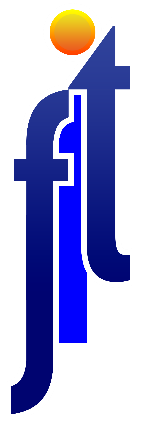
**TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN**

**KHOA CÔNG NGHỆ THÔNG TIN**

**MÔN: KIẾN TRÚC MÁY TÍNH VÀ HỢP NGỮ**



|  |
| --- |
| BÁO CÁO ĐỒ ÁN |
| BIỂU DIỄN VÀ TÍNH TOÁN SỐ NGUYÊN LỚN |

GIẢNG VIÊN HỖ TRỢ:

* Giảng viên lý thuyết: **Phương Phước Hưng**
* Giảng viên thực hành: **Lê Quốc Hòa**

**Lê Viết Long**

Thành phố Hồ Chí Minh, ngày 10 tháng 10 năm 2015

**TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN**

**KHOA CÔNG NGHỆ THÔNG TIN**

**MÔN: KIẾN TRÚC MÁY TÍNH VÀ HỢP NGỮ**

|  |
| --- |
| BÁO CÁO ĐỒ ÁN |
| BIỂU DIỄN VÀ TÍNH TOÁN SỐ NGUYÊN LỚN |

THÔNG TIN THÀNH VIÊN:

1. **Võ Hiếu Trung – 1412592**
2. **Huỳnh Anh Vinh**  **– 1412641**
3. **Lưu Trọng Sang – 1412679**

Thành phố Hồ Chí Minh, ngày 10 tháng 10 năm 2015

Mục lục

[A. MIÊU TẢ ĐỒ ÁN: 1](#_Toc432927564)

[**I.** **Môi trường lập trình:** 1](#_Toc432927565)

[**II.** **Ý tưởng thiết kế:** 1](#_Toc432927566)

[B. MIÊU TẢ ĐỒ ÁN: 1](#_Toc432927567)

[**I.** **Tổng quát:** 1](#_Toc432927568)

[**II.** **Chi tiết:** 2](#_Toc432927569)

[**1.** **Thuật toán và hướng giải quyết các chức năng của chương trình:** 3](#_Toc432927570)

[**2.** **Phạm vi biểu diễn của những kiểu dữ liệu đã thiết kế:** 8](#_Toc432927571)

[**3.** **Hướng dẫn sử dụng chương trình:** 9](#_Toc432927572)

[C. ĐÁNH GIÁ 11](#_Toc432927573)

[**1.** **Tổng quát:** 11](#_Toc432927574)

[**2.** **Chi tiết:** 12](#_Toc432927575)

[D. TÀI LIỆU THAM KHẢO: 12](#_Toc432927576)

# **MIÊU TẢ ĐỒ ÁN:**

## **Môi trường lập trình:**

Visual Studio 2013:

* Sử dụng ngôn ngữ lập trình C++ để xây dựng mã nguồn.
* Sử dụng chế độ Debug để chạy kiểm tra từng dòng lệnh.

## **Ý tưởng thiết kế:**

* Áp dụng phương pháp lập trình hướng đối tượng: xây dựng lớp đối tượng để lưu trữ và quản lý cùng các phương thức, toán tử để thực hiện các phép tính số học và xử lý bit.
* Dữ liệu được lưu trữ, xử lý, tính toán trên bit.
* Tối ưu hóa bộ nhớ lưu trữ và chỉ cấp phát bộ nhớ một vùng nhớ cần thiết.

Dễ dàng nâng thay đổi chuẩn thể hiện của kiểu dữ liệu đã thiết kế, ví dụ số nguyên 8-byte, 10-byte, 16-byte, 32-byte...

* Hỗ trợ đọc dữ liệu và xuất kết quả trên tệp tin.

