

|  |
| --- |
| BÁO CÁO  ĐỒ ÁN 1 |
| KHẢO SÁT SỐ |
| March 24  MÔN: KIẾN TRÚC MÁY TÍNH VÀ HỢP NGỮ |

|  |  |
| --- | --- |
| NỘI DUNG   |  | | --- | | 1. Thành viên nhóm ……………………………………………………………………….…… Trang 3 2. Bảng đánh giá ……………………………………………………………………………….… Trang 4 3. Môi trường lập trình ………………………………………………………………………. Trang 5 4. Ý tưởng thực hiện …………………………………………………………………..……… Trang 5    1. Số nguyên lớn ……………………………………………………………………….…. Trang 5    2. Số chấm động chính xác cao ……………………………………………….…….. Trang 7 5. Kết quả chạy chương trình …………………………………………………………….. Trang 10   5.1 Chạy trên visual studio …………………………………………………………….. Trang 10  5.2 Chạy trên Command Line …………………………………………………………. Trang 12   1. Nhận xét chung ………………………………………………………………………………. Trang 13 2. Tài liệu tham khảo …………………………………………………………………………...Trang 13 | |
|  |
|  |

Thành viên nhóm

1

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Họ và tên** | **MSSV** | **Nhiệm vụ** |
| 1 | Nguyễn Phúc Khôi Nguyên | 1712106 | Viết các chức năng: 11,2,7,13,14 |
| 2 | Trịnh Tấn Tài | 1712147 | Viết các chức năng: 3,5,6,7,9. Chạy test chương trình. Làm video báo cáo. |
| 3 | Hà Duy Tân | 1712148 | Viết các chức năng: 3,4,5,6,10 |
| 4 | Nguyễn Huỳnh Minh Thuận | 1712175 | Viết các chức năng: 6,8, 12,14,16. Phụ viết báo cáo phần QFloat |
| 5 | Võ Trọng Phúc | 1712126 | Viết các chức năng: 1,2,3 7,9. Viết báo cáo |

Bảng đánh giá mức độ hoàn thành

2

|  |  |  |
| --- | --- | --- |
| STT | Yêu cầu | Đánh giá |

|  |  |  |
| --- | --- | --- |
| *Số nguyên lớn* | | |
| 1 | Hàm Nhập: void ScanQInt(QInt &x) | 100/100 |
| 2 | Hàm xuất: void PrintQInt( QInt x) | 100/100 |
| 3 | Hàm chuyển đổi số QInt thập phân sang nhị phân: bool \* DecToBin (QInt x) | 100/100 |
| 4 | Hàm chuyển đổi số QInt nhị phân sang thập phân: QInt BinToDec(bool \*bit) | 100/100 |
| 5 | Hàm chuyển đổi số QInt nhị phân sang thập lục phân: char \*BinToHex(bool \*bit) | 100/100 |
| 6 | Hàm chuyển đổi số QInt thập phân sang thập lục phân: char \*DecToHex(QInt x) | 100/100 |
| 7 | Các operator toán tử : “+”, “-” , “\*”, “/” | 100/100 |
| 8 | Các toán tử so sánh và gán: “”, “==”, “<=”, “>=”, “=” | 100/100 |
| 9 | Các toán tử: AND “&”, OR “|”, XOR “^”, NOT “~” | 100/100 |
| 10 | Các toán tử: dịch trái “<>”, xoay trái: “rol”, xoay phải: “ror” | 100/100 |

|  |  |  |
| --- | --- | --- |
| *Số chấm động chính xác cao* | | |
| 11 | Hàm Nhập: void ScanQfloat (Qfloat &x) | 100/100 |
| 12 | Hàm xuất: void PrintQfloat( Qfloat x) | 100/100 |
| 13 | Hàm chuyển đổi số Qfloat nhị phân sang thập phân: Qfloat BinToDec(bool \*bit) | 95/100 |
| 14 | Hàm chuyển đổi số Qfloat thập phân sang nhị phân bool \*DecToBin(Qfloat x) | 100/100 |

Môi trường lập trình:

3

* + Ngôn ngữ C++
  + IDE: Visual Studio 2015, Visual Studio 2017

4

Ý tưởng thực hiện:

1. Số nguyên lớn:

*Phạm vi biểu diễn:* -2127 -> 2127 – 1

- Với yêu cầu kiểu dữ liệu QInt có độ lớn 16 bytes (128 bits), dùng một mảng BigInt kiểu int 4 phần tử để biểu diễn , với mỗi BigInt[i] (i:0->3) chứa 1 số hệ Decimal thể hiện cho 32 bits.

