

# **EE1003 Introduction to Computer I**

## **Programming Assignment 3 Big Number Calculator**

**授課教師：陳聿廣教授**

**學生：王佑恩**

**學號：108601205**

**系級：電機三 A**

**2022.12.21**

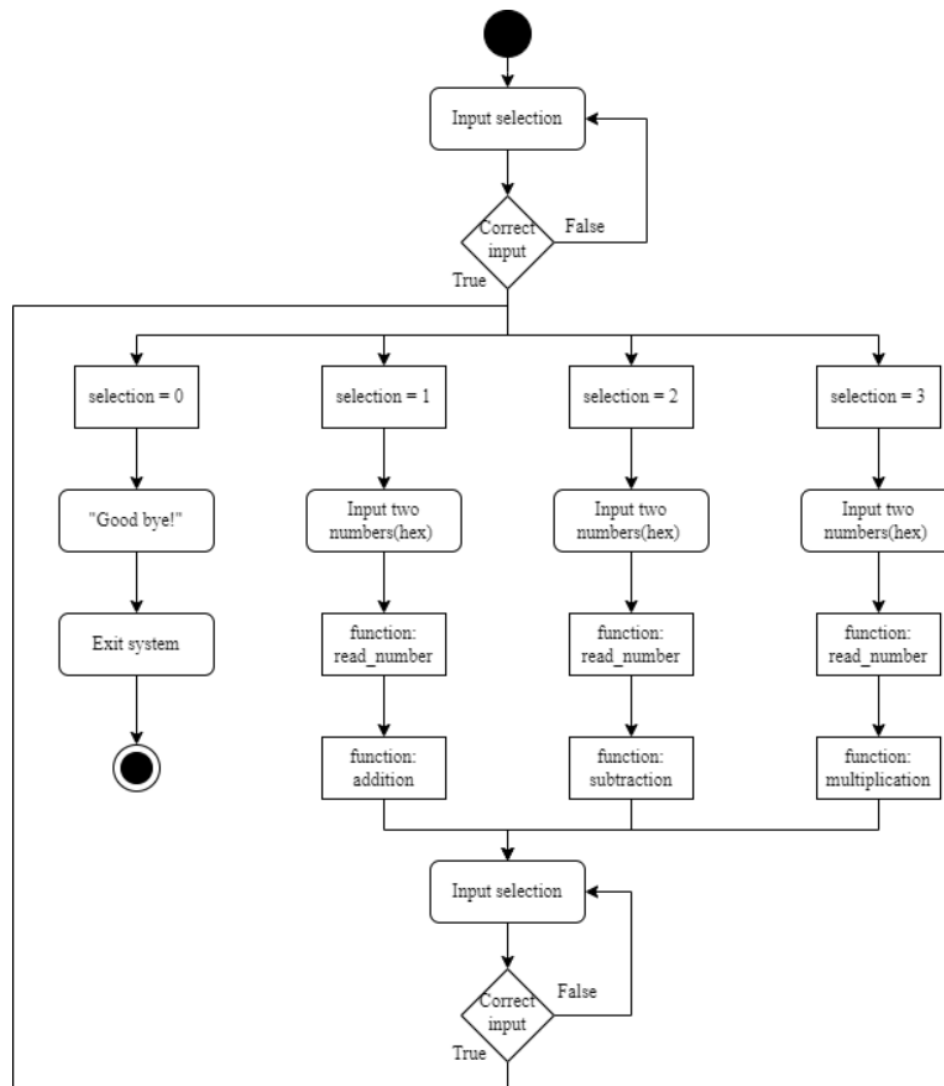
## 一、 題目簡介

我們曾經學過 long long 它可以用來存到 9,223,372,036,854,775,807 這麼大的數字，若我們需要比這個數字更大的數字，例如 20 位數、100 位數、1000 位數這種數字，就算是用了 long long 也沒有辦法進行處理。這種連 long long 都無法存的數字，我們就稱作「大數」。

在本次 Programming Assignment 中，我們需要達成兩個大數的加法、減法以及乘法，此兩個大數之位數上限為 100 位，為十六進位制且都為正數。

## 二、 程式敘述

### 1. 程式運作分析



圖一、程式運作主要流程圖

此次程式最主要的邏輯即是在各個算術運算，將會在後續論述說明，而主要流程圖如圖一。在讀取使用者輸入之大數方面，我選擇以 string 來進行讀取，接著透過 function: read\_number，將輸入的大數轉為 integer 以利後續運算。

使用者可以透過輸入 0~3 不同的 selection 來選擇想要的操作。輸入 0 為離開程式，而輸入 1~3 則分別進行加法、減法以及乘法。對應的算術運算有對應的 function 進行運作。

## 2. Pseudocode

Function Declaration: 決定此程式所需的所有 function，需要的 function 功能大致有「讀取數字」、「加法」、「減法」、「乘法」以及「將數字印出」。

Prompt the user to enter selection

當 selection 不為 0，程式會持續進行。

按照使用者所選的算術運算，來進行後續運作。

selection = 1:

Prompt the user to enter the two numbers(hex)

加法運算需要考慮進位，並將結果存進新的 array 當作答案。

印出 16 進制的答案

selection = 2:

Prompt the user to enter the two numbers(hex)

減法運算需要考慮退位，並將結果存進新的 array 當作答案。

印出 16 進制的答案

selection = 3:

Prompt the user to enter the two numbers(hex)

