第4課：流程控制語句 - 條件分支

第4課：流程控制語句 - 條件分支

📚 課程目標

- 掌握if、else條件語句的使用
- 理解switch-case多分支選擇結構
- 學會使用break控制流程跳轉
- 掌握條件表達式的嵌套使用
- 理解條件語句的執行流程

🎯 本課關鍵字

- `if` - 條件判斷
- `else` - 否則分支
- `switch` - 多路分支
- `case` - 分支標籤
- `default` - 默認分支
- `break` - 跳出當前結構

📝 範例1：基本if語句

```
/*
* 範例4-1：基本if語句
* if語句用於根據條件決定是否執行某段代碼
*/
#include <iostream>
int main() {
std::cout << "=== 基本if語句演示 ===" << std::endl;
int age = 18;
// 最基本的if語句
if (age >= 18) {
std::cout << "你已經成年了！" << std::endl;
}
std::cout << "程序繼續執行..." << std::endl;
// if語句可以不用大括號，但只控制緊接著的一條語句
int score = 85;
if (score >= 60)
```

```
std::cout << "恭喜，你及格了！" << std::endl;
// 注意：沒有大括號時，只控制下一條語句
if (score >= 60)
std::cout << "這是if控制的語句" << std::endl;
std::cout << "這是不受if控制的語句，總是會執行" << std::endl; // 注意縮進不代表控制
// 推薦總是使用大括號，即使只有一條語句
int temperature = 30;
if (temperature > 25) {
std::cout << "今天天氣炎熱，建議減少外出" << std::endl;
}
// 比較運算符在條件中的使用
int a = 10, b = 20;
if (a < b) {
std::cout << a << " 小於 " << b << std::endl;
}
// 邏輯運算符在條件中的使用
int hour = 14; // 24小時制
if (hour >= 9 && hour <= 17) {
std::cout << "現在是工作時間" << std::endl;
}
return 0;
}
```

📝 範例2：if-else語句

```
/*
* 範例4-2：if-else語句
* if-else用於在兩個選項中選擇一個執行
*/
#include <iostream>
int main() {
std::cout << "=== if-else語句演示 ===" << std::endl;
// 1. 基本的if-else
int age = 16;
if (age >= 18) {
std::cout << "你可以觀看所有級別的電影" << std::endl;
} else {
std::cout << "你只能觀看輔導級以下的電影" << std::endl;
```

```
}
// 2. 多個條件判斷
int score = 85;
if (score >= 90) {
std::cout << "成績優秀！" << std::endl;
} else if (score >= 80) {
std::cout << "成績良好" << std::endl;
} else if (score >= 70) {
std::cout << "成績中等" << std::endl;
} else if (score >= 60) {
std::cout << "成績及格" << std::endl;
} else {
std::cout << "成績不及格，需要補考" << std::endl;
}
// 3. 條件表達式可以包含運算
int num = 15;
if (num % 2 == 0) {
std::cout << num << " 是偶數" << std::endl;
} else {
std::cout << num << " 是奇數" << std::endl;
}
// 4. 嵌套if語句
int year = 2024;
bool isLeapYear = false;
if (year % 4 == 0) {
if (year % 100 == 0) {
if (year % 400 == 0) {
isLeapYear = true;
} else {
isLeapYear = false;
}
} else {
isLeapYear = true;
}
} else {
isLeapYear = false;
}
if (isLeapYear) {
std::cout << year << " 年是閏年" << std::endl;
} else {
```

```
}
// 5. 簡化嵌套if（使用邏輯運算符）
bool isLeapYear2 = (year % 4 == 0 && year % 100 != 0) || (year % 400 == 0);
if (isLeapYear2) {
std::cout << "簡化判斷：" << year << " 年是閏年" << std::endl;
} else {
std::cout << "簡化判斷：" << year << " 年不是閏年" << std::endl;
}
// 6. if-else語句的返回值可以賦值給變數
int x = 10, y = 20;
int maxValue;
if (x > y) {
maxValue = x;
} else {
maxValue = y;
}
std::cout << x << " 和 " << y << " 中較大的是：" << maxValue << std::endl;
// 也可以使用條件運算符簡化
int minValue = (x < y) ? x : y;
std::cout << x << " 和 " << y << " 中較小的是：" << minValue << std::endl;
return 0;
}
```