- QInt được xây dựng thành 1 class gồm:

+ Thuộc tính: Mảng BigInt[4]

+ Các phương thức: những hàm tính toán theo yêu cầu cùng các hàm bổ trợ.

* Chuyển đổi từ hệ 10 -> hệ 2: chia dần số thập phân cho 2 cho đến khi kết  
  quả bằng 0, mỗi lần chia lấy số dư phép chia gán cho bit theo thứ tự.  
  Nếu là số âm tìm dạng bù 2 tức đảo bit kết quả nhận được rồi +1 (ở dạng biểu diễn QInt).
* Chuyển đổi từ hệ 2 -> hệ 10: kiểm tra bit đầu của dãy bit là 1 (âm) hay 0 (dương). Nếu là số âm thì chuyển sang dạng bù 2. Chuyển dãy bit sang hệ 10 bằng hàm bổ trợ BatBit(i) (nếu i =0 , trừ bit dấu) . Nếu là số âm thì thêm vào dấu -. Chuyển sang số hệ 10 sang QInt rồi trả 1 QInt kết quả.
* Chuyển đổi từ hệ 2 -> hệ 16: với mỗi kí tự trong số hệ 16, chuyển thành 4  
  bit tương ứng trong hệ nhị phân rồi gán vào giá trị của số. Lặp lại cho  
  đến hết.
* Chuyển đổi từ hệ 16 -> hệ 2: nhóm 4 bit trong dạng biểu diễn nhị phân rồi chuyển sang số hệ 16 tương ứng. Lặp lại đến hết.
* Toán tử gán = : set mỗi bit i (i : 0,127) của QInt a bằng mỗi bit I của QInt b. Lấy giá trị bit tại i của b bằng toán tử [].
* Toán tử cộng +: chuyển 2 số ban đầu về dạng hệ 2 (dãy bit 128 phần tử dạng string). Dùng hàm addBinary cộng 2 dãy bit dưới dạng string được dãy bit kết quả, chuyển sang dạng QInt rồi trả về 1 QInt mới là tổng cần tính.
* Toán tử trừ -: Coi a – b như a + (-b). Chuyển b về thành dãy bit bù 2. Sử dụng toán tử + đê cộng a với bù 2 của b.
* Toán tử nhân \*: Để tính Q \* M, thêm bit 0 vào cuối Q, khởi tạo A là dãy bit 0. Thực hiện lặp 128 lần (độ dài dãy bit).

Với mỗi vòng lặp nếu 2 bit cuối của Q là 10 thì gán A = A – M, 01 thì gán A = A + M, còn lại thì để nguyên A. Sau đó ghép 2 dãy bit A, Q và dịch phải 1 lần, cập nhật lại A, Q sau khi lặp.

Sau 128 vòng lặp bỏ bit cuối của Q, trả về Q là kết quả phép nhân.

Nhận biết tràn số bằng cách xét dấu 2 thừa số và tích.

* Toán tử chia /: Để tính Q \* M, thêm bit 0 vào cuối Q, khởi tạo A là dãy bit 0. Thực hiện lặp 128 lần (độ dài dãy bit).

Với mỗi vòng lặp nếu 2 bit cuối của Q là 10 thì gán A = A – M, 01 thì gán A = A + M, còn lại thì để nguyên A. Sau đó ghép 2 dãy bit A, Q và dịch phải 1 lần, cập nhật lại A, Q sau khi lặp.

Sau 128 vòng lặp bỏ bit cuối của Q, trả về Q là kết quả phép nhân.

