**ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH**

**TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN**

**KHOA CÔNG NGHỆ THÔNG TIN**

****

**HOÀNG DÂN AN – 1612001**

**NGUYỄN PHƯỚC AN – 1612009**

**NGUYỄN HOÀNG ANH – 1612023**

**TRƯƠNG THÀNH DANH - 1612085**

**HOÀNG HẢI GIANG – 1612154**

**ĐỒ ÁN GIỮA KỲ**

**BIỂU DIỄN VÀ TÍNH TOÁN**

**SỐ HỌC TRÊN MÁY TÍNH**

**Lớp: Kiến trúc máy tính & Hợp ngữ 16\_1**

Thành phố Hồ Chí Minh – 2017

**ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH**

**TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN**

**KHOA CÔNG NGHỆ THÔNG TIN**

****

**HOÀNG DÂN AN – 1612001**

**NGUYỄN PHƯỚC AN – 1612009**

**NGUYỄN HOÀNG ANH – 1612023**

**TRƯƠNG THÀNH DANH - 1612085**

**HOÀNG HẢI GIANG – 1612154**

**ĐỒ ÁN GIỮA KÌ**

**Đề tài**

**Biểu diễn và tính toán số học trên máy tính**

**Giảng viên hướng dẫn**

**Lê Viết Long**

**Lớp: Kiến trúc máy tính & Hợp ngữ 16\_1**

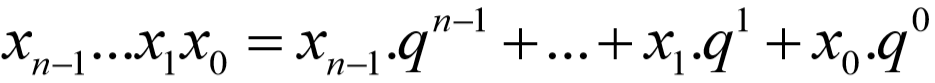
Thành phố Hồ Chí Minh – 2017

**LỜI CẢM ƠN**

**MỤC LỤC**

1. **SƠ LƯỢC VỀ BIỂU DIỄN SỐ NGUYÊN**
2. Hệ cơ số q tổng quát

* Tổng quát số nguyên có n chữ số thuộc hệ cơ số q bất kỳ được biểu diễn:



(mỗi chữ số xi lấy từ tập X có q phần tử )

* Ví dụ:

– Hệ cơ số 10: A = 123 = 100 + 20 + 3 = 1.102 + 2.101 + 3.100

– q = 2, X = {0, 1}: hệ nhị phân (binary)

– q = 8, X = {0, 1, 2,…, 7}: hệ bát phân (octal)

– q = 10, X = {0, 1, 2,…, 9}: hệ thập phân (decimal)

– q = 16, X = {0, 1, 2,…,9, A, B,…, F}: hệ thập lục phân (hexadecimal)

* Chuyển đổi: A = 123 d = 01111011 b = 173 o = 7B h
* Hệ cơ số thường được biển diễn trong máy tính là hệ cơ số 2

1. Chuyển đổi giữa các hệ cơ số
2. Từ hệ thập phân (10) sang hệ nhị phân (2)

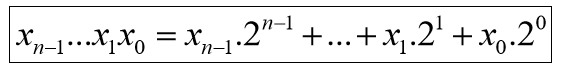
* Lấy số cơ số 10 chia cho 2 🡪 Số dư đưa vào kết quả 🡪 Số nguyên đem chia tiếp cho 2 🡪 Quá trình lặp lại cho đến khi số nguyên = 0

1. Từ hệ thập phân (10) sang hệ thập lục phân (16)

* Lấy số cơ số 10 chia cho 16 🡪 Số dư đưa vào kết quả 🡪 Số nguyên đem chia tiếp cho 16 🡪 Quá trình lặp lại cho đến khi số nguyên = 0

