Homework8

日期:2024/11/10
撰寫者:喻慈恩

題目:

請把你為第三題寫的 Matrix 類別拿出來重新檢視，並且判斷每一個成員函數是否應該是constant 成員函數、每個成員函數的回傳值是否應該是 constant，以及每個成員函數的參數是否應該被加上 constant。

作答時，請把這個類別的宣告(包含成員變數和成員函數的宣告，但不包含成員函數的定義)貼到這一題，在合適的地方加上關鍵字 const，並且在每一個有 const 的地方簡要地說明為什麼這裡應該要有 const。

最後，請說明 Matrix 為什麼需要 constructor、copyconstructor、assignment、operator、destructor，以及這些函數應該扮演什麼功能、完成什麼任務。

回答:

1.類別宣告截圖

```
 8  class Matrix {
 9  private:
10      int n;
11      int m;
12      int** data;
13
14  public:
15      char name;
16      Matrix();
17      Matrix(int n, int m);
18      Matrix(const Matrix& mat);
19      ~Matrix();
20
21      friend ostream& operator<<(ostream& out, const Matrix& mat);
22      friend istream& operator>>(istream& in, Matrix& mat);
23      Matrix& operator=(const Matrix& mat);
24      Matrix operator+(const Matrix& mat) const;
25      Matrix operator-(const Matrix& mat) const;
26      Matrix operator*(const Matrix& mat) const;
27      Matrix operator*(int num) const;
28      Matrix operator!() const;
29      bool isEmpty() const;
30  };
```

2.const解釋

`const Matrix& mat`(在 **`operator<<`** 和 **`operator+`**, **`operator-`**, **`operator*`** 中)：這些參數使用 `const` 修飾，表示它們在函數內不可被修改，確保傳入的 `Matrix` 物件保持原狀。特別在運算子和 I/O 操作中，這樣可以避免意外更動數據，提高程式安全性。

`Matrix operator+(const Matrix& mat) const` 等函數結尾的 **`const`**(在 **`operator+`**, **`operator-`**, **`operator*`**, **`operator!`**, **`isEmpty`** 中)：這些運算子結尾的 `const` 表示這些函數不會更改物件的狀態，因此可以被 `const` 物件調用，例如在常數環境下操作的情況。這對於運算子、比較和檢查類的函數尤其重要。

3.有關Matrix內的函數

Constructor：`Matrix` 的Constructor(包括預設Constructor與參數化Constructor)負責初始化矩陣物件。初始化物件的行數和列數(`n` 和 `m`)，並分配內部的資料儲存空間(如動態分配 `data`)。這樣可以在宣告 `Matrix` 物件時直接定義其大小和形狀。

CopyConstructor：在新創建的 `Matrix` 物件作為已存在物件的拷貝時被呼叫，負責複製原有物件的資料(深拷貝)，以避免兩個物件共用同一內部資料空間並造成記憶體衝突。

AssignmentOperator：處理已存在物件的賦值操作。此函數應該包含檢查自我賦值的步驟，並進行深拷貝，確保不會將不同的物件指向相同記憶體位址。

Destructor：負責釋放建構過程中動態分配的記憶體，以防止記憶體洩漏。當物件生命周期結束時，解構子自動被呼叫來清除 `data` 所佔的空間。