乘法運算需要考慮進位以及 arraysize，並將結果存進新的 array 當作答案。

印出 16 進制的答案

使用者再次輸入 selection，若為 1~3 則繼續做新的運算，若為 0 則離開程式。

### 3. 程式參數簡介

根據題意，我們需要在 main function 宣告多個參數：

Parameter	Data Type	Meaning	Range
selection	int	使用者依想要的操作進行輸入	0, 1, 2, 3
str1	string	存取使用者輸入的第一個大數	—————
str2	string	存取使用者輸入的第二個大數	—————
arraysize	int	依照不同的算術運算以及兩個大數的位數，來決定運算以及輸出答案時的 array size	0 ~ 201
a	int []	透過 function: read_number，將第一個大數轉為 integer 以利後續運算	element: 0~15 array size: 100
b	int []	透過 function: read_number，將第二個大數轉為 integer 以利後續運算	element: 0~15 array size: 100
answer	int []	將各個運算的結果存入	element: 0~15 array size: 201

### 4. 程式內 function 簡介

依照所發想的 Pseudocode，我們需要做出多個 function：

Fuction	Meaning	Prototype
read_number	將輸入的大數轉為 integer 並存進整數 array 中以便後續運算	void read_number(string, const int, int[])
addition	將兩個 array(a[], b[]) 中的 element 相加並處理進位	void addition(int [], int[], int [], const int)
subtraction	將兩個 array(a[], b[]) 中的 element 相減並處理退位	void subtraction(int [], int[], int [], const int)
multiplication	將兩個 array(a[], b[]) 中的 element 相乘並處理進位	void subtraction(int [], int[], int [], const int)
show_hex_answer	把整數型態之 array 中的值依序轉為 char 並存成 string 印出	void show_hex_answer(int [], string, const int)

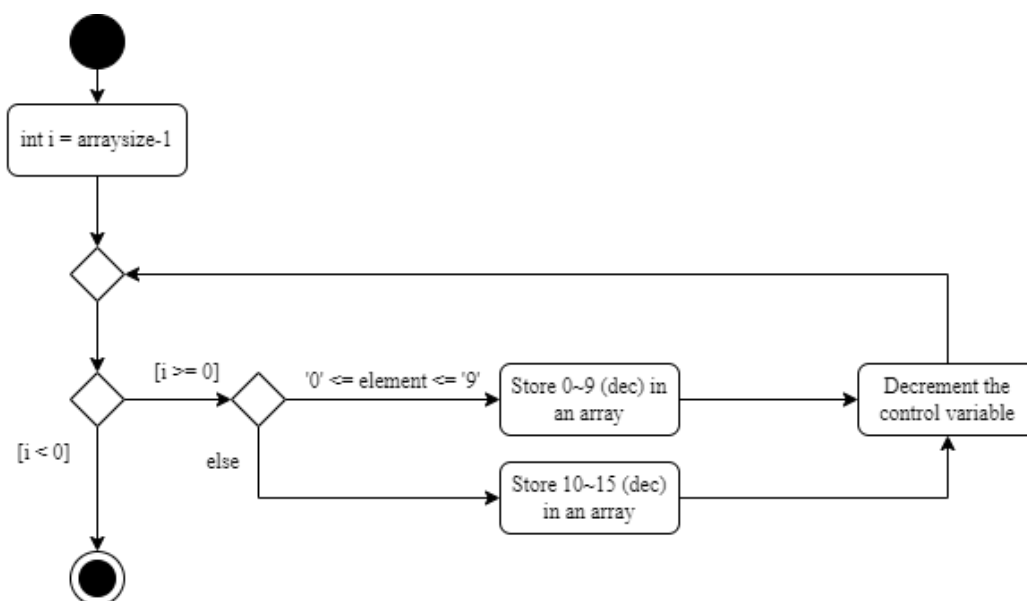
## 5. Function 細節介紹

### (1) read\_number

```

7 void read_number(string str, const int arraysize, int number[])
8 {
9     for(int i = arraysize-1; i >= 0; i--)
10     {
11         if(str[i] >= '0' && str[i] <= '9')
12         {
13             number[arraysize - i - 1] = str[i] - '0';
14         }
15         else
16         {
17             number[arraysize - i - 1] = str[i] - 'A' + 10;
18         }
19     }
20 }

```



圖二、function: read\_number 流程圖

- (a) 目的：將 char 換成 int 以利後續計算。
- (b) 因為 string 內儲存的 element 為 16 進制的 char，藉由 if...else statement 判斷 element 所對應的 10 進制之值並將其存入 array 中。
- (c) 第 13 與 17 行：存 10 進制數字時，將 string 從右至左的 element 轉換後，由左至右存進 array，以利後續運算。