1. Từ hệ nhị phân (2) sang hệ thập phân (10)

* Khai triển biểu diễn và tính giá trị biểu thức

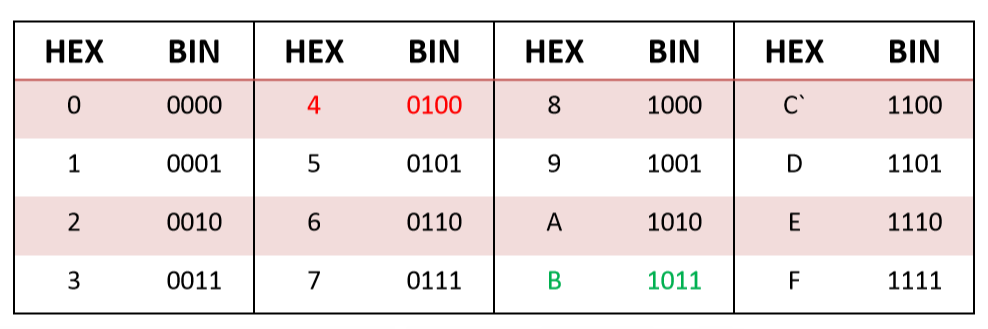


1. Từ hệ nhị phân (2) sang hệ thập lục phân (16)

* Nhóm từng bộ 4 bit trong biểu diễn nhị phân rồi chuyển sang ký số tương ứng trong hệ thập lục phân (0000 🡪 0,…, 1111 🡪 F)

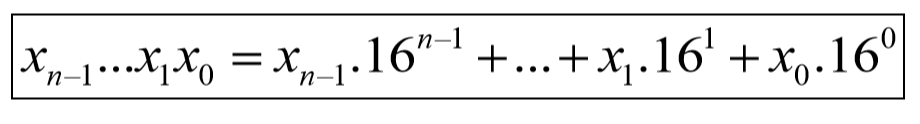
1. Từ hệ thập lục phân (16) sang hệ nhị phân (2)

* Sử dụng bảng dưới đây để chuyển đổi

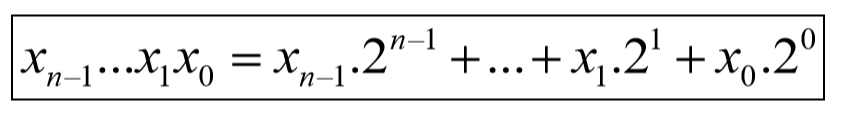


1. Từ hệ thập lục phân (16) sang hệ thập phân (10)

* Khai triển biểu diễn và tính giá trị biểu thức



1. Hệ nhị phân



* Được dùng nhiều trong máy tính để biểu diện các giá trị lưu trong các thanh ghi hoặc trong các ô nhớ. Thanh ghi hoặc ô nhớ có kích thước 1byte (8 bit) hoặc 1 word (16 bit).
* n được gọi là chiều dài bit của số đó
* Bit trái nhất xn-1 là bit có giá trị (nặng) nhất MSB (Most Significant Bit)
* Bit phải nhất x0 là bit ít giá trị (nhẹ) nhất LSB (Less Significant Bit)

1. Số nguyên có dấu

* Lưu các số dương hoặc âm (số có dấu) có 4 cách phổ biến:

– Dấu lượng

– Bù 1

– Bù 2

– Số quá (thừa) K

 - Số có dấu trong máy tính được biểu diễn ở dạng số bù 2

1. Dấu lượng

* Bit trái nhất (MSB): bit đánh dấu âm / dương

– 0: số dương

– 1: số âm

* Các bit còn lại: biểu diễn độ lớn của số (hay giá trị tuyệt đối của số)
* Ví dụ:

– Một byte 8 bit: sẽ có 7 bit (trừ đi bit dấu) dùng để biểu diễn giá trị tuyệt đối cho các số có giá trị từ 0000000 (010) đến 1111111 (12710)

* Ta có thể biểu diễn các số từ −12710 đến +12710
* -N và N chỉ khác giá trị bit MSB (bit dấu), phần độ lớn (giá trị tuyệt đối) hoàn toàn giống nhau

1. Bù 1

* Tương tự như phương pháp dấu lượng, bit MSB dùng làm bit dấu

– 0: Số dương

– 1: Số âm

* Các bit còn lại để biểu diễn giá trị (\*)
* Số âm: thực hiện tất cả các phép đảo bit của (\*)

1. Bù 2

* Biểu diễn giống như số bù 1 + ta phải cộng thêm số 1 vào kết quả (dạng nhị phân)
* Số bù 2 ra đời khi người ta gặp vấn đề với hai phương pháp dấu lượng và bù 1, đó là:

– Có hai cách biểu diễn cho số 0 (+0 và -0) 🡪 không đồng nhất

– Bit nhớ phát sinh sau khi đã thực hiện phép tính phải được cộng tiếp vào kết quả 🡪 dễ gây nhầm lẫn 🡪 Phương pháp số bù 2 khắc phục hoàn toàn 2 vấn đề đó

1. Số quá k

* Còn gọi là biểu diễn số dịch (biased representation)
* Chọn một số nguyên dương K cho trước làm giá trị dịch
* K từ -2127 đến 2127
* Biểu diễn số N:

+N (dương): có được bằng cách lấy K + N, với K được chọn sao cho tổng của K và một số âm bất kỳ trong miền giá trị luôn luôn dương

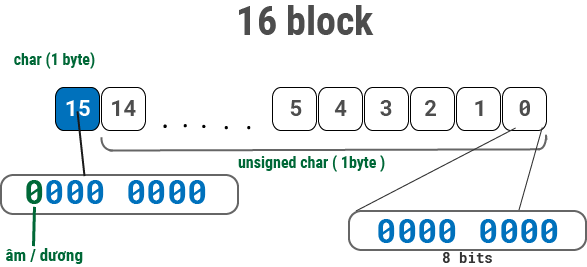
-N (âm): có được bằng cáck lấy K - N (hay lấy bù hai của số vừa xác định)

1. **THUẬT TOÁN VÀ KIỂM TRA TRƯỜNG HỢP**
2. Mô tả cách thức thuật toán:
3. Nguyên lý lưu thông tin số nguyên lớn của đồ án :

- Chia 128 bits dữ liệu thành 1 mảng 16 block độ lớn 1 byte (unsigned char)

- Riêng block cao nhất được định nghĩa là [char] để có thể dễ nhận dạng số âm.

- Bit cao nhất (tức là bit cao nhất của block thứ 15) dùng để đánh dấu số âm.

- Lưu số nguyên theo định nghĩa số bias bù 2.  


1. Chuyển chuỗi thành số nguyên lớn (16 bytes) :  
   (\*) Chuỗi ký tự bit  
   - Ví dụ: “10101011”, “00111001”, ….  
   - Mỗi kí tự có 2 giá trị là ‘0’ và ‘1’

* Ý tưởng :
  + Kí tự thứ n trong chuỗi sẽ là giá trị của bit thứ n trong số nguyên
  + Do ta chia số nguyên thành 16 block nên ta cần phải tính toán vị trí bit trong số ứng
  + Sử dụng các phép logic để gán bit đó vào vị trí cụ thể của block.

Bit = ChuoiBin[len - Position - 1];

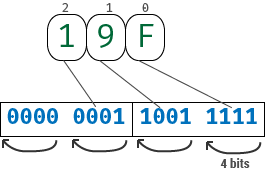
Vị\_Tri\_Block = Position / 8;

Vi\_Tri\_Trong\_Block = Position % 8;

//Su dung phep toan ^ de gan bit vao 1 so

Block[Vi\_Tri\_Block] = Block[Vi\_Tri\_Block] ^ (Bit Vi\_Tri\_Trong\_Block);

1. Chuỗi kí tự hex :

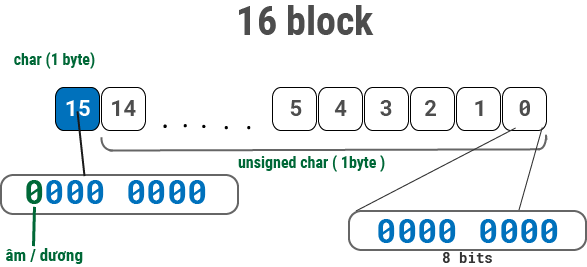
* Ví dụ : ”00FFAA”, “13B”, …
* Mỗi kí tự là một số tương ứng trong hệ số thập lục phân (1,2,3,4,5,6,7,8,9,A,B,C,D,E,F)  
  
* Ý tưởng giống với chuyển chuỗi kí tự bit nhưng :
  + Giá trị ở hệ số n trong chuỗi sẽ tương ứng với 4 bits thứ n trong số nguyên 16 bytes.
  + Sử dụng các phép logic để gán 4 bit vào vị trí cụm 4 bit thứ n.