📝 範例3：switch-case語句

```
/*
* 範例4-3：switch-case語句
* 用於多路分支選擇，比多個if-else更清晰
*/
#include <iostream>
int main() {
std::cout << "=== switch-case語句演示 ===" << std::endl;
// 1. 基本的switch-case
int dayOfWeek = 3;
std::cout << "今天是星期";
switch (dayOfWeek) {
case 1:
std::cout << "一" << std::endl;
```

```
break;
case 2:
std::cout << "二" << std::endl;
break;
case 3:
std::cout << "三" << std::endl;
break;
case 4:
std::cout << "四" << std::endl;
break;
case 5:
std::cout << "五" << std::endl;
break;
case 6:
std::cout << "六" << std::endl;
break;
case 7:
std::cout << "日" << std::endl;
break;
default:
std::cout << "？ (無效的輸入，請輸入1-7)" << std::endl;
break;
}
// 2. case的多值處理（多個case執行相同代碼）
char grade = 'B';
std::cout << "你的成績等級 " << grade << " 對應的評語是：";
switch (grade) {
case 'A':
case 'a':
std::cout << "優秀！" << std::endl;
break;
case 'B':
case 'b':
std::cout << "良好" << std::endl;
break;
case 'C':
case 'c':
std::cout << "中等" << std::endl;
break;
case 'D':
```

```
std::cout << "及格" << std::endl;
break;
case 'F':
case 'f':
std::cout << "不及格" << std::endl;
break;
default:
std::cout << "無效的成績等級" << std::endl;
break;
}
// 3. 計算器示例
char operatorChar = '+';
double num1 = 10.5, num2 = 5.5, result = 0.0;
switch (operatorChar) {
case '+':
result = num1 + num2;
break;
case '-':
result = num1 - num2;
break;
case '*':
result = num1 * num2;
break;
case '/':
if (num2 != 0) {
result = num1 / num2;
} else {
std::cout << "錯誤：除數不能為零！" << std::endl;
return 1; // 返回非零表示錯誤
}
break;
default:
std::cout << "錯誤：不支持的運算符！" << std::endl;
return 1;
}
std::cout << num1 << " " << operatorChar << " " << num2 << " = " << result <<
std::endl;
// 4. break的重要性（沒有break會發生穿透）
int option = 2;
std::cout << "\n沒有break的switch-case（演示穿透現象）：" << std::endl;
```

```
case 1:
std::cout << "執行case 1" << std::endl;
// 注意：這裡沒有break！
case 2:
std::cout << "執行case 2" << std::endl;
// 注意：這裡沒有break！
case 3:
std::cout << "執行case 3" << std::endl;
// 注意：這裡沒有break！
default:
std::cout << "執行default" << std::endl;
}
std::cout << "\n有break的switch-case：" << std::endl;
switch (option) {
case 1:
std::cout << "執行case 1" << std::endl;
break;
case 2:
std::cout << "執行case 2" << std::endl;
break;
case 3:
std::cout << "執行case 3" << std::endl;
break;
default:
std::cout << "執行default" << std::endl;
}
// 5. switch的限制：只能用於整數類型或枚舉
int value = 65;
switch (value) {
case 'A': // 'A'的ASCII碼是65，所以這個case會匹配
std::cout << "值等於字符'A'的ASCII碼" << std::endl;
break;
case 66:
std::cout << "值等於66" << std::endl;
break;
default:
std::cout << "其他值" << std::endl;
}
return 0;
}
```