# **MIÊU TẢ ĐỒ ÁN:**

## **Tổng quát:**

* Lớp số nguyên có tên là **Integer**. Lớp có phạm vi thay đổi và không cố định 16-byte nên nhóm quyết định dùng tên **Integer** thay vì là **QInt**.
* Lớp này có đầy đủ các tính năng như :
  + Đọc các số được lưu trữ ở dạng chuỗi với cơ số 2, 8, 10, 16 và chuyển chúng về đối tượng **Integer**.
  + Xử lý hoàn toàn trên bit các phép toán số học +, -, \*, / cũng như các toán tử xử lý bit *And* (&), *Or* (|), *Xor* (^), *Not* (~), *dịch* và *xoay bit* (<<, >>, ror, rol).
  + Chuyển đổi qua lại giữa các cơ số nhị phân, bát phân, thập phân và thập lục phân.
  + Xuất đối tượng **Integer** ra dạng chuỗi ở cơ số 2, 8, 10, 16.
* Ngoài ra nhóm chúng tôi còn hỗ trợ việc thực hiện trên tệp tin.

## **Chi tiết:**

* Vì giới hạn của đối tượng **Integer** rất lớn và không có kiểu dữ liệu định sẵn nào có thể lưu trữ được (nếu số lượng byte lưu trữ lớn hơn 8) nên nhóm quyết định dùng kiểu chuỗi ký tự để thể hiển và xuất kết quả khi cần thiết.
* Với việc đọc giá trị số nguyên thì dữ liệu đầu vào là kiểu chuỗi (nên sử dụng kiểu **string**). Đầu ra là đối tượng **Integer** với vùng nhớ cấp phát vừa đủ để lưu trữ giá trị này.

*Ví dụ:*

* + Số 127 chỉ cần 1 byte.
  + Số 255 cần 2 byte, vì byte đầu tiên dùng làm byte độn dể chỉ rõ dãy bit biểu diễn giá trị số dương (bởi phạm vi biễu diễn không cố định).
  + Số -1 chỉ cần 1 byte.
* Việc xuất đối tượng luôn đảm bảo đúng chuẩn đề nghị.

*Ví dụ:*

* Khi quy định chuẩn đầu ra là *16-byte*, mặc dù số -1 chỉ lưu trữ trong 1 byte nhưng đầu ra vẫn đầy đủ 128 ký tự ‘1’ cho hệ nhị phân hoặc 32 ký tự ‘F’ cho hệ thập lục phân.
* Nếu chuẩn đầu ra là *32-byte* thì đầu ra ở hệ thập lục phân số biễu diễn số -1 sẽ gồm 64 ký tự ‘F’.
* Có thể xử lý giá trị đầu vào cho hệ nhị phân, bát phân, thập phân và thập lục phân. Tương tự cho việc xuất thể hiện của đối tượng.
* Dãy bit được lưu trữ dưới dạng bit có trọng số nhỏ nhất đầu tiên (Least Signigficant Bit).
* Quản lý tốt bộ nhớ trong việc xử lý và tính toán dữ liệu.

### **Thuật toán và hướng giải quyết các chức năng của chương trình:**

#### **Thuật toán chuyển đổi nhị nhân sang thập phân và ngược lại:**

* Chia nguyên chuỗi chứa giá trị thập phân (gọi tắt là chuỗi thập phân) cho:
  + Biến nhớ **bien\_nho** = 0.
  + Nếu ký tự đầu tiên của chuỗi đầu vào là ‘-’ bắt đầu vòng lặp phía sau với chỉ số **i** bằng 1 và chèn ký tự ‘-’ vào chuỗi kết quả khi kết thúc vòng lặp này.

Ngược lại thì bắt đầu vòng lặp với chỉ số **i** bằng 0.