1. Mô tả cách thức thuật toán :.
2. Nguyên lý lưu thông tin số nguyên lớn của đồ án :

- Chia 128 bits dữ liệu thành 1 mảng 16 block độ lớn 1 byte (unsigned char)

- Riêng block cao nhất được định nghĩa là [char] để có thể dễ nhận dạng số âm.

- Bit cao nhất (tức là bit cao nhất của block thứ 15) dùng để đánh dấu số âm.

- Lưu số nguyên theo định nghĩa số bias bù 2.  


1. Chuyển chuỗi thành số nguyên lớn (16 bytes) :  
   (\*) Chuỗi ký tự bit  
   - Ví dụ: “10101011”, “00111001”, ….  
   - Mỗi kí tự có 2 giá trị là ‘0’ và ‘1’

* Ý tưởng :
  + Kí tự thứ n trong chuỗi sẽ là giá trị của bit thứ n trong số nguyên
  + Do ta chia số nguyên thành 16 block nên ta cần phải tính toán vị trí bit trong số ứng
  + Sử dụng các phép logic để gán bit đó vào vị trí cụ thể của block.

**bit** = chuoiBin[len – position - 1];

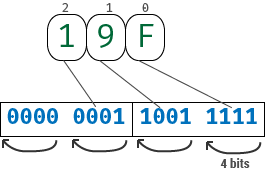
vi\_tri\_block = position / 8;

vi\_tri\_trong\_block = position % 8;

//su dung phep toan ^ de gan bit vao 1 so

block[vi\_tri\_block] = block[vi\_tri\_block] ^ (**bit** << vi\_tri\_trong\_block);

1. Chuỗi kí tự hex :

* Ví dụ : ”00FFAA”, “13B”, …
* Mỗi kí tự là một số tương ứng trong hệ số thập lục phân (1,2,3,4,5,6,7,8,9,A,B,C,D,E,F)  
  
* Ý tưởng giống với chuyển chuỗi kí tự bit nhưng :
  + Giá trị ở hệ số n trong chuỗi sẽ tương ứng với 4 bits thứ n trong số nguyên 16 bytes.
  + Sử dụng các phép logic để gán 4 bit vào vị trí cụm 4 bit thứ n.

**Hex** = chuoiHex**[**position**];**

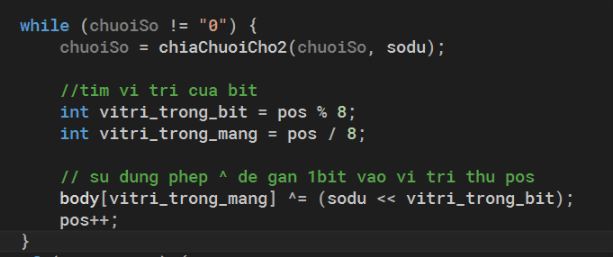
Vi\_tri\_block = position / 2;

Vi\_tri\_trong\_block = position % 2;

Block**[**vi\_tri\_block**]** = block**[**vi\_tri\_block**]** ^ (**hex** << 4\*(vi\_tri\_trong\_block));

1. Chuỗi thập phân :

* Ví dụ :”12345”,”93”, …
* Mỗi kí tự là số 0 – 9.
* Do sơ cấp 10 không là bội ước của 255, nên việc chia thành từng dãy bits không dễ dàng. Cách duy nhất chúng ta có thể làm được là “chia 2” chuỗi số dần để lấy số dư – là bit thứ 2n của chuỗi số đó.
* Ý tưởng :
  + Thiết lập thuật toán “chia 2” chuỗi số và xuất ra số dư
  + Chia dần chuỗi số, tìm số dư
  + Gán bit (là số dư tìm được) vào vị trí tăng dần khi chia.



Trong hình: **body** là *dãy block*, **pos** là vị trí đếm từ 0 lên.

1. Hàm chuyễn đổi thập phân xang nhị phân :  
   - Đem chuỗi ***chia*** 2 lấy dư

* Số dư đưa vào kết quả
* Qúa trình lập lại cho đến khi số nguyên bằng 0.