Nhận biết tràn số bằng cách xét dấu 2 thừa số và tích.

* Toán tử AND &: chuyển 2 số về hệ nhị phân, thực hiện toán tử AND các bit tương ứng .
* Toán tử OR |: chuyển 2 số về hệ nhị phân, thực hiện toán tử ORcác bit tương ứng .
* Toán tử XOR ^: chuyển 2 số về hệ nhị phân, thực hiện toán tử XOR các bit tương ứng .
* Toán tử NOT ~: chuyển số đó về hệ nhị phân, thực hiện toán tử NOT các bit của số đó và gán vào bit của số kết quả.
* Phép dịch trái <<: thực hiện trên hệ 2, dịch lần lượt từng bit sang bên trái 1 bit rồi gán giá trị ở bit[0] = 0, lặp lại n lần (số bit dịch).
* Phép dich phải >> :thực hiện trên hệ 2, dịch lần lượt từng bit sang bên phải 1 bit rồi gán giá trị ở bit[0] = 0, lặp lại n lần (số bit dịch).
* Phép toán ROL: thực hiện trên hệ 2, cắt dãy bit ở đầu từ vị trí i = 0 đến i = n (số bit ROL) rồi ghép vào cuỗi dãy bit vừa cắt. Chuyển sang dạng QInt trả về kết quả.
* Phép toán ROR: thực hiện trên hệ 2, cắt dãy bit ở cuối từ vị trí i = 128 - n đến i = 128 (số bit ROL) rồi ghép vào đầu dãy bit vừa cắt. Chuyển sang dạng QInt trả về kết quả.
* Dùng hàm bổ trợ remove\_zero() để xóa các số 0 ở đầu kết quả hệ 2,10,16 trước khi xuất ra file kết quả.

1. Số chấm động chính xác cao:

*Phạm vị biểu diễn:* -2127 -> 2127 – 1

* Chuyển từ QFloat nhị phân sang thập phân:

Khởi tạo 3 biến kiểu string sign (lưu dấu), exponent (lưu bit phần Exponent), fraction (lưu bit fraction)

+ Bước 1: Chuyển exponent về Decimal

+ Bước 2: Tìm số mũ E

+ Bước 3: Nếu số mũ E >= 0 , Số Thập Phân có dạng ±N,…. (N≠0)

Bước 3.1: Tìm nhị phân của phần nguyên rồi chuyển về Decimal

Bước 3.2: Tìm nhị phân của phần thập phân rồi chuyển về Decimal

Bước 3.3: Return kết quả theo yêu cầu đề bài. KẾT THÚC (Không cần làm bước 4)

+ Bước 4: Nếu số mũ E < 0, Số Thập Phân có dạng ±0,…….

Bước 4.1: Không cần lấy phần nguyên, tìm nhị phân của thập phân rồi chuyển về Decimal

Bước 4.2: Return kết quả theo yêu cầu đề bài. KẾT THÚC

* Chuyển từ QFloat thập phân sang nhị phân:

+ Bước 1: Xác định dấu của số thập phân (Nếu âm bit đầu là 1, dương bit đầu là 0)

+ Bước 2: Lấy phần nguyên của số thập phân

+ Bươc 3: Kiểm tra phần nguyên của số thập phân

+ Bước 4: Nếu phần nguyên là số ≠ 0 (tức số thập phân khác 0,…)

\_ Bước 4.1: Xác định số chữ số phần thập phân để khi chuyển về dạng ±1.F\*2E phần fraction có đủ 112bit

\_ Bước 4.2: Chuyển phần thập phân về dạng nhị phân bằng cách nhân 2 phần thập phân rồi kiểm tra phần thập phân sau khi nhân . Nếu >= 1 thì them bit 1, < 0 thì thêm bit 0