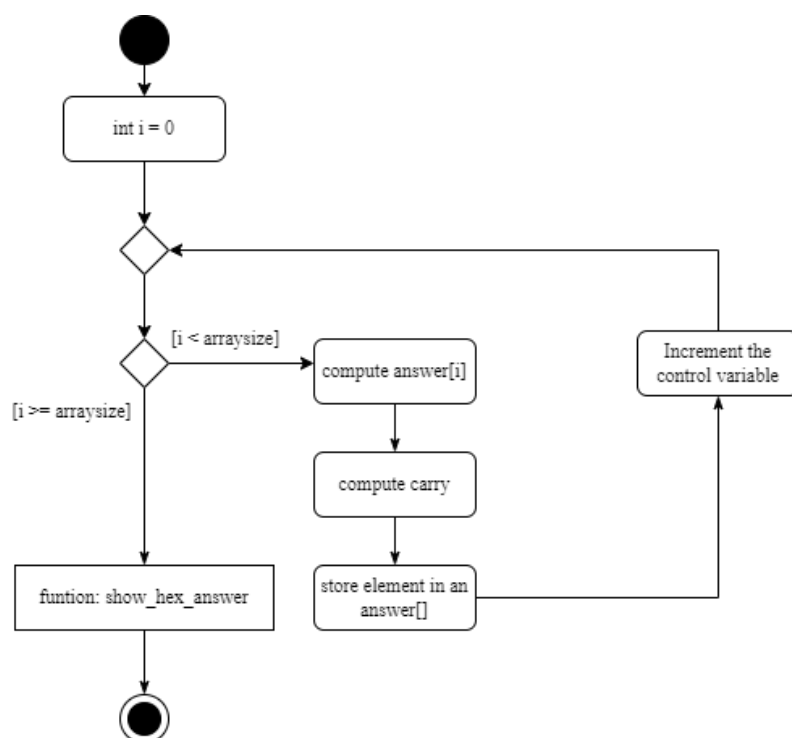
### (2) addition

```

73 void addition(int a[], int b[], int answer[], const int arraysize)
74 {
75     int carry = 0;
76     string answer_hex;
77
78     for(int i = 0; i < arraysize; i++)
79     {
80         answer[i] = a[i] + b[i] + carry;
81         carry = answer[i] / 16;
82         answer[i] = answer[i] % 16;
83     }
84
85     show_hex_answer(answer, answer_hex, arraysize);
86 }

```

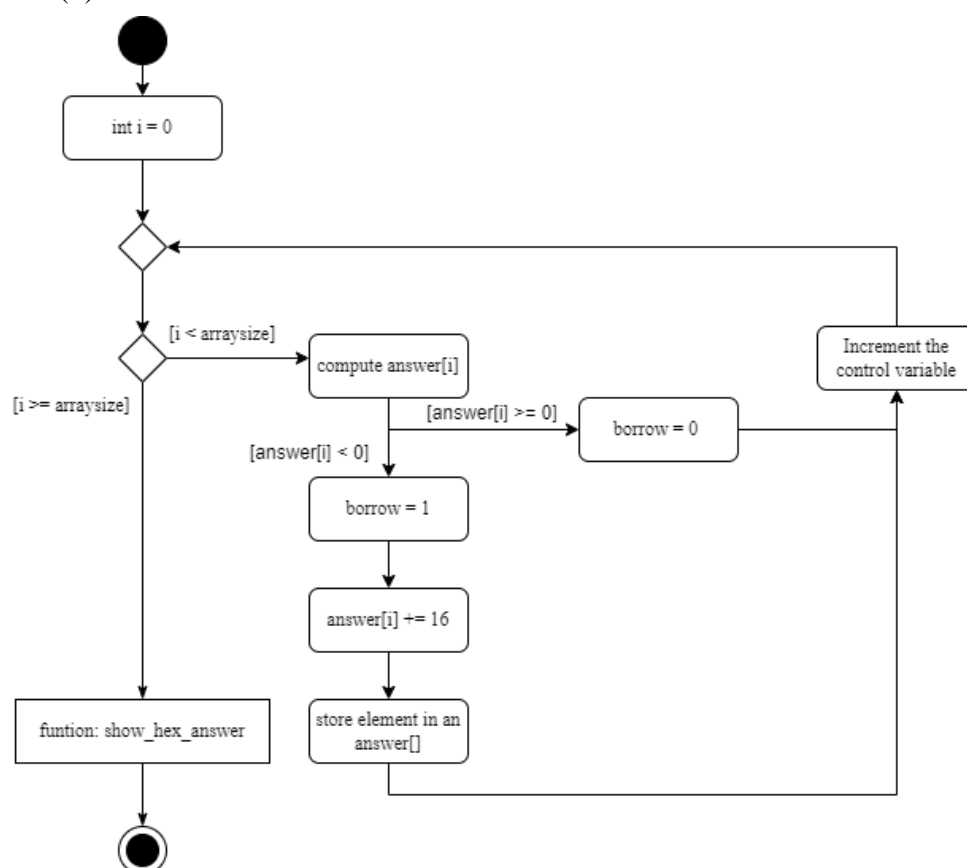
// 計算相加後的進位  
// 將正確(小於16)的數值放進answer中



圖三、funtion: addition 流程圖

- (a) 目的：計算兩個大數相加。
- (b) 變數 carry：表示為相加後的進位。
- (c) 流程如下：
- 計算  $a[i]$ 、 $b[i]$  以及 carry 相加的結果存入  $answer[i]$ （第 80 行）。
  - 計算進位 carry，若是前一步驟計算出的  $answer[i]$  大於或等於 16，則 carry 大於 0（第 81 行）。
  - 因已經計算出進位，所以最後存入  $answer[i]$  的數值便是一個小於 16 且為 10 進制的數值（第 82 行）。
  - control variable 加 1 後，第二步算出的進位 carry 則會與 array answer 的下一個 element 做相加（第 80 行）。
  - 重複以上步驟直至  $i \geq arraysize$  即計算完整數型態的 array answer。
  - 將整數型態的 array answer 送入 function: show\_hex\_answer 可將其轉換為 16 進制並印出結果。