* + Chừng nào **i** còn nhỏ hơn độ dài chuỗi đầu vào thì
    - Chữ số **a** là chữ số tại chỉ số **i** của chuỗi đầu vào.
    - **gia\_tri** = ((tương ứng của chữ số **a**) + 10 \* **bien\_nho**) chia lấy nguyên cho 2.
    - **a** = **gia\_tri** + ‘0’.
    - **bien\_nho** = (tương ứng của chữ số **a**) chia lấy dư cho 2.
    - Chèn **a** vào cuối chuỗi kết quả.
  + Loại bỏ tất cả các chữ số ‘0’ ở đầu chuỗi kết quả.
* Thập phân sang nhị phân:
  + **vi\_tri\_bit** bằng 0 (đây là vị trí vật lý của bit lưu trữ ở dạng bit có trọng số nhỏ nhất ở đầu (Least Signigficant Bit)).
  + Chừng nào chuỗi đầu vào có độ dài lớn hơn 1 thì
    - Nếu ký tự cuối có giá trị tương ứng là lẻ thì gán bit tại **vi\_tri\_bit** bằng **true**.
    - Gọi thủ tục chia nguyên chuỗi đầu vào cho 2.
    - **vi\_tri\_bit** tăng 1.
  + Nếu ký tự cuối cùng là ‘-’ thì gọi hàm chuyển kết quả thành số đối.

Ngược lại tiếp tục vòng lặp bên trên đến khi nào chuỗi có độ dài bằng không thì dừng.

* Nhân chuỗi thập phân cho 2:
  + Biến nhớ **bien\_nho** = 0, biến đếm **i** = 1.
  + Nếu ký tự đầu tiên của chuỗi đầu vào không phải là ‘-’ thì chừng nào i còn chưa lớn hơn độ dài chuỗi đầu vào
    - Chữ số **a** là chữ số tại chỉ số (**độ-dài-chuỗi** – **i**) của chuỗi đầu vào.
    - **gia\_tri** = (giá trị tương ứng của a) \* 2 + **bien\_nho**.
    - **bien\_nho** = **gia\_tri** chia lấy nguyên cho 10.
    - **gia\_tri** = **gia\_tri** chia dư cho 10.
    - **a** = chữ số tương ứng của **gia\_tri**.
    - Chèn chữ số **a** vào đầu chuỗi kết quả.
    - Nếu **bien\_nho** khác 0 thì chèn ký tự tương ứng với giá trị của **bien\_nho** vào đầu chuỗi kết quả.

Ngược lại thì tương tự như trên nhưng vòng lặp sẽ dừng khi **i** bằng độ dài chuỗi đầu vào và chèn thêm ký tự ‘-’ vào đầu chuỗi kết quả.

* Cộng hai chuỗi thập phân có giá trị dương:
  + Thực hiện phép cộng cho 2 chuỗi **str1** và **str2** có **str1**.**lenght** <= **str2**.**lenght**.
  + **i** = **str1**.**lenght** – 1, **j** = **str2**.**lenght** – 1, **bien\_nho** = 0.
  + Chừng nào **i** chưa âm thì
    - **gia\_tri** = (giá trị tương ứng của **str1**[**i**]) + (giá trị tương ứng của **str2**[**j**]) + **bien\_nho**.
    - **bien\_nho** = **gia\_tri** chia lấy nguyên cho 10.
    - **gia\_tri** = **gia\_tri** chia lấy dư cho 10.
    - Chữ số **a** = chữ số tương ứng của **gia\_tri**.
    - Chèn **a** vào đầu chuỗi kết quả.
    - **i** giảm 1, **j** giảm 1.
  + Chừng nào **j** chưa âm thì
    - **gia\_tri** = (giá trị tương ứng của **str2**[**j**]) + **bien\_nho**.
    - **bien\_nho** = **gia\_tri** chia lấy nguyên cho 10.
    - **gia\_tri** = **gia\_tri** chia lấy dư cho 10.
    - Chữ số **a** = chữ số tương ứng của **gia\_tri**.
    - Chèn **a** vào đầu chuỗi kết quả.
    - **j** giảm 1.
  + Nếu **bien\_nho** khác 0
    - **gia\_tri** = **bien\_nho** chia lấy dư cho 10.
    - Chữ số **a** = chữ số tương ứng của **gia\_tri**.
    - Chèn **a** vào đầu chuỗi kết quả.
* Nhị phân sang thập phân:
  + Biến đếm **i** = 0.

