程式設計（106-1）
作業十二

作業設計：孔令傑
國立臺灣大學資訊管理學系

繳交作業時，請至 PDOGS（`http://pdogs.ntu.im/judge/`）為第一題上傳一個 PDF 檔、為第二題做同儕互評，再為第三、四題各上傳一份 C++ 原始碼（以複製貼上原始碼的方式上傳）。第四題是 bonus 加分題。每位學生都要上傳自己寫的解答。不接受紙本繳交；不接受遲交。請以英文或中文作答。

這份作業的截止時間是 ***2017 年 12 月 19 日凌晨一點***。在你開始前，請閱讀課本的第 9–10 章[1]。為這份作業設計測試資料並且提供解答的助教是楊其恆。

第一題

（40 分）請回答以下兩個問題：

(a)（10 分）請考慮課堂練習時介紹的 `Time`，去實做一個成員函數

```
int Time::earlierBySec(Time t);
```

使得 invoking object 呼叫此函數並得到另一個 `Time` 物件 `t` 之後，可以回傳 invoking object 的時間比 `t` 的時間早幾秒。如果 invoking object 的時間比 `t` 晚，則回傳負數。舉例來說，

```
Time t1(14, 30, 0);
Time t2(14, 25, 5);

cout << t1.earlierBySec(t2);
```

會印出 −295。

(b)（10 分）承上題，請寫一個 global 函數

```
int earlierBySec(Time earlyTime, Time lateTime);
```

去回傳 `earlyTime` 的時間比 `lateTime` 的時間早幾秒。如果 `earlyTime` 的時間比 `lateTime` 的晚，則回傳負數。舉例來說，

```
Time t1(14, 30, 0);
Time t2(14, 25, 5);

cout << earlierBySec(t1, t2);
```

會印出 −295。你可能會發現需要幫 `Time` 加 getter，或是幫 `Time` 設定 friend。請問哪一種比較好？為什麼？不論是哪一種，請說明應如何修改 `Time`。

[1]課本是 Deitel and Deitel 著的 *C++ How to Program: Late Objects Version* 第七版。

(c)（20 分）請考慮課堂練習時介紹的 `Event`。請解釋在投影片第九頁我們新增 `Event::setName()` 但沒有寫 copy constructor 時，為什麼 `e1.printNicely()` 那一句指令可能會出錯。

第二題

（0 分）請在 PDOGS 上批改你被隨機分配到的作業十一第四題的程式碼，根據它在正確性以外的部份給它 1 至 5 分的評分，並且說明你給分的依據。建議在評分時參考以下六個面向。在前五個面向上，一個面向上做得好就得一分，還不錯則半分，不好則零分；在第六個面向上則在有必要時扣分。六個面向的分數合計後無條件進入即為你最後給的總分。

- 可讀性：變數與函數名稱是否具有合適的資訊量？程式碼排版是否良好且具有前後一致性？是否有合適的註解？關於註解，當然不需要每一行都有註解，但若你發現在某一大段落裡都沒有註解，或某個你感覺很不易看懂的部份沒有註解，你可以指出來；不要直接說「註解太少」但沒有說是哪邊缺乏註解。
- 模組化程度：是否有宣告合適的函數、structure 或 class[2]？是否有避免將非常類似的程式片段寫複數次而非寫成函數？是否有避免一個函數做非常多事情？函數間是否有合適的 decoupling？直接閱讀 main function 是否能很快地理解程式在大方向上的運算邏輯？
- 效率：程式運算是否有合理的運算效率？當然我們不要求每個同學都寫出超級有效率的精妙演算法，但至少一個程式不應該進行過多不必要的運算，也不應該耗用過多不必要的記憶體空間。如果你看不出這個程式的效率有明顯的問題，我們建議你直接給一分。
- 擴充性：當要解的問題變得更複雜的時候，我們能不能簡單地修改這個程式以解決新的問題，而不是寧可砍掉重練？這個議題當然也很主觀，所以如果你不能明確地指出在怎樣的新問題上，這個程式會有擴充性問題，我們建議你直接給一分；如果你不能指出很嚴重的問題，我們建議你至少給半分。但對批改者來說，這個關於擴充性的思考其實是很好的訓練。試試看吧！
- 其他：如果有任何其他令你想扣分的理由，請明確地寫出來並且在這個面向上扣分；沒有的話就給一分。
- 題目規範：你應該檢查那份程式碼有沒有違反題目的規範，如果有（例如題目說不可以用上課沒教過的東西，但他用了，或者題目說一定要用指標和動態記憶體配置，但他沒用），就扣他三分。當然，請明確地指出他哪邊違反了題目的規範。