## (3) subtraction



圖四、functon: subtraction 流程圖

```

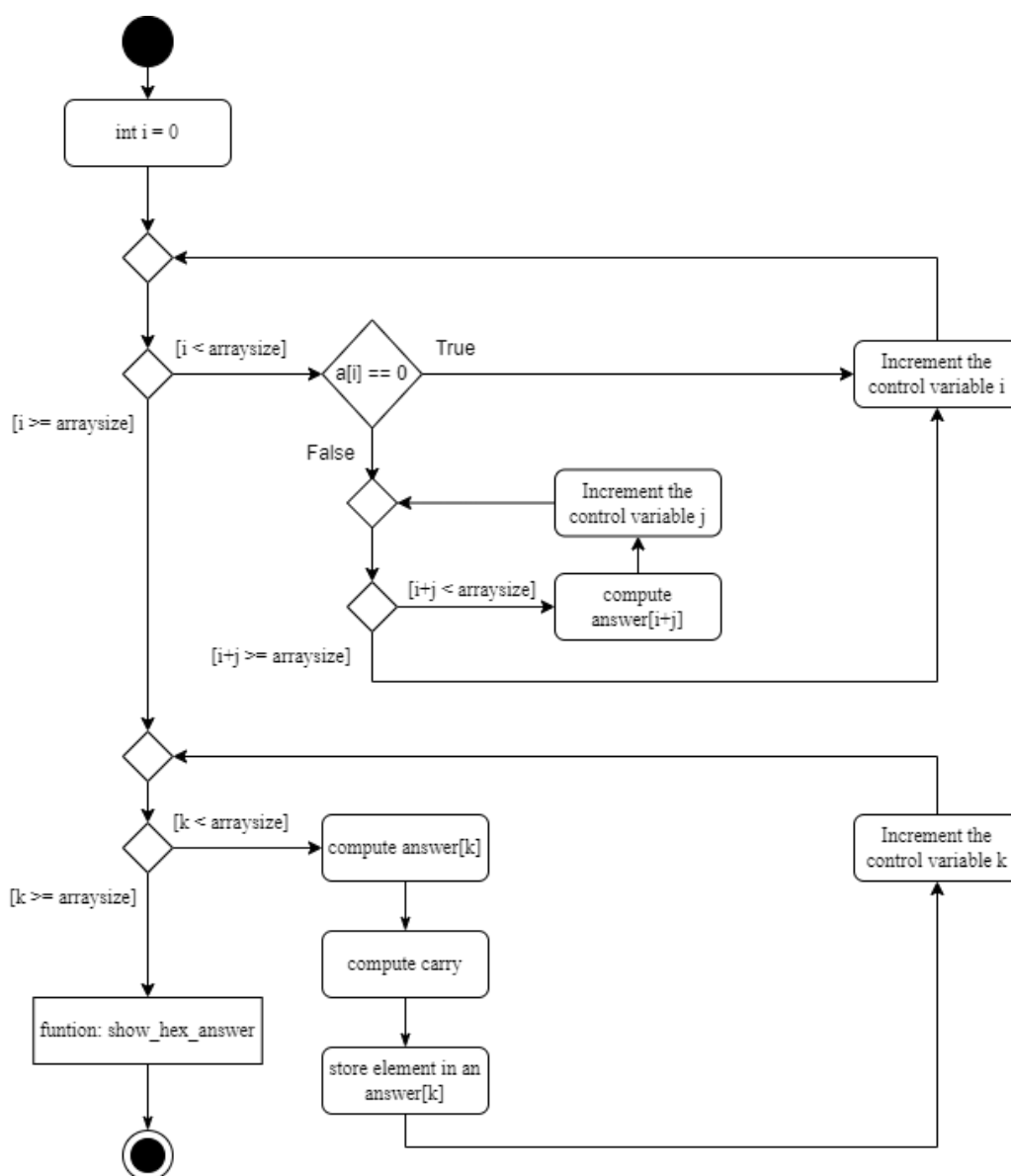
88 void subtraction(int a[], int b[], int answer[], const int arraysize)
89 {
90     int borrow = 0;
91     string answer_hex;
92
93     for(int i = 0; i < arraysize; i++)
94     {
95         answer[i] = a[i] - b[i] - borrow;
96         if(answer[i] < 0)
97         {
98             borrow = 1;
99             answer[i] += 16;
100         }
101         else
102         {
103             borrow = 0;
104         }
105     }
106
107     show_hex_answer(answer, answer_hex, arraysize);
108 }
  
```

// 借16給下一位數，所以在下次的for-loop中會被多減1  
 // 若計算後為負數，即頭上一位數借16來相加

- (a) 目的：計算兩個大數相減。
- (b) 變數 borrow：表示為相減後的退位。
- (c) 流程如下：
- 計算  $a[i]$ 、 $b[i]$  以及 borrow 相減的結果存入  $answer[i]$  (第 95 行)。
  - 計算退位 borrow，若是前一步驟計算出的  $answer[i]$  小於 0，代表相減的結果為負數，則 borrow 等於 1；大於或等於 0 則 borrow 等於 0 (第 98 與 103 行)。

- III. 因已經計算出退位，若  $\text{answer}[i]$  為負數，在減法計算上需要跟下一位數借 16 來相加（第 99 行）；若  $\text{answer}[i]$  大於或等於 0，則直接存入  $\text{answer}[i]$ （第 95 行）。所以最後存入  $\text{answer}[i]$  的數值便是一個大於或等於 0 且為 10 進制的數值。
- IV. 重複以上步驟直至  $i \geq \text{arraysize}$  即計算完整數型態的 array  $\text{answer}$ 。
- V. 將整數型態的 array  $\text{answer}$  送入 function:  
 $\text{show\_hex\_answer}$  可將其轉換為 16 進制並印出結果。

#### (4) multiplication



圖五、functon: multiplication 流程圖



```

110 void multiplication(int a[], int b[], int answer[], const int arraysze)
111 {
112     int carry = 0;
113     string answer_hex;
114     for(int i = 0; i < arraysze; i++)
115     {
116         if(a[i] == 0)
117         {
118             continue;
119         }
120         for(int j = 0; i+j < arraysze; j++)
121         {
122             answer[i+j] = answer[i+j] + a[i]*b[j];
123         }
124     }
125     for(int k = 0; k < arraysze; k++)
126     {
127         answer[k] += carry;
128         carry = answer[k]/16;
129         answer[k] = answer[k]%16;
130     }
131     show_hex_answer(answer, answer_hex, arraysze);
132 }

