資料結構與進階程式設計（106-2）
程式作業一

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/`）為第一題上傳一份 C++ 原始碼（以複製貼上原始碼的方式上傳），為第二題上傳一份 PDF 檔，為第三題做同儕互評。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。

這份作業的前兩題的截止時間是 ***2018 年 3 月 26 日下午一點***；第三題的截止時間是 ***2018 年 3 月 30 日下午一點***。在你開始前，請閱讀課本的第 8、10、11 和 18 章[1]。為這份作業設計測試資料並且提供解答的是兩位助教。

第一題

（60 分）如大家所知，任何基本資料型態都有其被分配的記憶體空間。例如以 C++ 來說，在現在大部份的編譯器上 `int` 是 4 位元組、`double` 是 8 位元組。以整數來說，4 位元組讓我們可以表現大約 -2^{31} 和 2^{31} 間的整數，也就是大約負 21 億到正 21 億之間。如果想要儲存的整數超過這個範圍，就不能用 `int` 存了。但就算是 `long int` 或 `double`，總也是有個上限。

為了讓我們可以存更大（或更小）的整數，讓我們來寫一個 class 叫 `BigInt`（大整數）吧！這個 class 的原理，是利用一個型態為 `string` 的 instance variable 來存數值，並且提供許多 instance function 供外界存取此值。讓我們假設在我們的應用中，我們的 `BigInt` 需要可以印出值、寫入值、彼此做加、減、乘、除的四則運算、取餘數、取相反數、取平方、取絕對值、取出某個指定的位數、判斷是否為質數：

- 印出值（`<<`）：對於運算 `cout << b`，只要 `a` 是 `BigInt`，就要印出 `a` 所代表的值，例如

  ```
  BigInt a = 9876543210;
  cout << a;
  ```

  要印出

  ```
  9876543210
  ```

  此運算要回傳一個 `ostream`，讓此回傳值要可以立刻再跟別的印出運算串接。

- 寫入值（`>>`）：對於運算 `cin >> b`，只要 `a` 是 `BigInt`，就要讓使用者輸入一個整數，並將之寫入 `a`，例如

  ```
  BigInt a = 9876543210;
  cin >> a;
  cout << a;
  ```

  若使用者輸入 `-1`，則要印出

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

```
-1
```

此運算要回傳一個 `istream`，讓此回傳值要可以立刻再跟別的寫入運算串接。

- 加法（`+`）：對於運算 `a + b`，只要 `a` 或 `b` 有至少一個是 `BigInt`，就要回傳一個 `BigInt`，其值是 `a` 和 `b` 所代表的值的和，例如

```
BigInt a = 9876543210;
int b = 1;
cout << a + b;
```

要印出

```
9876543211
```

此回傳值要可以立刻再跟別的 `BigInt` 或 `int` 做運算，但不可以放在一個 assignment operator 的左邊。

- 減法（`-`）：同加法，除了回傳值應該回傳 `a` 代表的值減掉 `b` 代表的值，例如

```
BigInt a = 9876543210;
int b = 1;
cout << a - b;
```

要印出

```
9876543209
```

乘法（`*`）：同加法，除了回傳值應該回傳 `a` 代表的值乘上 `b` 代表的值，例如

```
BigInt a = 1234412344;
int b = 2;
cout << a * b;
```

要印出

```
2468824688
```

- 除法（`/`）：同加法，除了回傳值應該回傳 `a` 代表的值除以 `b` 的商數，例如

```
BigInt a = 8866442211;
int b = 2;
cout << a / b;
```

要印出

```
4433221105
```

- 取餘數（%）：同加法，除了回傳值應該回傳 a 代表的值除以 b 的餘數，例如

```
BigInt a = 8866442211;
int b = 2;
cout << a % b;
```

要印出

```
1
```

如果 a 和 b 之中有負數，請根據 C++ 處理 int 的取餘數規則做處理。

- 取相反數（-）：對於運算 -a，如果 a 是 BigInt，就要回傳一個 BigInt，其值是 a 所代表的值乘以負一，例如

```
BigInt a = 9876543210;
cout << -a;
```

要印出

```
-9876543210
```

此回傳值要可以立刻再跟別的 BigInt 或 int 做運算，但不可以放在一個 assignment operator 的左邊。

- 取平方（square()）：對於運算 a.square，如果 a 是 BigInt，就要回傳一個 BigInt，其值是 a 所代表的值的平方，例如

```
BigInt a = 99999;
cout << a.square();
```

要印出

```
9999800001
```

此回傳值要可以立刻再跟別的 BigInt 或 int 做運算，但不可以放在一個 assignment operator 的左邊。

- 取絕對值（abs()）：對於運算 a.abs()，如果 a 是 BigInt，就要回傳一個 BigInt，其值是 a 取絕對值，例如

```
BigInt a = -9876543210;
cout << a.abs();
```

