**演算法程式作業一**

資工2B 林晉宇 110403518

1. **大數相乘**

本次作業實作大數相乘的兩種方法，分別為傳統作法及Karatsuba優化版本，並用python產生的測資拿來測試運行時間，並做比較，找出超車點為第幾位。

1. **傳統作法**

傳統的部分，我使用模擬直式乘法，每一位相乘完後相加。

* + **資料結構:** 使用兩個字串代表乘數與被乘數；另開一個陣列儲存乘積的每一位數字。
  + **時間複雜度:** O(n2)
  + **Pseudo code:**

***string n,m <- input;***

***int num\_dig;***

***int product[10001]={0}***

***for(i = n.length()-1 ~ 0){***

***for(j = m.length()-1 ~ 0){***

***int ia=n.length()-1-i,ib=m.length()-1-j;***

***product[ia+ib] += n[i]\*m[j];***

***num\_dig = max(num\_dig, ia+ib);***

***if(product[ia+ib]>=10){***

***product[ia+ib+1]+=product[ia+ib]/10;***

***product[ia+ib]%=10;***

***num\_dig=max(num\_dig,ia+ib+1);***

***}***

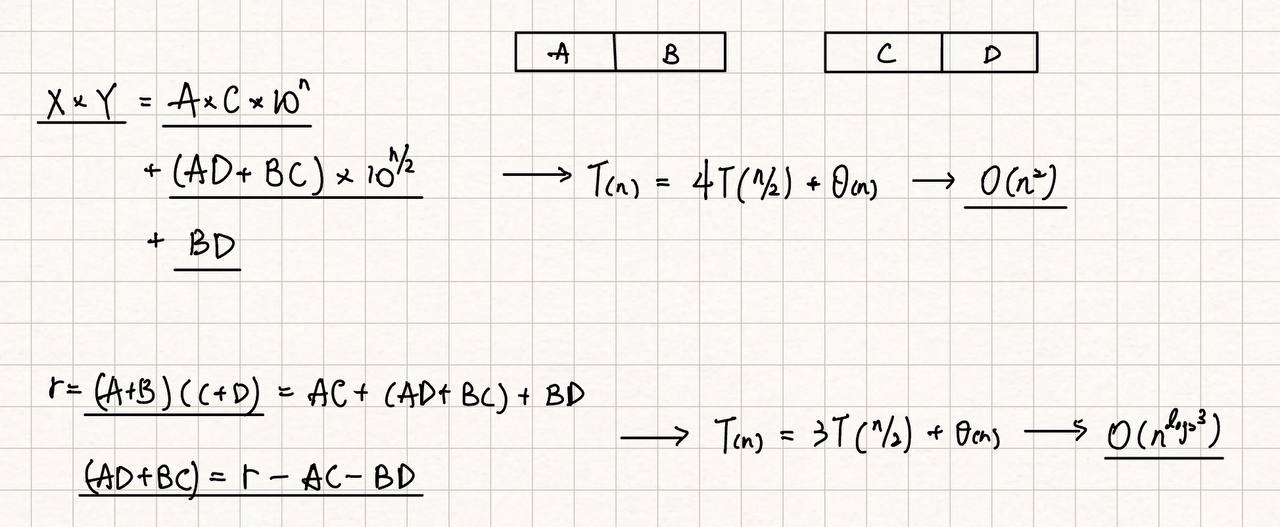
***}***

***}***

1. **Karatsuba乘法**

Karatsuba使用上課教的方法，可以將時間複雜度降低。

* + **時間複雜度:**



* + **資料結構:** 一樣使用兩個字串去儲存乘數以及被乘數。
  + **函式:** 包含大數的 相乘、相減、相加、平移、補0這些function。
  + **Pseudo code:**