Chuỗi **tam** = ‘‘1’’.

Chuỗi kết quả **ket\_qua** = ‘‘0’’.

* + Nếu most significant bit là **true** thì gọi phương thức chuyển đối tượng **Integer** đầu vào thành số đối của nó, sau khi thực hiện vòng lăp sẽ chèn ký tự ‘-’ vào chuỗi kết quả **ket\_qua**.

Ngược lại thì chỉ thực hiện vòng lặp.

* Chừng nào **i** còn bé hơn số bit của đối tượng **Integer**
  + Nếu bit có chỉ số vật lý **i** là true thì **ket\_qua** = **ket\_qua** + **tam**.
  + Gọi hàm nhân giá trị **tam** cho 2.

#### **Thuật toán chuyển đổi số nhị phân sang thập lục phân và ngược lại:**

* Nhị phân sang thập lục phân:
  + Tách dãy nhị phân ra thành từng nhóm 4 bit từ phải qua trái (tức là bắt đầu từ bit có trọng số nhỏ nhất).
  + Sau đó tính giá trị thập phân của 4 bit đó, giá trị sẽ là số nguyên từ 0 đến 15.
    - Với giá trị từ 0 đến 9 thì chuyển nó thành chữ số có giá trị tương ứng.
    - Còn giá trị từ 10 đến 15 thì quy đổi qua các chữ cái tương đương. Cụ thể như sau: **10 –> A; 11 –> B; 12 –> C; 13 –> D; 14 –> E, 15 –> F**;
  + Sau đó chèn ký tự ở đầu chuỗi kết quả.
* Thập lục phân sang nhị phân:
  + Lấy từng ký tự của dãy thập lục phân rồi thực hiện chuyển đổi số sang dãy nhị phân 4 bit với giá trị tương ứng.
    - Nếu ký tự từ ‘0’ đến ‘9’: thì lấy giá trị tương ứng của chữ số chuyển đổi qua số nhị phân 4 bit.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Chữ số** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** |
| **Dãy bit** | 0000 | 0001 | 0010 | 0011 | 0100 | 0101 | 0110 | 0111 | 1000 | 1001 |

* + - Nếu ký tự là chữ cái ‘A’, ‘B’, ‘C’, ‘D’, ‘E’, ‘F’: thì chuyển về giá trị thập phân tương ứng, rồi từ đó đổi sang số nhị phân 4 bit. Cụ thể:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Ký tự** | **A** | **B** | **C** | **D** | **E** | **F** |
| **Giá trị thập phân** | 10 | 11 | 12 | 13 | 14 | 15 |
| **Dãy bit** | 1010 | 1011 | 1100 | 1101 | 1110 | 1111 |

* + Sau đó chèn vào cuối chuỗi kết quả.

#### **Các toán tử số học:**

* Toán tử *cộng/trừ*: thực hiện cộng/trừ trên bit là thuật toán đơn giản nên không trình bày trong báo cáo. Nhóm đã rút gọn việc xét trường hợp để mã nguồn được ngắn nhất chỉ với một điều kiện cho việc cộng/trừ một cặp bit.
* Toán tử nhân: sử dụng thuật toán nhân **Booth**.

(*Xem chi tiết tại:* [*http://tinyurl.com/thuattoanBooth*](http://tinyurl.com/thuattoanBooth))

* Toán tử chia: tham khảo trong bài giảng trên lớp lý thuyết do thầy Trương Phước Hưng trình bày.