```

- (a) 目的：計算兩個大數相乘。
- (b) 變數 carry：表示為相乘後的進位。
- (c) 流程如下：
- 將被乘數 a[i] 乘以乘數 array b 的每個 element，若 a[i] 為 0 則跳至下一位數，並將結果依序存入 answer[i+j]，直至被乘數 array a 的所有 element 都已經與乘數 array b 的每個 element 做過相乘。
  - 將 answer 內每個 element 與進位 carry 相加，並存入 answer[k] (第 129 行)。
  - 計算進位 carry，若是前一步驟計算出的 answer[k] 大於或等於 16，則 carry 大於 0 (第 130 行)。
  - 因已經計算出進位，所以最後存入 answer[k] 的數值便是一個小於 16 且為 10 進制的數值 (第 131 行)。
  - 重複以上步驟直至 k ≥ arraysze 即計算完整數型態的 array answer。
  - 將整數型態的 array answer 送入 function: show\_hex\_answer 可將其轉換為 16 進制並印出結果。

(5) show\_hex\_answer

```

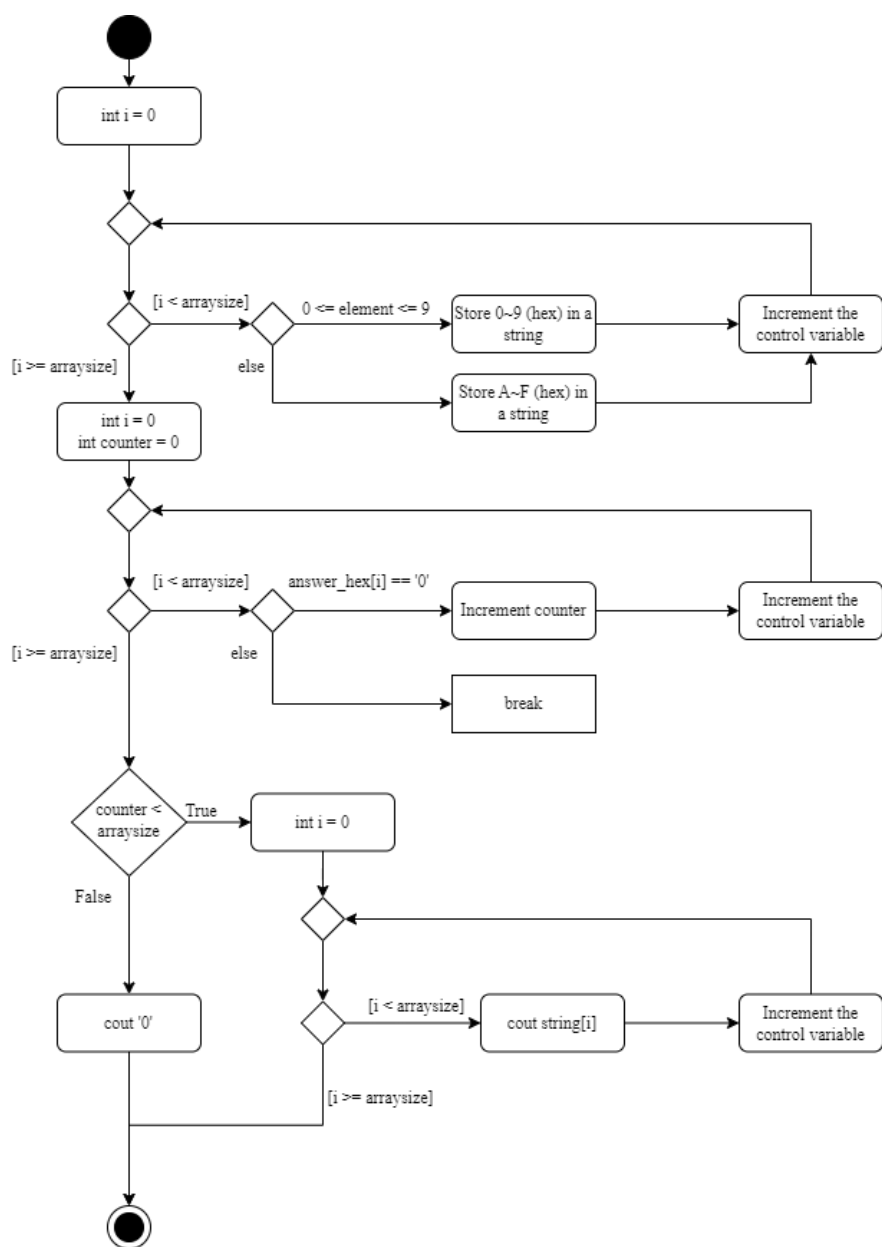
25 void show_hex_answer(int answer[], string answer_hex, const int arraysze)
26 {
27     // 將10進位轉換為各自的16進位(ex: 10 -> 'A')
28     // 將數字以存進string，而array從右至左各element的16進位會由右至左存進string
29     // 此轉換可較方便的印出16進位的結果
30     cout << "Result(hex): ";
31     for(int i = 0; i < arraysze; i++)
32     {
33         if(answer[i] >= 0 && answer[i] <= 9){
34             answer_hex[arraysze-i-1] = answer[i] + 48;
35         }
36         else{
37             answer_hex[arraysze-i-1] = answer[i] + 65 - 10;
38         }
39     }
40 }