📝 範例4：嵌套條件與複雜條件

```
/*
* 範例4-4：嵌套條件與複雜條件
* 展示複雜的條件判斷結構
*/
#include <iostream>
#include <string>
int main() {
std::cout << "=== 嵌套條件與複雜條件演示 ===" << std::endl;
// 1. 多重嵌套條件
bool isMember = true;
double purchaseAmount = 1200.0;
bool hasCoupon = true;
double discount = 0.0;
if (isMember) {
if (purchaseAmount > 1000.0) {
if (hasCoupon) {
discount = 0.2; // 會員 + 大額購買 + 優惠券：20%折扣
std::cout << "尊貴會員，您享受20%折扣！" << std::endl;
} else {
discount = 0.15; // 會員 + 大額購買：15%折扣
std::cout << "尊貴會員，您享受15%折扣！" << std::endl;
}
} else {
discount = 0.1; // 會員：10%折扣
std::cout << "尊貴會員，您享受10%折扣！" << std::endl;
}
} else {
if (purchaseAmount > 1000.0) {
discount = 0.05; // 大額購買：5%折扣
std::cout << "感謝大額購買，您享受5%折扣！" << std::endl;
} else {
discount = 0.0; // 無折扣
std::cout << "抱歉，您沒有折扣" << std::endl;
}
}
double finalAmount = purchaseAmount * (1 - discount);
```

```
std::cout << "原價：" << purchaseAmount << " 元，折扣後：" << finalAmount << "
元" << std::endl;
// 2. 使用邏輯運算符簡化嵌套條件
std::cout << "\n使用邏輯運算符簡化：" << std::endl;
int hour = 20;
bool isWeekend = true;
bool isHoliday = false;
// 判斷是否應該播放背景音樂
if ((hour >= 8 && hour <= 22) || isWeekend || isHoliday) {
std::cout << "播放背景音樂" << std::endl;
} else {
std::cout << "靜音時間，不播放音樂" << std::endl;
}
// 3. 複雜的條件組合
int age = 25;
bool hasLicense = true;
bool hasInsurance = true;
bool isIntoxicated = false;
std::cout << "\n駕駛資格檢查：" << std::endl;
std::cout << "年齡：" << age << " 歲" << std::endl;
std::cout << "有駕照：" << (hasLicense ? "是" : "否") << std::endl;
std::cout << "有保險：" << (hasInsurance ? "是" : "否") << std::endl;
std::cout << "是否酒駕：" << (isIntoxicated ? "是" : "否") << std::endl;
if (age >= 18 && hasLicense && hasInsurance && !isIntoxicated) {
std::cout << "✅ 可以合法駕駛" << std::endl;
} else {
std::cout << "❌ 不能駕駛，原因：" << std::endl;
if (age < 18) {
std::cout << " - 年齡未滿18歲" << std::endl;
}
if (!hasLicense) {
std::cout << " - 沒有駕照" << std::endl;
}
if (!hasInsurance) {
std::cout << " - 沒有保險" << std::endl;
}
if (isIntoxicated) {
std::cout << " - 酒後不能駕駛" << std::endl;
}
}
```

```
int a = 5;
int b = 10;
// 錯誤示例：少了一個等號（這是賦值，不是比較）
if (a = 10) { // 注意：這裡是賦值，不是比較！
std::cout << "\n注意：if (a = 10) 總是為真，因為賦值表達式的值為10（非零）" <<
std::endl;
std::cout << "現在 a 的值是：" << a << std::endl;
}
// 正確示例：比較
a = 5; // 恢復原始值
if (a == 10) { // 這是比較
std::cout << "這行不會執行，因為 a 不等於 10" << std::endl;
}
// 防止錯誤的技巧：將常量放在左邊
if (10 == a) { // 如果寫成 10 = a，編譯器會報錯
std::cout << "a 等於 10" << std::endl;
} else {
std::cout << "a 不等於 10" << std::endl;
}
return 0;
}
```