本題其中 10 分取決於檢視你的程式碼的同學給你的分數總和（必要時助教會出來主持公道，請不用緊張），另外 10 分取決於你對同學的程式碼的評語和評分的合理性和建設性（原則上除非被申訴，且助教檢視後發現你確實評得很不公允，否則只要有評就會得到 10 分）。

第三題

（40 分）我們曾經學過基本上就是字元陣列與字元指標的 C string，好用歸好用，有些地方也挺不方便的（例如靜態字元陣列能存的字數有限、`strcat` 等函數有超出陣列邊界的可能）。既然我們學過 class，

[2]我們還沒教 class，所以本次批改請看函數和 structure 就好。

現在該是結合兩者的時候了！我們將要實作一個 class `MyString`，以便我們所建立的每個 `MyString` 物件都是一個字串，並且透過封裝的概念讓其他程式設計師方便使用。

首先，我們的 `MyString` 勢必要有如下的成員變數：

```
class MyString
{
private:
  int len;
  char* str;
// something else...
};
```

其中 `len` 是此字串目前有幾個字元，`str` 則是指向該字串的指標。請注意因為我們不想要規範一個字串的上限，所以我們使用動態陣列而非靜態陣列。

在成員函數方面，當然我們應該要有 constructor 和 destructor：

```
MyString::MyString();
MyString::MyString(char* c);
MyString::MyString(const MyString& c);
MyString::~MyString();
```

default constructor 應該讓 `str` 指向 `nullptr`，`len` 自然就應該被設成 0。接收一個字元指標（亦即一個 C string）的 constructor，應該要求系統動態配置合適的記憶體空間，裡面存有該 C string 的內容，接著將 `str` 指向該空間。copy constructor 和 destructor 則各做它們該做的事。

接著我們想要實做幾個成員函數：

```
int MyString::length();
int MyString::find(char ch, int pos = 1);
void MyString::assign(const char* st);
void MyString::assign(MyString mStr);
void MyString::concat(const char* st);
void MyString::concat(MyString mStr);
char* MyString::getStr();
void MyString::print();
```

說明如下：

- `length()` 會回傳目前字串的字元個數。
- `find()` 會從字串的第 `pos` 個字元開始往後搜尋，回傳第一個遇到的 `ch` 所在的位置編號。在這個字串裡，我們讓第一個字元編號為 1，第二個字元編號為 2，依此類推。如果 `ch` 不存在，則回傳 0。
- `assign()` 會把這個字串徹底改成 `st` 或 `mStr` 中的字串。
- `getStr()` 會回傳現在這個字串的初始字元的記憶體位址。

- `print()` 會印出現在這個字串。

舉例來說，假設我們完整地實作了 `MyString`，並且執行下方程式：

```
int main()
{
  MyString s1("that"), s2;
  MyString s3(s1);
  cout << s1.length() << "\n";
  cout << s3.find('a') << "\n";
  cout << s3.find('a', 3) << "\n";
  cout << s1.find('h', 3) << "\n";
  s1.assign("Hi, it is me!");
  s2.assign(s3);
  char* aString = s1.getStr();
  cout << aString[4] << "\n";
  s2.print();
  return 0;
}
```

