

C++程式數計：例外處理

目錄

C++程式數計：檔案處理	1
1. 資料流	錯誤! 尚未定義書籤。
2. 檔案 (File)	錯誤! 尚未定義書籤。
3. 檔案分類	錯誤! 尚未定義書籤。
3.1 文字檔 (Text File) :	錯誤! 尚未定義書籤。
3.2 二進位檔 (Binary File) :	錯誤! 尚未定義書籤。
3.3 檔案存取方式	錯誤! 尚未定義書籤。
4. 檔案的輸出入管理	錯誤! 尚未定義書籤。
4.1 檔案開啟	錯誤! 尚未定義書籤。
4.2 文字檔寫入	錯誤! 尚未定義書籤。
4.3 文字檔讀取	錯誤! 尚未定義書籤。
4.4 二進位檔的讀取	錯誤! 尚未定義書籤。
5. 隨機存取模式	錯誤! 尚未定義書籤。

1. 常見的例外情況

- 除數為零 (Division by zero)
- 陣列下標越界 (Array out-of-bounds)
- 整數溢位 (Overflow)
- 函數參數無效 (Invalid arguments)
- 使用 `new` 分配記憶體失敗

1.1 try-catch 區塊的基本語法：

```
// 當程式出現錯誤時，會「丟出」(throw) 例外。
// 如果有對應的 `catch` 區塊，系統就會「捕捉」(catch) 到例外並加以處理。
// 若未處理，將導致程式終止或異常行為。
```

```
try {
    // 可能會發生錯誤的程式碼
} catch (ExceptionType e) {
    // 處理該錯誤的方式
}
```

1.2 範例：try 指令

```
#include <iostream>
```

```
using namespace std;

int main() {
    cout << "\n 簡單的例外範例\n";
    try {
        int n1;
        cout << "請輸入除數：";
        cin >> n1;

        if (n1 == 0) throw 1;
        cout << "沒有錯誤繼續執行" << endl;
    }
    catch (int i) {
        cout << "捕捉到除數為 0 的例外\n";
    }
    cout << "程式正常結束執行" << endl;
    return 0;
}
```

1.3 範例：catch 區塊多載

使用多個 catch 區塊處理不同型別例外

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

int main() {
    int num;
    try {
        cout << "輸入一個值:";
        cin >> num;

        if ((num > 0) && (num < 10)) throw 1;      // 整數型別
        if ((num > 10) && (num < 20)) throw 0.99; // 實數型別
    }
    catch (int ex1) {
        cout << "執行區段內的 catch 匹配 int" << endl;
    }
    catch (double ex2) {
        cout << "執行區段內的 catch 匹配 double" << endl;
    }
    cout << "程式繼續執行" << endl;
}
```

```
    return 0;
}
```

1.4 範例：一次捕捉所有例外

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

int main() {
    int num;
    cout << "輸入 num 的值：";
    cin >> num;
    try {
        if ((num > 10) && (num < 20)) throw 1;
        if (num < 10) throw '*';
    }
    catch (...) {
        cout << "目前是由 catch(...) 捕捉到例外" << endl;
    }
    return 0;
}
```