📝 範例5：break與continue在條件中的使用

```
/*
* 範例4-5：break與continue在條件中的使用
* 雖然break和continue主要用於循環，但在switch中也有重要作用
*/
#include <iostream>
int main() {
std::cout << "=== break與continue在條件中的使用 ===" << std::endl;
// 1. break在switch中的應用
int menuChoice = 2;
std::cout << "菜單選擇：" << std::endl;
switch (menuChoice) {
case 1:
std::cout << "1. 開始新遊戲" << std::endl;
// 這裡可以執行開始遊戲的代碼
```

```
break; // 跳出switch
case 2:
std::cout << "2. 載入遊戲" << std::endl;
// 這裡可以執行載入遊戲的代碼
break; // 跳出switch
case 3:
std::cout << "3. 遊戲設置" << std::endl;
// 這裡可以執行設置的代碼
break; // 跳出switch
case 4:
std::cout << "4. 退出遊戲" << std::endl;
// 這裡可以執行退出的代碼
break; // 跳出switch
default:
std::cout << "無效選擇，請重新輸入" << std::endl;
break; // 雖然default是最後一個，但養成習慣總是加break
}
// 2. 使用break提前結束條件判斷
std::cout << "\n使用break提前結束條件判斷：" << std::endl;
int score = 95;
bool hasCheated = false;
bool isDisqualified = false;
// 模擬一系列檢查
if (hasCheated) {
std::cout << "作弊，取消資格！" << std::endl;
isDisqualified = true;
}
if (!isDisqualified) {
if (score < 0 || score > 100) {
std::cout << "分數無效！" << std::endl;
isDisqualified = true;
}
}
if (!isDisqualified) {
if (score >= 90) {
std::cout << "優秀！" << std::endl;
} else if (score >= 60) {
std::cout << "及格" << std::endl;
} else {
std::cout << "不及格" << std::endl;
```

```
}
// 3. 在循環中使用break和continue（預覽，下一課詳細講解）
std::cout << "\n循環中的break和continue（預覽）：" << std::endl;
for (int i = 1; i <= 10; i++) {
if (i == 3) {
continue; // 跳過3
}
if (i == 8) {
break; // 在8時結束循環
}
std::cout << i << " ";
}
std::cout << std::endl;
// 4. 條件表達式的值
std::cout << "\n條件表達式的值：" << std::endl;
int x = 10, y = 20;
// 條件表達式本身有值
std::cout << "x > y 的值是：" << (x > y) << std::endl;
std::cout << "x < y 的值是：" << (x < y) << std::endl;
// 可以將條件表達式的值賦給變數
bool isGreater = (x > y);
bool isLess = (x < y);
std::cout << "isGreater: " << isGreater << std::endl;
std::cout << "isLess: " << isLess << std::endl;
// 5. 複雜條件表達式的求值順序
std::cout << "\n複雜條件表達式的求值順序：" << std::endl;
int a = 1, b = 2, c = 3;
// 注意運算符優先級：關係運算符高於邏輯運算符
bool result1 = a < b && b < c; // 等價於 (a < b) && (b < c)
bool result2 = a < b || b > c; // 等價於 (a < b) || (b > c)
bool result3 = !(a > b) && c > b; // 等價於 (!(a > b)) && (c > b)
std::cout << "a < b && b < c: " << result1 << std::endl;
std::cout << "a < b || b > c: " << result2 << std::endl;
std::cout << "!(a > b) && c > b: " << result3 << std::endl;
return 0;
}
```