(*Xem chi tiết tại:* [*http://tinyurl.com/bieudiensonguyen*](http://tinyurl.com/bieudiensonguyen)*)*

Nhóm có chỉnh sửa bổ sung để có thể thực hiện phép chia trên số âm.

#### **Toán tử xử lý bit:**

* Toán tử *And / Or / Xor / Not*:
  + Với toán tử *And / Or / Xor*: thực hiện lần lượt trên từng cặp bit cùng chỉ số của 2 đối tượng **Integer** theo bảng:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **And** | **0** | **1** |  | **Or** | **0** | **1** |  | **Xor** | **0** | **1** |
| **0** | 0 | 0 |  | **0** | 0 | 1 |  | **0** | 0 | 1 |
| **1** | 0 | 1 |  | **1** | 1 | 1 |  | **1** | 1 | 0 |

* Với toán tử *Not*: lấy lần lượt từng bit của đối tượng **Integer** rồi gán lại phủ định của bit đó, như bảng sau:

|  |  |  |
| --- | --- | --- |
| **Not** | **0** | **1** |
|  | 1 | 0 |

* Toán tử *dịch trái / dịch phải*:
  + Với toán tử *dịch trái*: Chuyển tất cả các bit sang vị trí kế tiếp bên trái, bỏ bit cuối bên trái, thêm 0 ở bit bên phải.
  + Với toán *tử dịch phải*: Chuyển tất cả các bit sang vị trí kế tiếp bên phải, bỏ bit cuối bên phải, giữ nguyên bit có trọng số lớn nhất (bit dấu).
* Toán tử *xoay trái / xoay phải* (*ROL / ROR*):
  + *Xoay phải* (*ROR*): Chuyển tất cả các bit sang vị trí kế tiếp bên phải, bit cuối bên phải đổi thành bit cuối bên trái trước khi xoay.
  + *Xoay trái* (*ROL*): Chuyển tất cả các bit sang vị trí kế tiếp bên trái, bit cuối bên trái đổi thành bit cuối bên phải trước khi xoay.

Ngoài những chức năng của đồ án yêu cầu, nhóm chúng tôi đã xây dựng thêm thuật toán chuyển đổi dãy nhị phân sang hệ bát phân và ngược lại.

### **Phạm vi biểu diễn của những kiểu dữ liệu đã thiết kế:**

* Nếu cài đặt lớp **Integer** với độ lớn là *16-byte* thì phạm vị biểu diễn như bảng sau:

|  |  |  |
| --- | --- | --- |
| Hệ cơ số | Giá trị nhỏ nhất có thể biểu diễn | Giá trị lớn nhất có thể biểu diễn |
| Nhị phân | 1000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 | 0111 1111 1111 1111 1111 1111 1111 1111 1111 1111 1111 1111 1111 1111 1111 1111 1111 1111 1111 1111 1111 1111 1111 1111 1111 1111 1111 1111 1111 1111 1111 1111 |
| Thập phân | -170 141 183 460 469 231 731 687 303 715 884 105 728 | 170 141 183 460 469 231 731 687 303 715 884 105 727 |
| Thập lục phân | 80 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | 7F FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF |

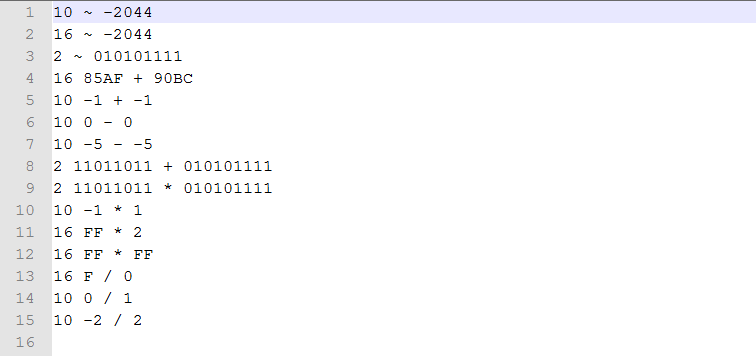
* Tuy nhiên, phạm vi biểu diễn sẽ thay đổi nếu cài đặt lớp **Integer** với độ lớn biểu diễn khác *16-byte.*

### **Hướng dẫn sử dụng chương trình:**

* Quy trình để thực hiện chương trình gồm trải qua các bước:
  + Chuẩn bị file *Input*.
  + Chạy chương trình bằng Command Prompt dưới dạng tham số dòng lệnh.
  + Chương trình sẽ xuất ra tệp tin kết quả.