則輸出會是

```
4
3
3
0
i
that
```

在本題中，你將會被給定許多字串間的運算，請照著運算並將要求的輸出印出。

輸入輸出格式

系統會提供一共 20 組測試資料，每組測試資料裝在一個檔案裡。在每個檔案中會有 $n+1$ 行，第一行包含一個整數 n，第二行起則每行包含一個動作，格式如下：

- 建立字串物件：在這行中，首先是一個英文單詞 `declare`，接著一個空格，接著一個字串物件的變數名稱，接著一個空格，接著一個由雙引號括起來的字串，最後是一個換行字元。此時請新增一個物件，並以雙引號內的內容為該字串物件變數所存的字串。
- 複製字串物件：在這行中，首先是一個英文單詞 `copy`，接著一個空格，接著是第一個字串物件的變數名稱，接著一個空格，接著是第二個字串物件的變數名稱，最後是一個換行字元。此時請把第二個字串的內容複製以建立第一個字串。
- 字串長度：在這行中，首先是一個英文單詞 `length`，接著一個空格，接著一個字串物件的變數名稱，最後是一個換行字元。此時請把該字串變數所存字串的長度印出，並在最後換行。

- 字元搜尋：在這行中，首先是一個英文單詞 `find`，接著一個空格，接著一個字串物件的變數名稱，接著一個空格，接著一個要被搜尋的字元 `ch`，接著一個空格，接著一個大於等於 1 的整數代表搜尋起始位置 `pos`，最後是一個換行字元。此時請印出該字串物件呼叫 `find(ch, pos)` 的結果，並在最後換行。
- 字串指派：在這行中，首先是一個英文單詞 `assignString`，接著一個空格，接著一個字串物件的變數名稱，接著一個空格，接著一個由雙引號括起來的字串，最後是一個換行字元。此時請把該字串物件內存的字串改寫成雙引號內的內容。
- 物件指派：在這行中，首先是一個英文單詞 `assignObject`，接著一個空格，接著是第一個字串物件的變數名稱，接著一個空格，接著是第二個字串物件的變數名稱，最後是一個換行字元。此時請把第一個字串物件內存的字串改寫成第二個字串物件內存的字串。
- 字串輸出：在這行中，首先是一個英文單詞 `print`，接著一個空格，接著一個字串物件的變數名稱，最後是一個換行字元。此時請印出該字串物件中所存的字串，並在最後換行。

當你要更新一個字串物件內所存的字串時，你當然也要更新該物件內所存的字串長度。

對於上述給定的資訊，你可以做一些假設：

- 字串物件的變數名稱只含有英文字母和數字，且第一個字一定是字母。
- 雙引號括起來的字串內，可能含有英文字母、數字、空白、以及各式鍵盤上能直接打出來的英文標點符號，但不含單引號、雙引號、斜線、反斜線。
- `copy` 後面的第一個字串物件一定還沒被建立過，其他所有被提到的字串物件一定已經被建立過了。
- $1 \le n \le 100$、字串物件最多 10 個、字串物件的變數名稱最長 20 個字元、雙引號內的字串內容最長 50 個字。

讀入資料後，請根據給定的規則印出相關資訊。舉例來說，如果輸入是

```
11
declare s1 "that"
declare s2 ""
copy s3 s1
length s1
find s3 a 1
find s3 a 3
find s3 h 3
assignString s1 "Hi, it is me!"
assignObject s2 s3
print s1
print s2
```

則輸出應該是

```
4
3
3
0
Hi, it is me!
that
```

你上傳的原始碼裡應該包含什麼

你的.cpp 原始碼檔案裡面應該包含讀取測試資料、做運算，以及輸出答案的 C++ 程式碼。當然，你應該寫適當的註解。針對這個題目，你***不可以***使用上課沒有教過的方法，但上課提過的 library 裡面的所有功能都可以用。除此之外，你**一定要**用 class。