📝 範例6：條件語句的實際應用

```
/*
* 範例4-6：條件語句的實際應用
* 綜合應用條件語句解決實際問題
*/
#include <iostream>
#include <string>
#include <iomanip>
int main() {
std::cout << "=== 條件語句的實際應用 ===" << std::endl;
// 1. 登錄系統
std::cout << "\n1. 登錄系統示例：" << std::endl;
std::string correctUsername = "admin";
std::string correctPassword = "123456";
std::string inputUsername, inputPassword;
// 模擬輸入
inputUsername = "admin";
inputPassword = "123456";
if (inputUsername == correctUsername && inputPassword == correctPassword) {
std::cout << "登錄成功！歡迎，" << inputUsername << std::endl;
} else {
std::cout << "登錄失敗！用戶名或密碼錯誤" << std::endl;
if (inputUsername != correctUsername) {
std::cout << "用戶名錯誤" << std::endl;
}
if (inputPassword != correctPassword) {
std::cout << "密碼錯誤" << std::endl;
}
}
// 2. 計算BMI並判斷健康狀況
std::cout << "\n2. BMI健康指數計算：" << std::endl;
double height = 1.75; // 米
double weight = 70.0; // 公斤
double bmi = weight / (height * height);
std::cout << std::fixed << std::setprecision(2);
std::cout << "身高：" << height << " 米" << std::endl;
std::cout << "體重：" << weight << " 公斤" << std::endl;
std::cout << "BMI：" << bmi << std::endl;
std::string healthStatus;
if (bmi < 18.5) {
healthStatus = "體重過輕";
```

```
} else if (bmi < 24.0) {
healthStatus = "正常範圍";
} else if (bmi < 27.0) {
healthStatus = "體重過重";
} else if (bmi < 30.0) {
healthStatus = "輕度肥胖";
} else if (bmi < 35.0) {
healthStatus = "中度肥胖";
} else {
healthStatus = "重度肥胖";
}
std::cout << "健康狀況：" << healthStatus << std::endl;
// 3. 電費計算（階梯電價）
std::cout << "\n3. 階梯電價計算：" << std::endl;
double electricityUsed = 350.0; // 度
double totalCost = 0.0;
if (electricityUsed <= 240.0) {
totalCost = electricityUsed * 0.4883; // 第一階梯
} else if (electricityUsed <= 400.0) {
totalCost = 240.0 * 0.4883 + (electricityUsed - 240.0) * 0.5383; // 第二階梯
} else {
totalCost = 240.0 * 0.4883 + 160.0 * 0.5383 + (electricityUsed - 400.0) *
0.7883; // 第三階梯
}
std::cout << "用電量：" << electricityUsed << " 度" << std::endl;
std::cout << "電費總額：" << totalCost << " 元" << std::endl;
// 4. 交通信號燈模擬
std::cout << "\n4. 交通信號燈模擬：" << std::endl;
std::string lightColor = "yellow";
int seconds = 3;
std::cout << "當前信號燈：" << lightColor << std::endl;
if (lightColor == "red") {
std::cout << "紅燈：停止" << std::endl;
std::cout << "等待時間：" << seconds << " 秒" << std::endl;
} else if (lightColor == "yellow") {
std::cout << "黃燈：準備" << std::endl;
std::cout << "剩餘時間：" << seconds << " 秒" << std::endl;
if (seconds <= 2) {
std::cout << "警告：即將變紅燈！" << std::endl;
}
```

```
std::cout << "綠燈：通行" << std::endl;
std::cout << "剩餘時間：" << seconds << " 秒" << std::endl;
} else {
std::cout << "錯誤：未知的信號燈顏色" << std::endl;
}
// 5. 學生獎學金評定
std::cout << "\n5. 學生獎學金評定：" << std::endl;
double averageScore = 88.5;
bool hasGoodBehavior = true;
bool noFailedCourses = true;
bool participatedActivities = true;
std::cout << "平均成績：" << averageScore << std::endl;
std::cout << "品德優良：" << (hasGoodBehavior ? "是" : "否") << std::endl;
std::cout << "無不及格科目：" << (noFailedCourses ? "是" : "否") << std::endl;
std::cout << "參與活動：" << (participatedActivities ? "是" : "否") <<
std::endl;
std::string scholarshipLevel = "無";
// 使用嵌套if-else評定獎學金等級
if (averageScore >= 85.0 && hasGoodBehavior && noFailedCourses) {
if (averageScore >= 95.0 && participatedActivities) {
scholarshipLevel = "特等獎學金";
} else if (averageScore >= 90.0) {
scholarshipLevel = "一等獎學金";
} else if (averageScore >= 85.0) {
scholarshipLevel = "二等獎學金";
}
} else if (averageScore >= 80.0 && hasGoodBehavior && noFailedCourses) {
scholarshipLevel = "三等獎學金";
} else if (averageScore >= 75.0 && noFailedCourses) {
scholarshipLevel = "進步獎";
}
std::cout << "獎學金等級：" << scholarshipLevel << std::endl;
// 6. 使用switch處理菜單選項
std::cout << "\n6. 簡單計算器：" << std::endl;
int choice = 3; // 1-加法 2-減法 3-乘法 4-除法
double num1 = 15.0, num2 = 3.0;
double calcResult = 0.0;
char op = ' ';
switch (choice) {
case 1:
```

```
op = '+';
break;
case 2:
calcResult = num1 - num2;
op = '-';
break;
case 3:
calcResult = num1 * num2;
op = '*';
break;
case 4:
if (num2 != 0.0) {
calcResult = num1 / num2;
op = '/';
} else {
std::cout << "錯誤：除數不能為零！" << std::endl;
return 1;
}
break;
default:
std::cout << "錯誤：無效的選擇！" << std::endl;
return 1;
}
std::cout << num1 << " " << op << " " << num2 << " = " << calcResult <<
std::endl;
return 0;
}
```