```

```

41
42     int counter = 0;
43
44     for(int i = 0; i < arraysize; i++)           // 尋找answer_hex從頭開始第幾個字母非0
45     {
46         if(answer_hex[i] == '0'){
47             counter++;
48         }
49         else{
50             break;
51         }
52     }
53
54     if(counter < arraysize){
55         for(int i = counter; i < arraysize; i++)
56         {
57             cout << answer_hex[i];
58         }
59     }
60     else{
61         cout << "0";                             // 當answer_hex為0時，需要印出0
62     }
63
64     cout << "\n" << endl;
65 }

```



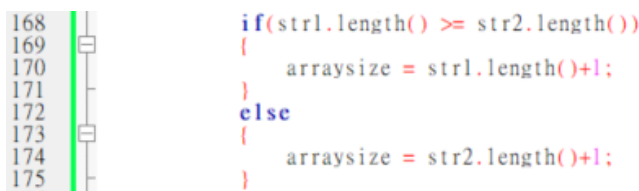
圖六、funtion: show\_hex\_answer 流程圖

- (a) 目的：將整數型態的 array answer 轉換為 16 進制並印出結果。
- (b) 變數 counter：用來計算 16 進制的 string 中，從左至右第幾個 element 非 '0'。
- (c) 流程如下：
  - I. 判斷整數型態的 array answer 內各個 element 所對應的 16 進制之值並將其存入 string answer\_hex 中。
  - II. 為了方便後續印出 string 中的 element，將 array 從左至右的 element 轉換後，由右至左存進 string 內（第 35 與 38 行）。
  - III. 為了避免印出的結果有多餘的 0，透過 for statement 找出 string 中從左至右第幾個 element 非 '0'。
  - IV. 把 string answer\_hex 內，從第 counter 的 element 開始將值印出。但若上一步驟計算所得的 counter 等於 arraysize，表示 string answer\_hex 內的 element 皆為 0，亦即先前算術運算的結果為 0，則將 0 印出。

## 6. 主程式運作 (main function)

除了印出使用者提示，以及運作 choose the wrong function foolproof 外，main function 中最重要的事判斷送入各個 function 的 arraysize，分析如下：

- (a) Addition:



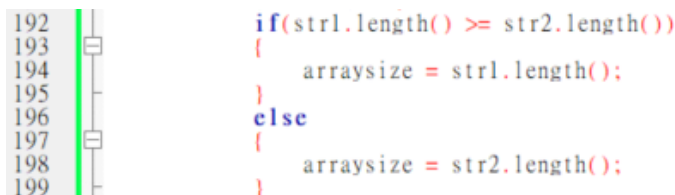
```

168 if(str1.length() >= str2.length())
169 {
170     arraysize = str1.length()+1;
171 }
172 else
173 {
174     arraysize = str2.length()+1;
175 }

```

藉由比較兩個大數的長度，將最長的長度再加 1 設為 array size。舉例：兩個位數皆為 3 的正數相加，相加的結果最長為 4 位數，此便是加 1 的原因。

- (b) Subtraction:



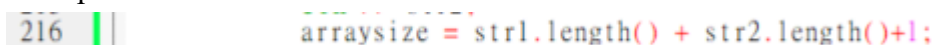
```

192 if(str1.length() >= str2.length())
193 {
194     arraysize = str1.length();
195 }
196 else
197 {
198     arraysize = str2.length();
199 }

```

藉由比較兩個大數的長度，將最長的長度設為 array size，即為兩個大數相減後的最大可能位數。

- (c) Multiplication:



```

216 arraysize = str1.length() + str2.length()+1;
217

```

兩個大數相乘，所得結果的最大可能位數雖為兩個大數的長度相加，但由於 function: multiplication 運作時，會多用到一個位數，故將 array size 再加 1。

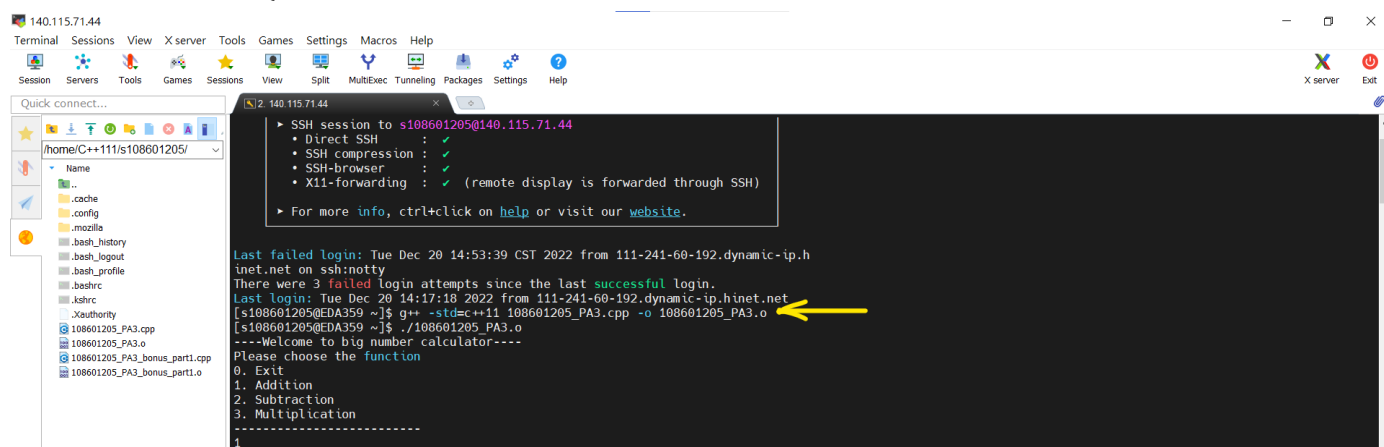
(d) 備註：

由於 function: show\_hex\_answer 的設計，有將多餘的 0 過濾掉的功能，故在 main function 中決定 array size 時，只需要找到可能的最大 array size 即可，不需要考慮最小的 array size。