1. Hàm chuyễn đổi nhị phân xang thập phân :

* Chuỗi kết quả ban đầu là 0
* Số x là vị trí bit cao nhất (khác bit dấu)
* Đem chuỗi nhân 2 công giá trị bit thứ x
* Giảm x xuống 1 đơn vị
* Lập lại cho đến khi x bằng 0.

1. Hàm chuyễn đổi nhị phân xang thập lục phân :

* Chọn 4 bit từ 8 bit của 1 ô rồi so sánh với bảng đã biết

1. Toán tử operator “ + ”

* Đem block thứ i của số A cộng cho block thứ i của số B
* Phần tràn đem cộng vào block i + 1 của kết quả
* Cộng hết các block tương ứng với nhau
* Do ta sử dụng nguyên tắc số bias thì phần tràn lớn nhất sẽ bị bỏ đi.

1. Toán tử operator” - ”

* Sử dụng nguyên tắt A – B = A + ( - B )
* Sử dụng lại toán tử cộng.

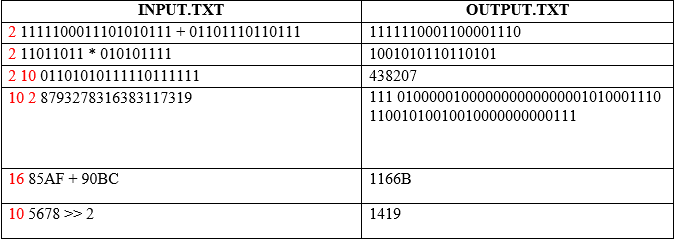
1. Toán tử operator” \* ”
2. Toán tử operator” / ”
3. Toán tử “&”, “|”, “^”:

* Đem từng block thực hiện phép toán theo phép toán tương ứng

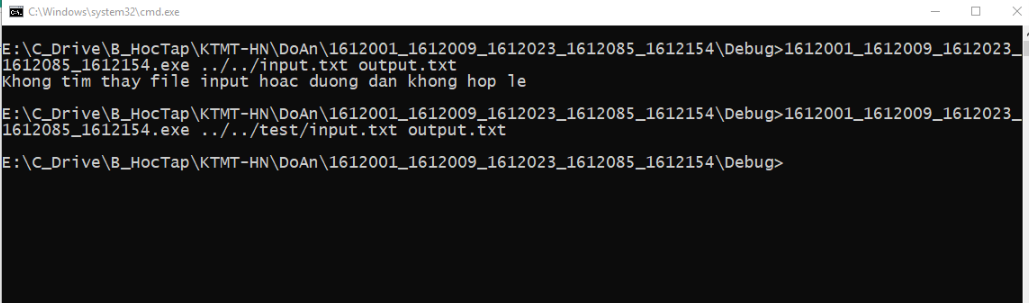
1. Toán tử NOT “~”

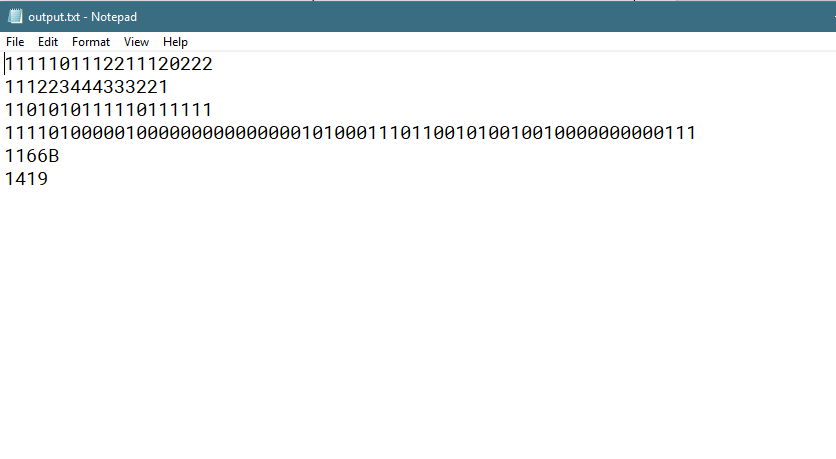
* Đem tất cả các block XOR với 0xFF (1111 1111).