🔧 練習題

練習4-1：成績評定系統

創建一個程序，根據輸入的分數輸出對應的等級：

- 90分以上：A
- 80-89分：B
- 70-79分：C
- 60-69分：D
- 60分以下：F

要求：使用if-else語句實現，並考慮分數無效的情況（小於0或大於100）。

練習4-2：計算天數

創建一個程序，輸入月份（1-12），輸出該月份的天數。 注意：

- 1、3、5、7、8、10、12月有31天
- 4、6、9、11月有30天
- 2月需要判斷是否為閏年（年份也需要輸入）

要求：使用switch-case語句實現。

練習4-3：簡單計算器

創建一個簡單的計算器程序，支持加、減、乘、除四種運算。 程序流程：

1. 輸入第一個數字
2. 輸入運算符（+、-、*、/）
3. 輸入第二個數字
4. 輸出計算結果

要求：使用switch處理運算符，並處理除零錯誤。

練習4-4：登錄驗證系統

創建一個登錄驗證系統，要求：

1. 預設用戶名：admin，密碼：123456
2. 用戶輸入用戶名和密碼
3. 驗證是否正確，並給出相應提示：
    - 用戶名和密碼都正確：登錄成功
    - 用戶名錯誤：用戶名不存在
    - 密碼錯誤：密碼錯誤

練習4-5：稅率計算

根據收入計算應繳稅款（簡化版）：

- 收入 ≤ 50000：免稅
- 50000 < 收入 ≤ 100000：稅率10%
- 100000 < 收入 ≤ 200000：稅率15%
- 收入 > 200000：稅率20%

要求：計算並輸出應繳稅款和稅後收入。

📚 重點摘要

1. if語句的三種形式

```
// 形式1：只有if
if (條件) {
// 條件為真時執行
}
// 形式2：if-else
if (條件) {
```

```
// 條件為真時執行
} else {
// 條件為假時執行
}
// 形式3：if-else if-else
if (條件1) {
// 條件1為真時執行
} else if (條件2) {
// 條件2為真時執行
} else {
// 所有條件都為假時執行
}
```

2. switch-case語句結構

```
switch (表達式) {
case 值1:
// 當表達式等於值1時執行
break; // 跳出switch
case 值2:
// 當表達式等於值2時執行
break;
// 可以有多個case
default:
// 當表達式不等於任何case值時執行
break;
}
```

3. 重要注意事項

if語句注意事項

1. 條件必須是布林表達式（true/false）
2. 如果只有一條語句，可以省略大括號，但不推薦
3. 注意賦值運算符(=)和比較運算符(==)的區別
4. 嵌套if語句時注意縮進，提高可讀性

switch語句注意事項

1. switch表達式必須是整數類型或枚舉類型
2. case值必須是常量表達式
3. 每個case後面通常要加break，否則會發生"穿透"
4. default子句是可選的，但建議總是包含

break的作用

1. 在switch中：跳出整個switch語句
2. 在循環中：跳出當前循環（下一課詳細講解）
3. 不能跳出if語句，只能跳出switch或循環

4. 條件表達式的短路求值

```
// 邏輯與：如果第一個為假，第二個不會求值
if (條件1 && 條件2) {
// 只有條件1和條件2都為真時執行
}
// 邏輯或：如果第一個為真，第二個不會求值
if (條件1 || 條件2) {
// 只要條件1或條件2有一個為真時執行
}
```