### 三、 How to compile and execute program

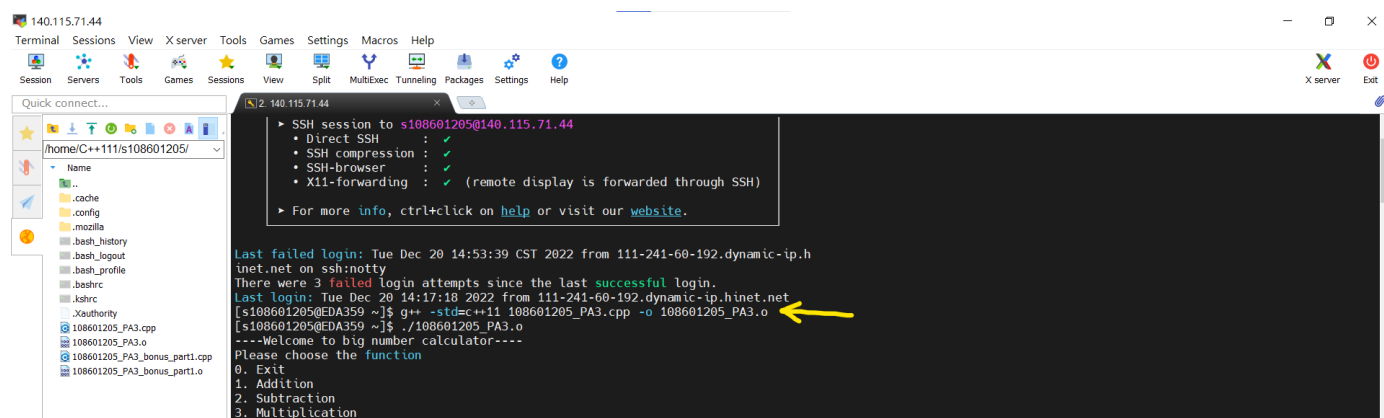
Step1.

將檔案上傳至 Workstation (System: Linux 3.10.0, Software: gcc/g++: 4.8.5 (C++ 11 supported))後，運用“g++ -std=c++11” command 將 output file 生成。



Step2.

運用“./108601205\_PA3.o”進程式執行。  
(bonus\_part1 則是運用“./108601205\_PA3\_bonus\_part1.o”進程式執行)



#### 四、 結果呈現

```
Last login: Tue Dec 20 14:17:18 2022 from 111-241-60-192.dynamic-ip.hinet.net
[s108601205@EDA359 ~]$ g++ -std=c++11 108601205_PA3.cpp -o 108601205_PA3.o
[s108601205@EDA359 ~]$ ./108601205_PA3.o
----Welcome to big number calculator----
Please choose the function
0. Exit
1. Addition
2. Subtraction
3. Multiplication
-----
1
----Now for Addition----
Please enter the two numbers(hex).
3F2A6967C02AC
13BCA5B404A7FE
Result(hex): 17AF4C4A80AAAA

----Welcome to big number calculator----
Please choose the function
0. Exit
1. Addition
2. Subtraction
3. Multiplication
-----
5
Error! Please try again.

----Welcome to big number calculator----
Please choose the function
0. Exit
1. Addition
2. Subtraction
3. Multiplication
-----
3
----Now for Multiplication----
Please enter the two numbers(hex).
1111111111
1111111111
1111111111
Result(hex): 123456789ABA987654321

----Welcome to big number calculator----
Please choose the function
0. Exit
1. Addition
2. Subtraction
3. Multiplication
-----
1
----Now for Addition----
Please enter the two numbers(hex).
996B3DA728
2E5BF271
Result(hex): 9999999999

----Welcome to big number calculator----
Please choose the function
0. Exit
1. Addition
2. Subtraction
3. Multiplication
-----
5
Error! Please try again.
```

```
----Welcome to big number calculator----
Please choose the function
0. Exit
1. Addition
2. Subtraction
3. Multiplication
-----
2

----Now for Subtraction----
Please enter the two numbers(hex).
9999999999
1111111111
Result(hex): 8888888888

----Welcome to big number calculator----
Please choose the function
0. Exit
1. Addition
2. Subtraction
3. Multiplication
-----
1

----Now for Addition----
Please enter the two numbers(hex).
BBABABABABAB
ABABABABABAB
Result(hex): BC5757575756

----Welcome to big number calculator----
Please choose the function
0. Exit
1. Addition
2. Subtraction
3. Multiplication
-----
5
Error! Please try again.
```

```
----Welcome to big number calculator----
Please choose the function
0. Exit
1. Addition
2. Subtraction
3. Multiplication
-----
2

----Now for Subtraction----
Please enter the two numbers(hex).
89A9A9A9A9A9A
ABABABABABAB
Result(hex): 899EEEEEEEEEEF

----Welcome to big number calculator----
Please choose the function
0. Exit
1. Addition
2. Subtraction
3. Multiplication
```

```

-----
1

----Now for Addition----
Please enter the two numbers(hex).
44444444444444444444
55555555555555555555
Result(hex): 99999999999999999999

----Welcome to big number calculator----
Please choose the function
0. Exit
1. Addition
2. Subtraction
3. Multiplication
-----
3

----Now for Multiplication----
Please enter the two numbers(hex).
44444444444444444444
4
Result(hex): 11111111111111111110

----Welcome to big number calculator----
Please choose the function
0. Exit
1. Addition
2. Subtraction
3. Multiplication
-----
1

----Now for Addition----
Please enter the two numbers(hex).
359359359359359359
359359359359359359
Result(hex): 6B26B26B26B26B26B2

----Welcome to big number calculator----
Please choose the function

