)
           cout << " a is not equal b\n";
        if (a > b)
           cout << "a is greater than b\n";
        if (a < b)
           cout << "a is smaller than b\n";
       return 0;
    }
```

if - else Statement

- The **if-else** statement will execute one group of statements if the expression is **true**, or another group of statements if the expression is **false**.
- An **if** statement can be followed by an optional **else** statement, which executes when the Boolean expression is **false**.
- **Syntax:**

```
if (condition)
    {
        statement(s) will be executed if the condition is true
    }
else
    {
        statement(s) will be executed if the condition is false
    }
```

- notes:
  - **If** the condition is true → **if statement(s)** will be executed.
  - **If** the condition is false → **else** statement(s) will be executed.

if - else Statement

- Example: Check Positive or Negative Number

```
#include <iostream>
using namespace std;
int main()
    {
      int num;
      cout << "Enter a number: ";
      cin >> num;
      if (num >= 0)
          cout << "The number is positive";
       else
            cout << "The number is negative";
    return 0;
    }
```

Nested if Statements

- A **nested if statement** means having one if inside another if.

- This structure allows your program to make more detailed decisions — one condition must be true before checking another.

- In simple words: The inner if will only run if the outer if condition is true.

Nested if Statements

- **Syntax:**

```
if (condition1)
    {
        if (condition2)
            {
             statement(s) will be executed if both condition1 and condition2 are true
             }
         else
            {
             statement(s) will be executed if condition1 is true but condition2 is false
            }
    }
 else
    {
        if (condition3)
            {
             statement(s) will be executed if condition1 is false but condition3 is true
            }
         else
            {
             statement(s) will be executed if both condition1 and condition3 are false
             }
      }
```

Nested if Statements

- **Example: Finding the Largest of Three Numbers**

```
#include <iostream>
using namespace std;
int main()
{
   int x, y, z;
   cout << "Enter three numbers: ";
   cin >> x >> y >> z;
   if (x > y)
       {
        if (x > z)
         cout << "x is the largest number:"<<x;
         else
              cout << "z is the largest number:"<< z;
       }
    else {
              if (y > z)
                 cout << "y is the largest number:"<< y;
              else
                   cout << "z is the largest number:"<< z;
            }
   return 0;
  }
```

Nested if – else Statement

- An **if** statement can be followed by an optional **else if – else** statement, which is very useful to test various conditions using a single **if-else if-else** statement.
- **Syntax :**

```
if  (boolean_expression 1)
{
   statement(s) will be executed when the boolean expression 1 is true
 }
else
      if ( boolean_expression 2)
            {
              statement(s) will be executed when the boolean expression 2 is true
            }
      else
            {
              statement(s) will be executed when none of the above conditions is true.
             }
```

Nested if – else Statement

- **Example — Find the largest number among three**

```
#include <iostream>
using namespace std;
int main ( )
        {
         int n1, n2, n3;
         cout << "Enter three numbers: ";
         cin >> n1 >> n2 >> n3;
         if (n1 > n2  &&  n1 > n3)
            cout << "n1 is the largest number: " << n1;
         else
              if (n2 > n1 && n2 > n3)
                   cout << "n2 is the largest number: " << n2;
              else
                   cout << "n3 is the largest number: " << n3;
         return 0;
        }
```

Switch Case Statement

- The **switch** statement is a **control statement** used to select **one action from multiple choices**, depending on the value of an expression or variable.
- **Syntax:**

```
switch (expression)
    {
     case constant1:
                          statement(s);
          break;              // optional  (break: Stops the execution and exits the switch)
     case constant2:
                          statements;
          break; // optional
           ...
      default:                                  // optional
                          statements;           //  (execute if no case matches)
     }
```

- **It is an alternative to writing many if...else if...else statements.**

Switch Case Statement

- **Example:**

```
#include <iostream>
using namespace std;
int main()
        {
          int n;
          cin >> n;
          switch (n)
                {
                  case 1:
                        cout << "One";
                  break;
                  case 2:
                        cout << "Two";
                  break;
                  default:
                        cout << "Invalid number!";
                  }
        return 0;
        }
```

Switch Case Statement

Example:

```
#include <iostream>
 using namespace std;
 int main() {
    char grade;
    cout << "Enter your grade: ";                // (A, B, C, D, E, F)
    cin >> grade;
    switch (grade) {
       case 'A':  cout << "Excellent";
          break;
       case 'B':   cout << "Very Good";
          break;
       case 'C':    cout << "Good";
          break;
       case 'D':    cout << "Average";
          break;
      case 'E':   cout << "Pass";
          break;
       case 'F':    cout << "Fail";
          break;
       default:    cout << "Invalid grade";
    }
    return 0;
 }
```

Nested Switch Case Statement

- **Example**

```
#include <iostream>
using namespace std;
int main ()
{
int a = 100;
int b = 200;
switch(a) {
        case 100:
                cout << "This is part of outer switch" << endl;
                cout << "Exact value of a is : " << a << endl;
         switch(b) {
                 case 200:
                 cout << "This is part of inner switch" << endl;
                 cout << "Exact value of b is : " << b << endl;
                }
        }
return 0;
}
```

Nested Switch Case Statement

```
#include <iostream>
using namespace std;
int main ()
{ int a,b;
Cout<< " enter the numbers :";
cin>>a, b;
switch(a) {
                case 100:
                          cout << "This is part of outer switch" << endl;
                         cout << "Exact value of a is : " << a << endl;
                 switch(b) {
                                  case 200:
                                  cout << "This is part of inner switch" << endl;
                                   cout << "Exact value of b is : " << b << endl;
                                  }
                 }
return 0;
}
```

Ternary ? Operator (Conditional Operator)

- The **ternary operator** is a **quick alternative** to the **if…else statement**.
- It allows assigning a value to a variable based on a condition in **one line**.
- **Syntax:**

```
variable = (condition) ? result1 : result2;
```

If the condition is true → result1 is executed / assigned.

If the condition is false → result2 is executed / assigned.

Equivalent if…else statement:

```
if (condition)
    variable = result1;
else
    variable = result2;
```

Ternary ? Operator

- **Examples:**

```
#include <iostream>
using namespace std;
int main()
{
    int a = 5,  b;
    b = a > 1 ? 10 : 20;        // a>1 is true → b=10
    cout << b;
    return 0;
}
```

References

- Gaddis, T. (2014). *Starting out with C++: From control structures through objects* (8th ed.). Pearson.
- Soulié, J. (2007, April 24). C++ language tutorial. cplusplus.com.
- Tutorials Point. (n.d.). *Learn C++ programming language*. Tutorials Point.