***string karatsuba(string x,string y)***

***{***

***if(x.length()==1 && y.length()==1)***

***return x[0]\*y[0];***

***if(x.length()>y.length())   y=appendZero();***

***else    x=appendZero();***

***if(x.length()%2)***

***{***

***x = "0" + x;***

***y = "0" + y;***

***}***

***string a,b,c,d,ac,bd,abcd; //abcd = (a+b) \* (c+d)***

***int n = x.length();***

***a = x.substr(0,n/2);***

***b = x.substr(n/2);***

***c = y.substr(0, n/2);***

***d = y.substr(n/2);***

***ac = karatsuba(a,c);***

***bd = karatsuba(b,d);***

***abcd = karatsuba(addXY(a,b),addXY(c,d));***

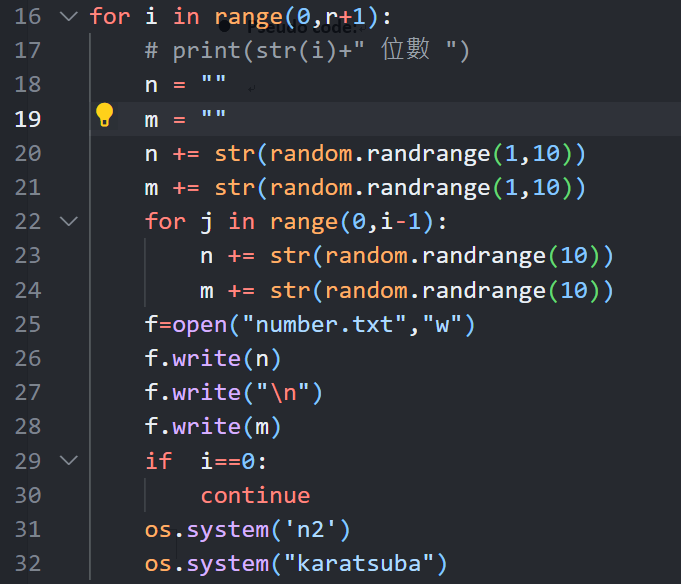
***return addXY(addXY(mul10(ac, n),mul10(subXY(subXY(abcd,ac), bd), n/2)), bd);***

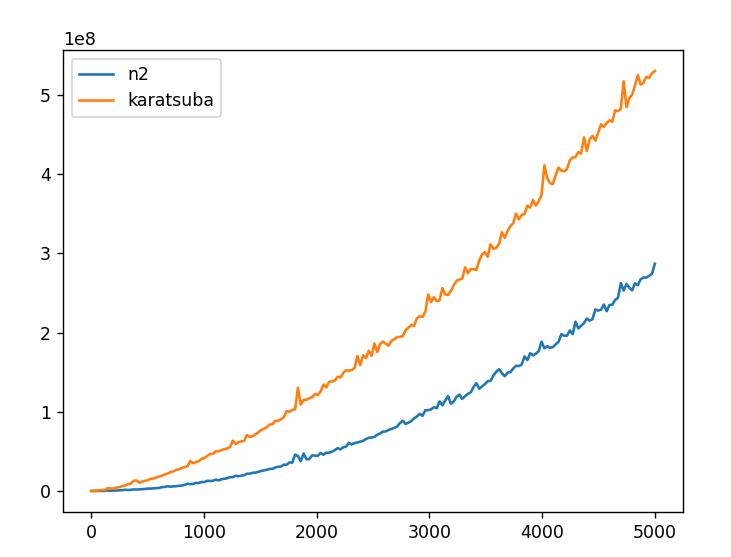
***}***

1. **比較**

使用python產生測資，並將每次執行兩種方法所花的時間記錄下來，透過圖表呈現出來。

* + **設計測資:** 範圍從1~5000位，每20位產生一比測資(用python random產生隨機數)，並丟進兩個C++程式去執行，計算執行時間，在透過圖表呈現(matplotlib)。
  + **Code: (隨機產生數字)**

****

* + **圖表:** 以下為實作出來的圖表，可以看到karatsuba跑出來的時間皆大於傳統方法，並不符合理論結果；然而我改進了許多次karatsuba的實作方式，也從網路上測試不同的程式，跑出來的結果皆慢於傳統方法。

1. **結論**

此次作業的實驗結果不是很理想，就理論來看的話，karatsuba應該在位數越大的情況下，執行時間相比傳統做法要快得多，但實作出來的結果卻是karatsuba比傳統方法慢好幾倍。

我的猜想是有可能我實作karatsuba的加、減、平移或是補0的function太慢(然而我不確定如何能改進更快)，雖然這些function應該都是O(n)級別的，但我的程式可能跑出來的常數十分大，導致其增長速度比傳統快很多。

Threshold的部分，如果以傳統方法為O(n^2)，karatsuba為O(3n^lg3)的情況去推斷的話，我認為理想值應該會落在13到20之間，然而由於我的實驗結果為傳統方法皆快於karatsuba，所以並沒有一個明確的臨界點，十分抱歉。