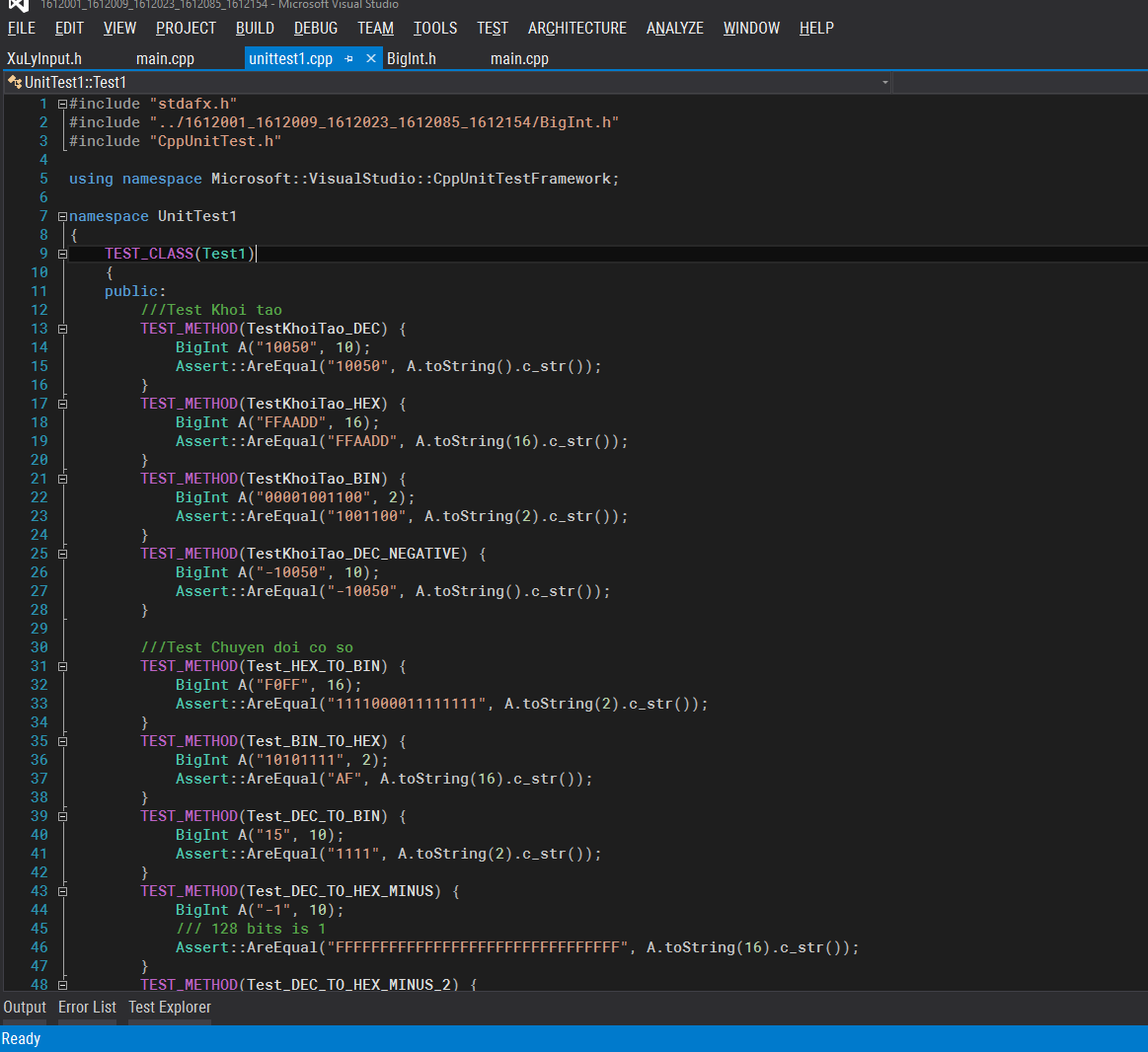
1. Kiểm tra trường hợp đề bai :
   1. Test theo ví dụ đề bài :