```

```

-----
0. Exit
1. Addition
2. Subtraction
3. Multiplication
-----
3

----Now for Multiplication----
Please enter the two numbers(hex).
1111111111111111
1111111111111111
Result(hex): 123456789ABCDEFEDCBA987654321

----Welcome to big number calculator----
Please choose the function
0. Exit
1. Addition
2. Subtraction
3. Multiplication
-----
2

```

```

----Now for Subtraction----
Please enter the two numbers(hex).
FFFFFF
FFFFFFE
Result(hex): 1

----Welcome to big number calculator----
Please choose the function
0. Exit
1. Addition
2. Subtraction
3. Multiplication
-----
1

----Now for Addition----
Please enter the two numbers(hex).
0
0
Result(hex): 0

----Welcome to big number calculator----
Please choose the function
0. Exit
1. Addition
2. Subtraction
3. Multiplication
-----
3

----Now for Multiplication----
Please enter the two numbers(hex).
FDACCCCCCEDD
0
Result(hex): 0

----Welcome to big number calculator----
Please choose the function
0. Exit
1. Addition
2. Subtraction
3. Multiplication
-----
0
Good bye!

```

## 五、 Bonus

1. 完成度：bonus part1
2. 程式講解：

```

134 bool handle_out_of_range(string str, const int arraysize)
135 {
136     bool correct_number = true;
137     for(int i = 0; i < arraysize; i++)
138     {
139         if(str[i] == '-') // 錯誤的輸入number，回傳false
140         {
141             correct_number = false;
142             break;
143         }
144         else if((str[i] >= '0' && str[i] <= '9') || str[i] == 'A' || str[i] == 'B' || str[i] == 'C' || str[i] == 'D' || str[i] == 'E' || str[i] == 'F')
145         {
146             continue;
147         }
148         else
149         {
150             correct_number = false; // 錯誤的輸入number，回傳false
151             break;
152         }
153     }
154     return correct_number;
155 }
156

```



利用 function: `handle_out_of_range` 回傳 boolean 值進 main function 來判斷使用者輸入的大數是否為正確的輸入。

- (a) 為了減少運算時間以及記憶體，在第 140 行判斷當大數的第一個 element 為 '-' 則直接回傳 false。
- (b) 當輸入的兩個大數中，有任何一個 element 不屬於 '0'~'9' 或是 'A'、'B'、'C'、'D'、'E' 以及 'F'，則回傳 false。
- (c) 在 main function 中：

```

192 cout << endl;
193 cout << "----Now for Addition----" << endl;
194 cout << "Please enter the two numbers(hex)." << endl;
195 cin >> str1;
196 cin >> str2;
197
198 correct_range1 = handle_out_of_range(str1, str1.length());
199 correct_range2 = handle_out_of_range(str2, str2.length());
200
201 while(correct_range1 == false || correct_range2 == false)
202 {
203     cout << "Please enter the two numbers(hex) again." << endl;
204     cin >> str1;
205     cin >> str2;
206     correct_range1 = handle_out_of_range(str1, str1.length());
207     correct_range2 = handle_out_of_range(str2, str2.length());
208 }

```

當回傳的兩個 boolean 任一者為 false 時，即會判定為錯誤的大數，並要求使用者再次輸入。

- (d) 執行結果：

```

[s108601205@EDA359 ~]$ g++ -std=c++11 108601205_PA3_bonus_part1.cpp -o 108601205_PA3_bonus_part1.o
[s108601205@EDA359 ~]$ ./108601205_PA3_bonus_part1.o
----Welcome to big number calculator----
Please choose the function
0. Exit
1. Addition
2. Subtraction
3. Multiplication
-----
1

----Now for Addition----
Please enter the two numbers(hex).
ABCDEFGHIJKLMNO
1111111111
Please enter the two numbers(hex) again.
222222222
-fjialgurmvp
Please enter the two numbers(hex) again.
-4564564567845641
Fbjwfhvnuwtle;'./l?
Please enter the two numbers(hex) again.
888888888888888A
1111111111111111
Result(hex): 999999999999999B

```

## 六、 難處發現與解決

### 1. 問題 1：

減法的退位計算，雖然知道有退位時，需要從較大的位數取數補入，但會印出多個負數，且結果差很多。

#### 問題解決：

設立幾個大數，用手計算結果後，順利推得減法運算的程式操作，另外也透過多次的嘗試才找到程式運行的順序。

### 2. 問題 2：

乘法需要將 element 相乘後存在 array 中正確的位置，原本想說將 string 轉成 10 進制的整數再進行運算，但程式上較為複雜，且不符合題目需求。

#### 問題解決：

利用巢狀的 for statement 並加上一個 if 的判斷式進行 continue 的動作，最後用類似加法運算的方式得到正確的 answer。

## 七、 回饋

這次的 PA 在編寫的時候有很多邏輯順序需要注意，一開始在寫 pseudocode 時沒有想太多，導致第一次將 code 打出來時，程式出現一堆奇怪的結果。也由於一開始心急想要將 PA 快點完成，所以在 debug 時花了很多時間，最後索性直接將 code 刪掉，認真思考過後再重打。

此外透過這次的 PA 操作到工作站，一開始操作時很不習慣，但整個操作過程幾乎只用鍵盤而鮮少用滑鼠，這種體驗讓人有點上癮。