評分原則

- 這一題的其中 20 分會根據程式運算的正確性給分。PDOGS 會編譯並執行你的程式、輸入測試資料，並檢查輸出的答案的正確性。一筆測試資料佔 2 分。
- 這一題的其中 20 分會在作業十三中被評定。屆時我們會讓同學們互相檢視彼此的本題程式碼，並且就可讀性、易維護性、模組化程度、排版等面向寫評語和給評分（當然一切都是匿名的）。該任務在本題中會佔 20 分，其中 10 分取決於檢視你的程式碼的同學給你的分數（必要時助教會出來主持公道，請不用緊張），另外 10 分取決於你對同學的程式碼的評語和評分的合理性和建設性。若你在本次作業中完全沒有寫這一題，那屆時自然沒有人能檢視你的程式碼，你也就得要損失這 10 分了。

第四題（bonus）

（40 分）承上題，我們現在要進一步擴充我們的 `MyString`。我們要加上兩個成員函數：

```
void MyString::concat(const char* st);
void MyString::concat(MyString mStr);
```

`concat()` 會把 `st` 或 `mStr` 中的字串接在現在這個字串後面。舉例來說，假設我們完整地實作了 `MyString`，並且執行下方程式：

```
int main()
{
  MyString s1("that");
  MyString s3(s1);
  s1.concat(s3);
  s1.print();
  return 0;
}
```

則輸出會是

```
thatthat
```

除此之外，我們也會放寬我們對測試資料的限制。首先，一行指令中給定的字串物件有可能還沒被建立過，或者 copy 後面的第一個字串物件可能已經被建立過了。此時請忽略整個指令，不要執行任何動作（當然也不要印出一行空字串）。其次，雙引號內可能含有斜線、反斜線、單引號、雙引號，並且以 escape sequence 的方式編寫。你可以假設雙引號內的字串是一個包含 escape sequence 的合法字串。

輸入輸出格式

系統會提供一共 10 組測試資料，每組測試資料裝在一個檔案裡。資料的輸入輸出格式都和第三題幾乎一模一樣，但輸入方面多了兩種指令：

- 字串串接：在這行中，首先是一個英文單詞 concatString，接著一個空格，接著一個字串物件的變數名稱，接著一個空格，接著一個由雙引號括起來的字串，最後是一個換行字元。此時請在該字串物件內存的字串後面串接雙引號內的內容。
- 物件串接：在這行中，首先是一個英文單詞 concatObject，接著一個空格，接著是第一個字串物件的變數名稱，接著一個空格，接著是第二個字串物件的變數名稱，最後是一個換行字元。此時請在第一個字串物件內存的字串後面串接第二個字串物件內存的字串。

當你要更新一個字串物件內所存的字串時，你當然也要更新該物件內所存的字串長度。

讀入資料後，請根據給定的規則印出相關資訊。舉例來說，如果輸入是

```
7
declare s1 "that\\"
copy s3 s1
concatObject s1 s3
copy s1 s3
copy s4 s2
print s2
print s1
```

則輸出應該是

```
that\that\
```

請注意有兩個指令有牽涉到 s2，但 s2 不曾被宣告過，因此那兩個指令都被完全略過了。此外，copy s1 s3 雖然 s3 被宣告過，但 s1 也被宣告過了，不符合 copy 後面的第一個字串物件應該沒被宣告過的要求，因此這一個指令也要被忽略。

你上傳的原始碼裡應該包含什麼

你的.cpp 原始碼檔案裡面應該包含讀取測試資料、做運算，以及輸出答案的 C++ 程式碼。當然，你應該寫適當的註解。針對這個題目，你可以使用任何方法，但你**一定要**用 class。

評分原則

這一題的全部 20 分會根據程式運算的正確性給分。PDOGS 會編譯並執行你的程式、輸入測試資料，並檢查輸出的答案的正確性。一筆測試資料佔 2 分。