5. 條件語句的最佳實踐

1. **使用大括號**：即使只有一條語句，也使用大括號
2. **明確優先級**：使用括號明確條件表達式的優先級
3. **避免深層嵌套**：深層嵌套降低可讀性，可以考慮重構
4. **使用switch處理多路分支**：比多個if-else更清晰
5. **包含default/default**：總是處理邊界情況
6. **保持條件簡單**：複雜條件可以拆分成多個簡單條件

🎯 常見問答

Q1: if和switch應該怎麼選擇？

A:

- 當條件是**範圍判斷**（如 `x > 10`）或複雜邏輯時，使用if
- 當條件是**離散值**比較（如 `x == 1` 、 `x == 2`）時，使用switch
- 當分支超過3個時，switch通常更清晰

Q2: 為什麼switch的case後面要加break？

A: 如果不加break，程序會繼續執行下一個case的代碼，這稱為"穿透"。大多數情況下我們不需要穿透，所以需要break。少數情況下故意利用穿透特性時，應該添加註釋說明。

Q3: if和switch哪個性能更好？

A: 一般來說，switch在多分支時性能更好，因為編譯器可能會使用跳轉表優化。但現代編譯器對if也有優化，所以差異通常不明顯。優先考慮代碼的可讀性。

Q4: 可以嵌套使用switch嗎？

A: 可以，但不推薦，因為會降低代碼可讀性。如果必須嵌套，要確保層次不要太深。

Q5: switch可以用字符串嗎？

A: C++標準不允許switch使用字符串，因為字符串比較不是常量時間操作。如果要根據字符串選擇分支，可以使用if-else或std::map。

🚀 下一步

下一課我們將學習：

- **流程控制：循環結構**（for、while、do-while）
- **循環控制語句**（break、continue）
- **嵌套循環**
- **循環的常見應用場景**