Ví dụ: 0.625

0.625 \* 2= 1.25 >= 1 -> bit 1

(1.25-1) \* 2= 0.5 < 1 -> bit 0

0.5 \* 2 = 1 >=1 -> bit 1

=> Nhị phân của 0.625 là 101

\_ Bước 4.3: Đưa về dạng ±1.F\*2E

\_ Bước 4.4: Tìm số mũ E và F

\_ Bước 4.5: Chuyển Exponent về nhị phân, Thêm 0 sau Fraction nếu chưa đủ 112 bit

\_ Bước 4.6: Return kết quả. KẾT THÚC

+ Bước 5: Nếu phần nguyên là số = 0 (tức số thập phân 0,…)

\_ Bước 5.1: Lấy chuỗi số phần thập phân

\_ Bước 5.2: Xác định số chữ số phần thập phân để khi chuyển về dạng ±1.F\*2E phần fraction có đủ 112bit

\_ Bước 5.3: Dùng vòng lặp để chuyển phần thập phân về nhị phân

\_Bước 5.4: GIỐNG BƯỚC 4.3 TRỞ VỀ SAU

Kết quả chạy chương trình

5

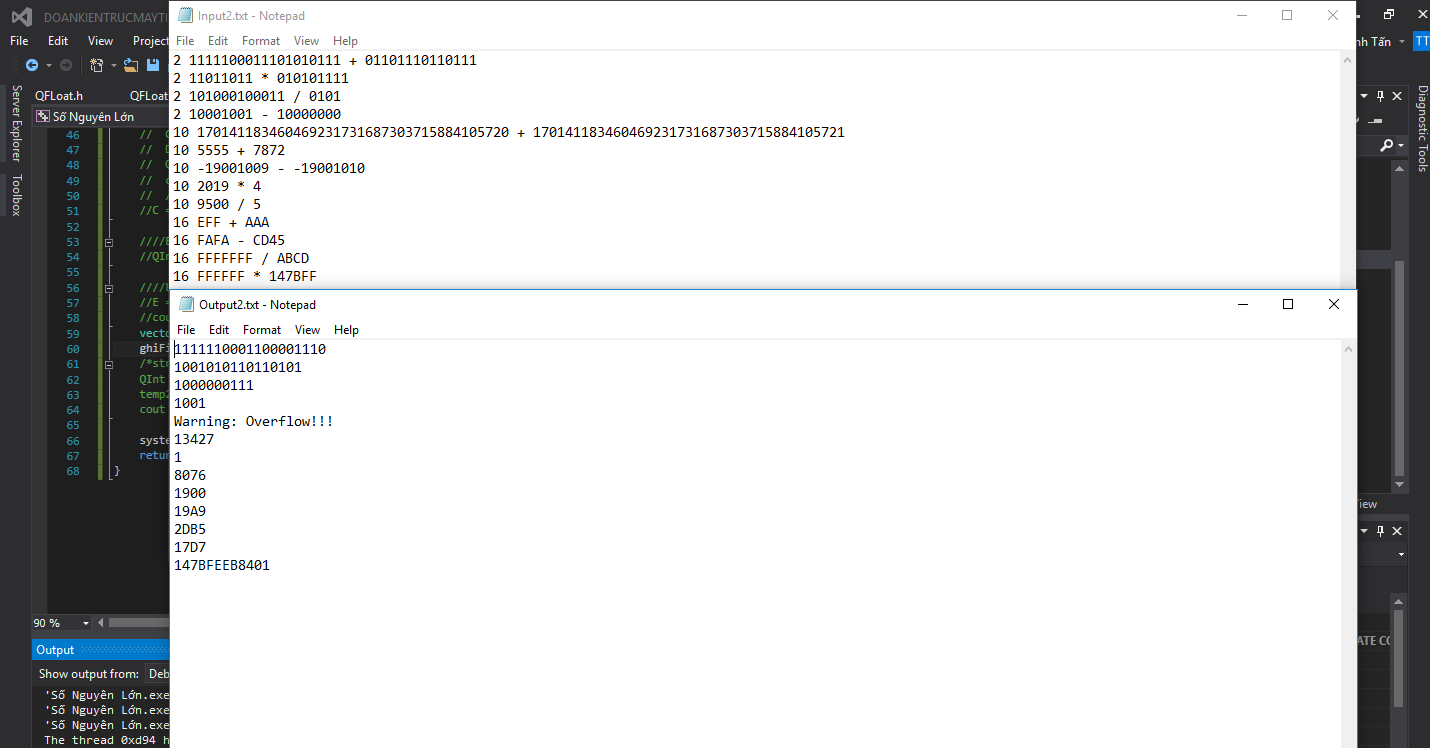
Chạy trên Visual Studio:

* SỐ NGUYÊN LỚN (QINT)

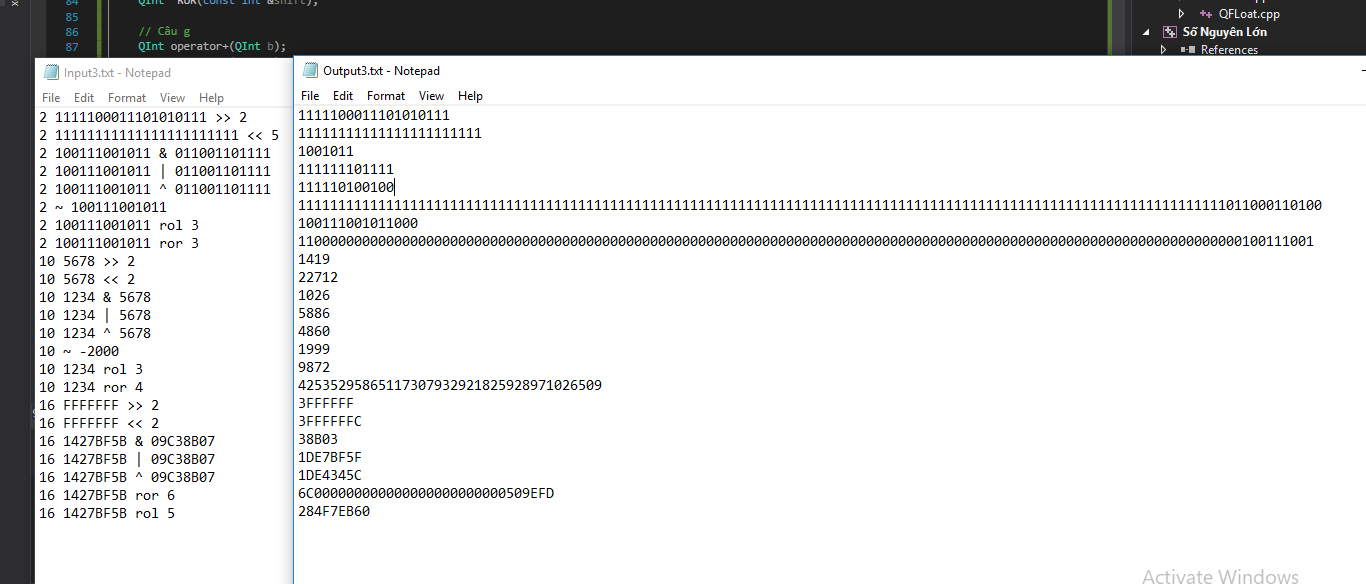
1. *Chuyển đổi cơ số:*



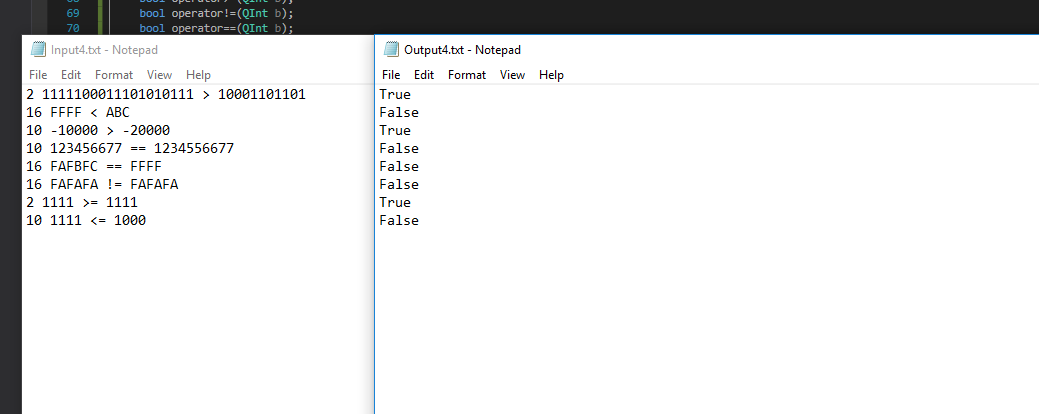
1. *Các Toán Tử: + - \* /*



1. *Các Toán Tử Logic(&, |, ^, ~ , >>, <<) và xoay phải(ROR) , xoay trái (ROL)*

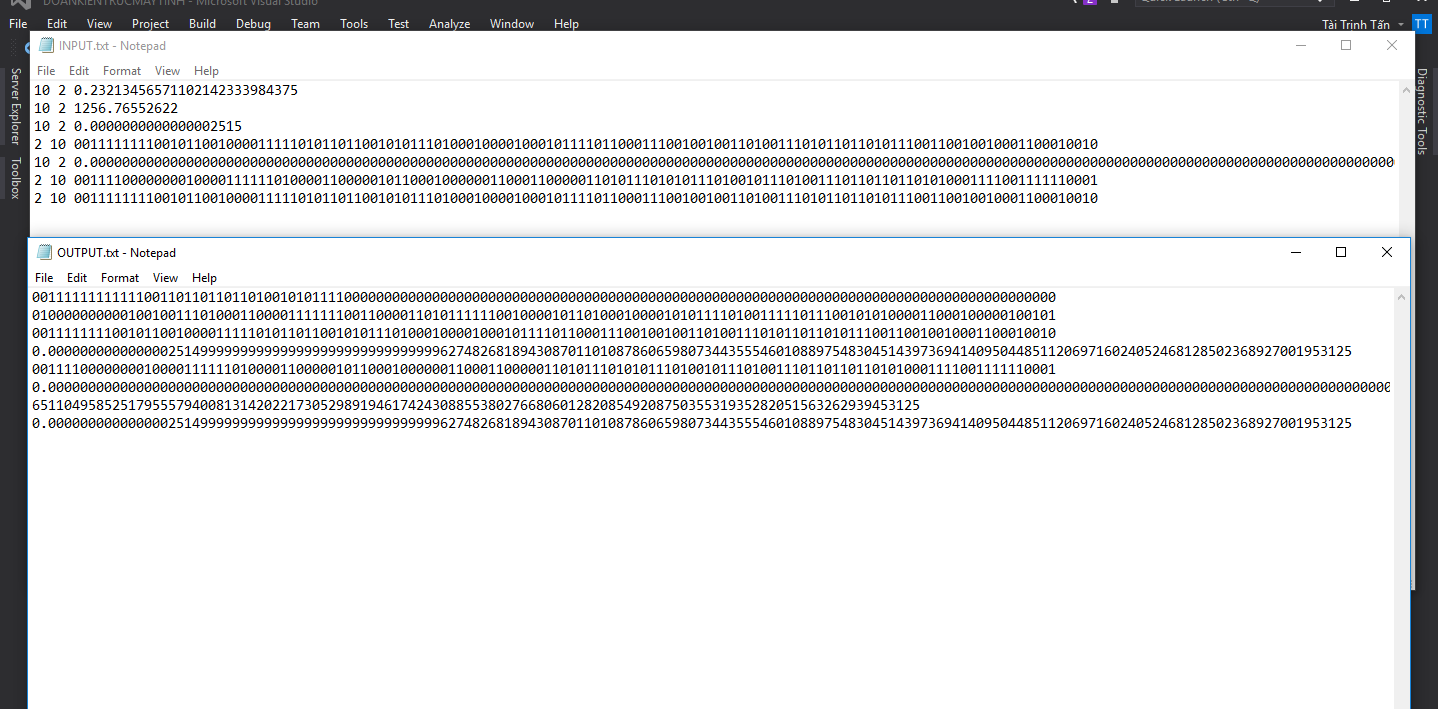


1. *Các toán tử so sánh (>,>=, <, <=,==, !=)*



SỐ CHẤM ĐỘNG CHÍNH XÁC CAO