要印出

```
9876543210
```

此回傳值要可以立刻再跟別的 BigInt 或 int 做運算，但不可以放在一個 assignment operator 的左邊。

- 取出某個指定的位數（[]）：對於運算 a[i]，如果 a 是 BigInt，就要回傳一個 int，其值是 a.abs() 的值的第 i 位，其中 0 表示個位數、1 表示十位數，依此類推，例如

```
BigInt a = -9876567890;
cout << a[3];
```

要印出

```
7
```

此回傳值要可以立刻再跟別的 BigInt 或 int 做運算，但不可以放在一個 assignment operator 的左邊。如果傳入的 i 是負的或超過位數上限，請回傳 −1。

- 判斷是否為質數（isPrime()）：對於運算 a.isPrime()，如果 a 是 BigInt，就要回傳一個 bool，若 a 的值是質數則為 true，反之則為 false。例如

```
BigInt a = 9876543210;
cout << a.isPrime();
```

要印出

```
0
```

此回傳值要可以立刻再跟別的 int 做運算，但不需要可以立刻再跟別的 BigInt 做運算，但不可以放在一個 assignment operator 的左邊。

請自行定義 BigInt 這個 class，並且實作上述所有的 overloaded operators 與 member function。若你需要更多其他東西，你都可以自行定義（例如整數位數、變數名稱之類的）。你的 BigInt 要能夠記錄 100 位以內的負整數或正整數。你可以假設所有的測試資料在經過各種運算後，都不會超過 100 位。

你的程式裡面要有一個 main function 去讀取測試資料、將值存入許多 BigInt 物件、讓這些 BigInt 交互運算，並且輸出結果。

輸入輸出格式

系統會提供一共 30 組測試資料，每組測試資料裝在一個檔案裡。在每個檔案中會有若干行，每一行都長得「像是」一個 C++ 敘述，有以下幾種。所有底下提到的「變數名稱」都是長度不超過 10 個字元、都是英文或數字的字串。

- 宣告一個變數並且初始化：該行會以 int 開頭，接著一個空白字元，接著一個變數名稱，接著一個空白字元，接著一個等號，再一個空白，接著一個整數，最後一個分號，例如

```
int i = 123456789012345;
```

此時請在你的程式中建立一個 BigInt 物件，並把給定的值存入。

- 印出整數的值：該行會以 cout << 開頭，接著一個空白字元，接著一個被宣告過的變數名稱，最後一個分號，例如

```
cout << i;
```

此時請印出該 BigInt 物件的值，然後印出一個換行符號。

- 加法：該行會以一個已經宣告的變數名稱開頭，接著一個空白字元，接著一個等號，接著一個空白字元，接著一個加法運算式，最後一個分號。該運算式左邊和右邊各是一個已經宣告的變數或一個整數，而加法運算子與兩側的運算元中間各以一個空格隔開，例如

```
k = i + j;
k = 123 + j;
```

此時請在你的程式中執行該加法運算，並將回傳值寫入開頭的變數。

- 減法：同加法，只是加法運算式換成減法運算式。例如

```
k = i - 456;
m = 789 - 1024;
```

此時請在你的程式中執行該減法運算，並將回傳值寫入開頭的變數。

- 乘法：同加法，只是加法運算式換成乘法運算式。例如

```
k = i * 456;
m = i * j;
```

此時請在你的程式中執行該乘法運算，並將回傳值寫入開頭的變數。

- 除法：同加法，只是加法運算式換成除法運算式。例如

```
k = i / 456;
m = 789 / 1024;
```

此時請在你的程式中執行該除法運算，並將回傳值寫入開頭的變數。

- 取餘數：同加法，只是加法運算式換成取餘數運算式。例如

```
k = i % 456;
m = 789 % 1024;
```

此時請在你的程式中執行該取餘數運算，並將回傳值寫入開頭的變數。

- 取相反數：該行會以一個已經宣告的變數名稱開頭，接著一個空白字元，接著一個等號，接著一個空白字元，接著一個負號，接著一個已經宣告的變數名稱，最後一個分號，例如

```
k = -i;
```

此時請在你的程式中執行該相反數運算，並將回傳值寫入開頭的變數。

- 取平方：該行會以一個已經宣告的變數名稱開頭，接著一個空白字元，接著一個等號，接著一個空白字元，接著一個已經宣告的變數名稱，接著是 .square()，最後一個分號，例如

```
k = i.square();
```

此時請在你的程式中執行該取平方運算，並將回傳值寫入開頭的變數。

- 取絕對值：同取平方，只是 `.square()` 改成 `.abs()`。例如

```
k = i.abs();
```

此時請在你的程式中執行該絕對值運算，並將回傳值寫入開頭的變數。

- 印出某個指定的位數：該行會以 `cout <<` 開頭，接著一個空白字元，接著一個被宣告過的變數名稱，接著一對 `[]` 中間包含一個整數，最後一個分號，例如

```
cout << i[3];
```