💻 完整測試程序

```
/*
* 第4課綜合測試
* 綜合應用條件語句
*/
#include <iostream>
#include <string>
#include <iomanip>
int main() {
std::cout << "=== 第4課綜合測試：條件分支 ===" << std::endl;
// 1. 年齡分組
std::cout << "\n1. 年齡分組：" << std::endl;
int age = 25;
std::string ageGroup;
if (age < 0) {
ageGroup = "無效年齡";
} else if (age < 13) {
ageGroup = "兒童";
} else if (age < 18) {
ageGroup = "青少年";
} else if (age < 65) {
ageGroup = "成年人";
} else {
ageGroup = "老年人";
}
std::cout << age << " 歲屬於：" << ageGroup << std::endl;
// 2. 計算折扣
std::cout << "\n2. 購物折扣計算：" << std::endl;
```

```
bool isMember = true;
double purchaseAmount = 800.0;
double discountRate = 0.0;
if (isMember) {
if (purchaseAmount > 1000.0) {
discountRate = 0.2;
} else if (purchaseAmount > 500.0) {
discountRate = 0.15;
} else {
discountRate = 0.1;
}
} else {
if (purchaseAmount > 1000.0) {
discountRate = 0.05;
}
}
double discountAmount = purchaseAmount * discountRate;
double finalAmount = purchaseAmount - discountAmount;
std::cout << std::fixed << std::setprecision(2);
std::cout << "購物金額：" << purchaseAmount << " 元" << std::endl;
std::cout << "折扣率：" << (discountRate * 100) << "%" << std::endl;
std::cout << "折扣金額：" << discountAmount << " 元" << std::endl;
std::cout << "最終金額：" << finalAmount << " 元" << std::endl;
// 3. 星期幾判斷
std::cout << "\n3. 星期判斷：" << std::endl;
int day = 3; // 1=星期一，7=星期日
std::string dayName;
std::string dayType;
switch (day) {
case 1:
case 2:
case 3:
case 4:
case 5:
dayName = (day == 1) ? "星期一" :
(day == 2) ? "星期二" :
(day == 3) ? "星期三" :
(day == 4) ? "星期四" : "星期五";
dayType = "工作日";
break;
```

```
dayName = "星期六";
dayType = "週末";
break;
case 7:
dayName = "星期日";
dayType = "週末";
break;
default:
dayName = "無效";
dayType = "無效";
break;
}
std::cout << "第 " << day << " 天是 " << dayName << "，屬於 " << dayType <<
std::endl;
// 4. 成績轉換
std::cout << "\n4. 百分制轉等級制：" << std::endl;
int score = 87;
char grade;
std::string comment;
if (score >= 90) {
grade = 'A';
comment = "優秀！";
} else if (score >= 80) {
grade = 'B';
comment = "良好";
} else if (score >= 70) {
grade = 'C';
comment = "中等";
} else if (score >= 60) {
grade = 'D';
comment = "及格";
} else {
grade = 'F';
comment = "不及格";
}
std::cout << "分數：" << score << std::endl;
std::cout << "等級：" << grade << std::endl;
std::cout << "評語：" << comment << std::endl;
// 5. 三角形判斷
std::cout << "\n5. 三角形類型判斷：" << std::endl;
```

```
std::string triangleType;
// 檢查是否為有效三角形
if (side1 + side2 > side3 &&
side1 + side3 > side2 &&
side2 + side3 > side1) {
if (side1 == side2 && side2 == side3) {
triangleType = "等邊三角形";
} else if (side1 == side2 || side1 == side3 || side2 == side3) {
triangleType = "等腰三角形";
} else if (side1*side1 + side2*side2 == side3*side3 ||
side1*side1 + side3*side3 == side2*side2 ||
side2*side2 + side3*side3 == side1*side1) {
triangleType = "直角三角形";
} else {
triangleType = "一般三角形";
}
} else {
triangleType = "不是三角形";
}
std::cout << "邊長：" << side1 << ", " << side2 << ", " << side3 << std::endl;
std::cout << "類型：" << triangleType << std::endl;
// 6. 登錄系統
std::cout << "\n6. 登錄系統：" << std::endl;
std::string username = "user123";
std::string password = "pass456";
std::string inputUser = "user123";
std::string inputPass = "wrongpass";
if (inputUser == username) {
if (inputPass == password) {
std::cout << "登錄成功！歡迎，" << username << std::endl;
} else {
std::cout << "密碼錯誤！" << std::endl;
}
} else {
std::cout << "用戶名不存在！" << std::endl;
}
return 0;
}
```

輸出結果：

```
=== 第4課綜合測試：條件分支 ===
1. 年齡分組：
25 歲屬於：成年人
2. 購物折扣計算：
購物金額：800.00 元
折扣率：15.00%
折扣金額：120.00 元
最終金額：680.00 元
3. 星期判斷：
第 3 天是 星期三，屬於 工作日
4. 百分制轉等級制：
分數：87
等級：B
評語：良好
5. 三角形類型判斷：
邊長：3.0, 4.0, 5.0
類型：直角三角形
6. 登錄系統：
密碼錯誤！
```

✅ 本課檢查清單

- [ ] 能夠正確使用if、else if、else結構
- [ ] 理解並能正確使用switch-case結構
- [ ] 掌握break在switch中的正確使用
- [ ] 能夠編寫嵌套條件語句
- [ ] 理解條件表達式中的邏輯運算符
- [ ] 能夠處理邊界情況和錯誤輸入
- [ ] 能夠使用條件語句解決實際問題
- [ ] 了解if和switch的適用場景

📞 需要幫助？

如果你在練習中遇到問題：

1. 檢查條件表達式是否正確（特別是==和=的區別）
2. 確保switch的case後面有break（除非故意要穿透）
3. 注意if語句的大括號使用，避免懸掛else問題
4. 測試邊界條件（如最小值、最大值、特殊值）

下一課見！ 🚀