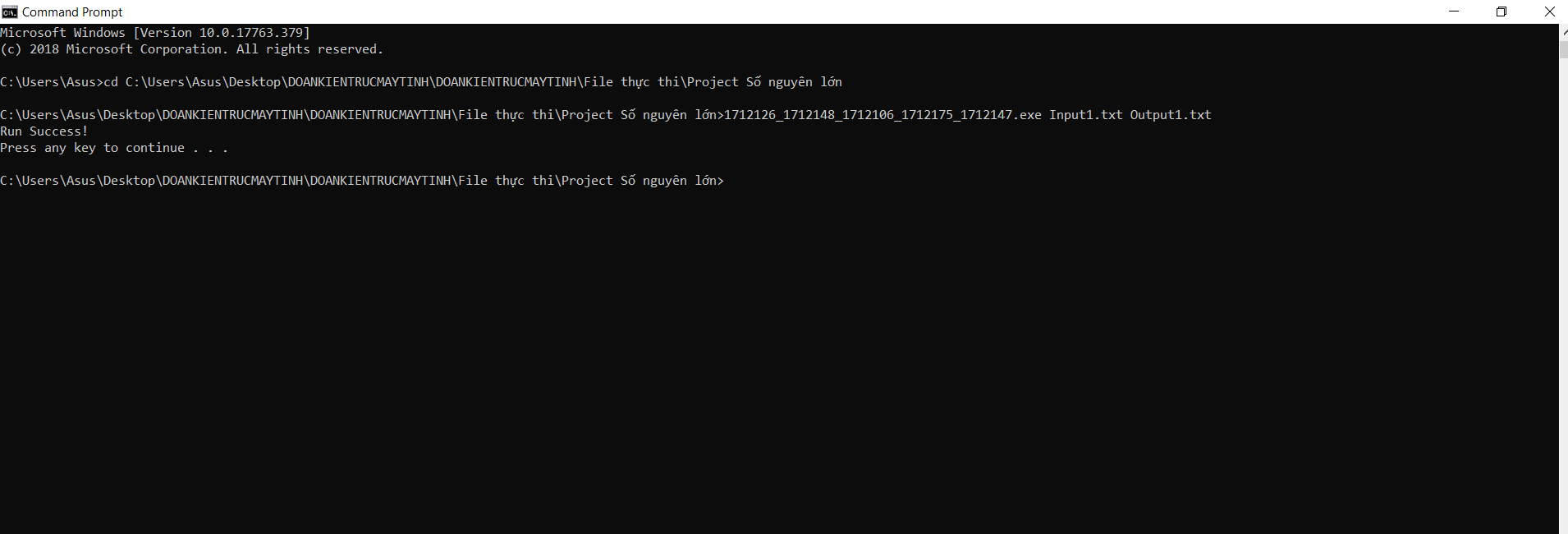
*CHUYỂN ĐỔI CƠ SỐ TỪ FLOAT SANG BINARY VÀ TỪ BINARY SANG FLOAT*



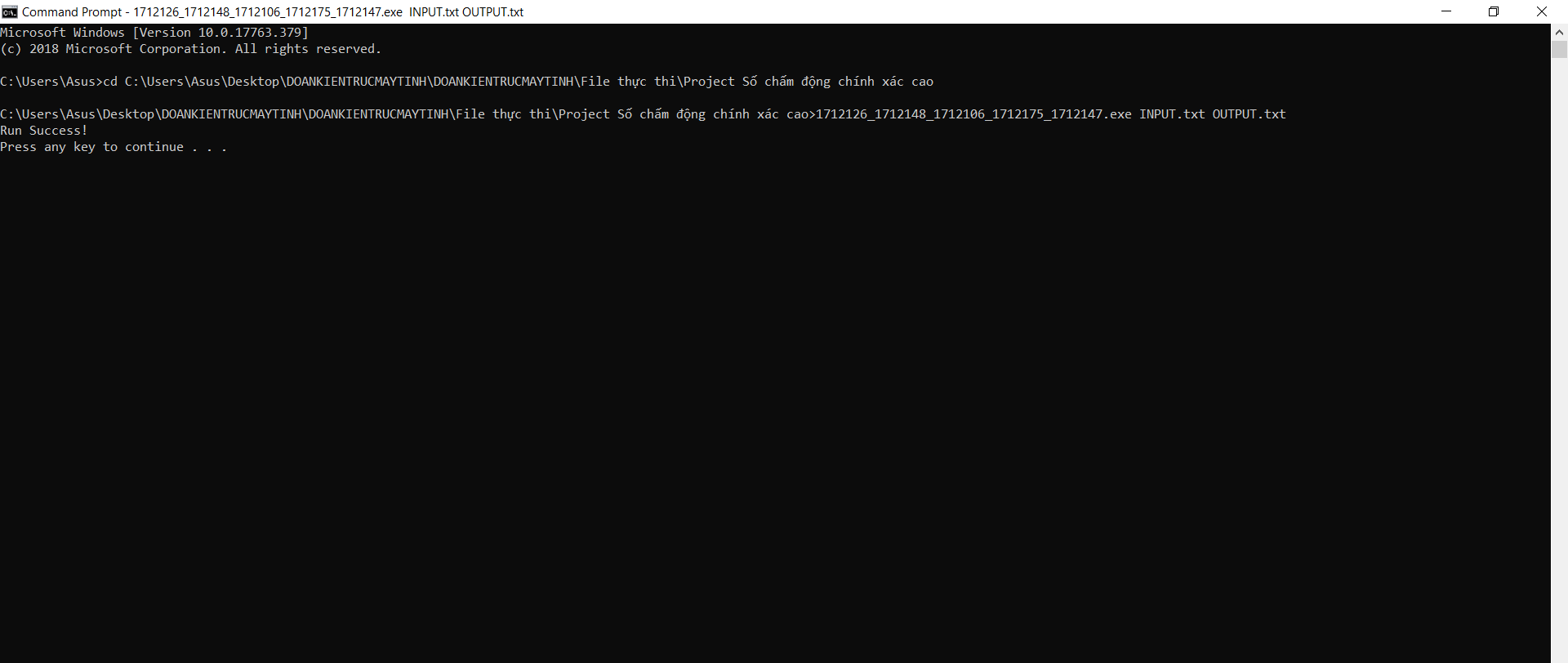
* *NOTE:*
* Input của case 5 xấp xỉ 0.(307 số 0)5
* Output của case 6 xấp xỉ 0.(307 số 0)5

Chạy trên Command Line:

* Số nguyên lớn



* Số chấm động chính xác cao



<https://www.youtube.com/watch?v=VuFZhU612PE&feature=youtu.be&fbclid=IwAR3UrLoh-5Yga_JQMXl9RnN7oyttzcbHvsYFT4axybhWjHJ2e2lhIJcSd8E>

*Link video chạy test chương trình:*

6

NHẬN XÉT CHUNG:

* Với phần số nguyên lớn: (Qint ) chương trình chạy rất tốt hầu hết dưới mọi test được đưa ra.
* Hàm viết đúng yêu cầu, chạy đúng yêu cầu và rõ ràng, dễ hiểu
* Xử lý được hầu hết các trường hợp được đưa ra, bao gồm thông báo, xử lý tràn số, xử lý số âm, ..v…v…
* Với phần số chấm động chính xác cao: (QFLoat) xử lý được yêu cầu của đề bài, hoàn thiện các chức năng và chạy đúng nhiều trường hợp.

*Nhược điểm*: Xử lý khá lâu ở bước chuyển đổi từ Nhị phân sang Float .

7

TÀI LIỆU THAM KHẢO:

* Bài giảng KTMTHN của thầy Lê Viết Long lớp CQ2019/1
* Stackoverflow
* Geeksforgeeks
* Github