此時請印出該位數之值（或 `-1`）與一個換行符號。

- 判斷是否為質數：該行會以 `cout <<` 開頭，接著一個空白字元，接著一個被宣告過的變數名稱，接著是 `.isPrime()`，最後一個分號，例如

```
cout << i.isPrime();
```

此時若該變數是質數則輸出 `1`，反之則輸出 `0`。

- 印出運算的回傳值：該行會以 `cout <<` 開頭，接著一個空白字元，接著一個運算式或函數呼叫，最後一個分號，例如

```
cout << i + j;
```

此時請印出該運算回傳之 `BigInt` 物件的值，然後印出一個換行符號。該運算式只會包含已經被宣告的變數，若為一元運算子則運算子與運算元中間沒有空格，若為二元運算子則運算子與左或右運算元中間皆有一個空格，若為函數呼叫則該變數與 `.` 中間無空格，`.` 與函數名稱間也無空格。

讀入資料後，請一行一行地執行相對應的運算，並根據運算結果新增變數、修改變數的值，並且印出結果。舉例來說，如果輸入是

```
int i = 9876543210;
int j = 74;
int k = 0;
k = i + j;
cout << k / 16;
int m = 0;
m = i % j;
cout << m;
m = -m;
m = m - k;
cout << m[6];
cout << m[18];
```

```
cout << m.abs();
k = m.abs();
k = k / 2;
k = 134003230 + k;
cout << k.isPrime();
```

則輸出應該是

```
617283955
10
6
-1
9876543294
1
```

其中最後一行印出 1 是因為 k 是質數。

本題中敘述的個數不限，但每筆測試資料中被宣告出來的變數個數最多 20 個。

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。

除此之外，你***必須***實做題目指定的 class、overloaded operator 和成員函數，並且利用這些東西來完成運算。若你沒有這麼做，「程式的品質」部份將被扣分。

評分原則

這一題的分數都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。測試資料的複雜度資訊如下：

- 前 10 筆測試資料只含有「宣告一個變數並且初始化」、「印出整數的值」、「取相反數」與「取絕對值」。
- 第 11 到 15 筆測試資料除了前面那些，還多了「加法」與「減法」。
- 第 16 到 20 筆測試資料除了前面那些，還多了「乘法」與「取平方」。
- 第 21 到 25 筆測試資料除了前面那些，還多了「除法」、「取餘數」與「印出某個指定的位數」。
- 第 26 到 30 筆測試資料除了前面那些，還多了「判斷是否為質數」與「印出運算的結果」。

第二題

（20 分）請為你在第一題設計的 class `BigInt` 寫一份給開發者的說明文件，讓其他開發者（或未來的你）能快速理解你設計的 `BigInt`。你只需要說明你的設計，不需要說明你的實做，因此你應該：

1. 展示你的 `BigInt`，列出所有成員（instance variable、static variable、instance function、static function）的所有資訊（變數的類別、名稱、初始值等；函數的完整 header）。
2. 針對每一個成員變數，說明其在 `BigInt` 中作何用途。
3. 針對每一個成員函數，說明其參數的性質與用途、其回傳值的性質與用途，以及呼叫此函數前後你的成員變數會有何變化。特別是若你寫了 constant member function 或回傳了 constant returned value，請說明原因；若你傳入了 constant、reference、constant reference 或回傳了 constant、reference、constant reference，請說明原因。
4. 加上其他應該補充的資訊。

你的說明應該精確而簡潔（precise and concise）。所謂的精確就是不模糊（廢話?!），例如「執行完此函數後，某某陣列內的整數會被排序」就比「執行完此函數後，某某陣列內的整數會被由小到大排序」不精確。所謂的簡潔就是用字精要、少說廢話。這份文件***不能超過四頁***，超過的部份不列入批改。

第三題

（20 分）第一題截止後，我們會讓同學們互相檢視彼此的本題程式碼。請在 PDOGS 上批改你被隨機分配到的第一題程式碼，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？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？
- class 設計：是否有宣告與設計合適的 class？這些 class 是否都確實應該被用來生成物件？是否提供了太多不必要的 public function 因而破壞了 data hiding？是否把做為 global function 更合適的函數寫成了 member function，或把做為 member function 更合適的函數寫成了 global function？
- 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，或者題目說一定要這樣那樣做，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（必要時助教會出來主持公道，請不用緊張）；若你在本次作業中完全沒有寫第一題，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。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；只要你有完成評語和評分，且沒人檢舉你或有人檢舉你但助教檢視後認為你沒問題，你就會得到這 10 分。