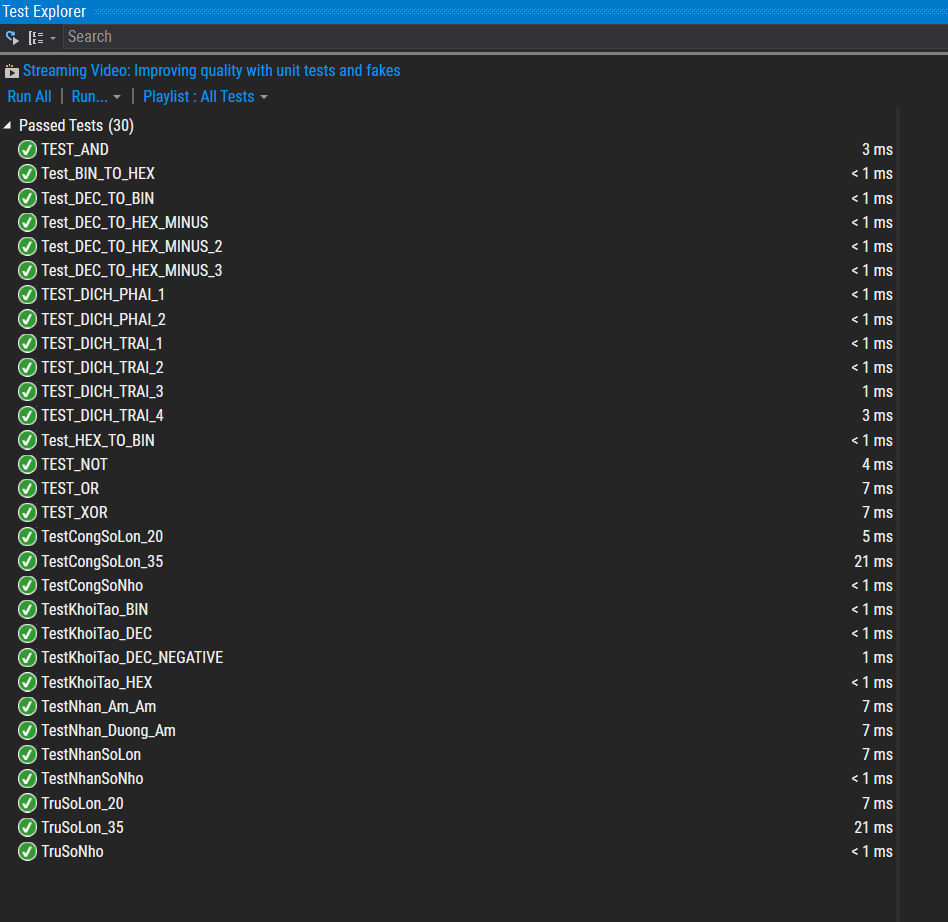


* Test Output :





* 1. **Test toàn diện:**
* Sử dụng framework [Unit Test](https://msdn.microsoft.com/en-us/library/hh598953.aspx) có sẵn của Visual Studio để test các chức năng và phương thức của class BigInt.
* Các hình thức test:
  + Test Khởi tạo
  + Test phép toán đại số
  + Test phép toán logic
  + Test các bộ số lớn
* Nguồn kết quả tin cậy:
  + Các bộ test sẽ tạo dựa trên các kết quả trên trang web <https://defuse.ca/big-number-calculator.htm>
  + Các bộ test tự cho, lấy kết quả để tạo bộ test
* Screenshots



1. **ĐÁNH GIÁ**

|  |  |  |
| --- | --- | --- |
| **STT** | **Mục đánh giá** | **Mức độ hoàn thành** |
| **1** | Lưu trữ trên đủ 16 bytes | 100% |
| **2** | Nhập và xuất | 90% |
| **3** | Các phép toán đại số | 100% |
| **4** | Các phép toán logic | 100% |
| **5** | Chuyển đổi cơ số | 100% |
| **6** | Dịch trái dịch phải | 100% |
| **7** | Biểu diễn được số âm | 100% |
| **8** | Xử lý lỗi logic và đại số | 10% |
| **9** | Xử lý đọc ghi file bằng command line | 100% |
| **10** | Source code và báo cáo rõ ràng, dễ hiểu | 80% |
| **Tổng cộng** | | **88%** |

**TÀI LIỆU THAM KHẢO**

* + [**Slide biểu diễn số nguyên**](Ch02_Bieu-dien-so-nguyen.pdf)
  + [**Stackoverflow**](https://stackoverflow.com)