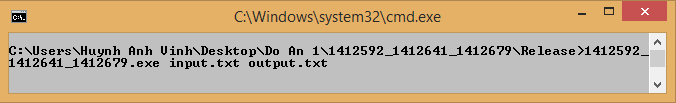
Cụ thể các bước như sau:

* *Chuẩn file input*:
* Vì mã nguồn của chương trình được xây dựng với dữ liệu đầu vào được đọc lên từ tệp tin nên chuẩn tệp tin *input* cần phải quy chuẩn để dữ liệu được đọc và xử lý một cách chính xác.
* Chuẩn *file input* cần đáp ứng đúng như yêu cầu đồ án đã quy định. Tệp tin *input* chắc chắn đảm báo tính đúng đắn thì chương trình mới cho kết quả chính xác.
* Tệp tin *input* mẫu như sau:



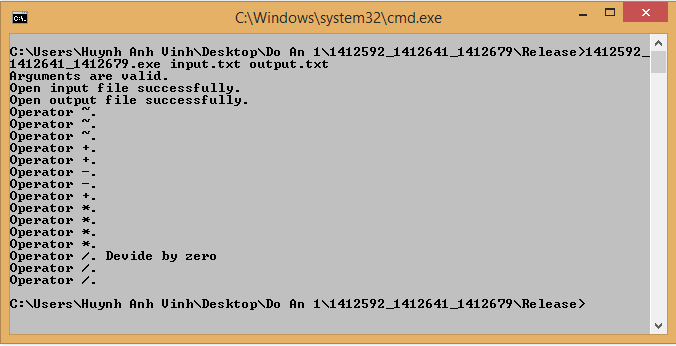
*Hình B2.1. File input mẫu*

* Cách chạy chương trình bằng Command Prompt (Admin):
* Mở trong thư mục *Release* của Folder *1412592\_1412641\_1412679*, chạy Command Prompt tại đây.
* Những tham số đầu vào có dạng: *<tên file .exe> <tên file input> <tên file output>* (như mẫu)



*Hình B2.2: Cách truyền các tham số đầu vào*

* Sau khi chạy xong, chương trình sẽ hiện thông tin như sau:



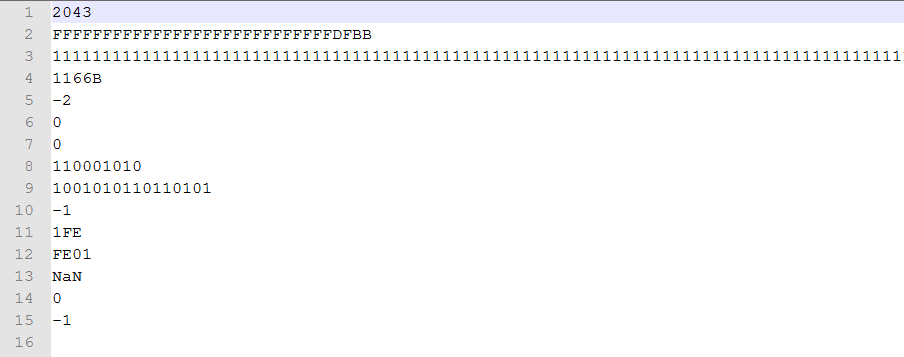
*Hình B2.3: Thông tin xuất ra trên màn hình console*

Trong màn hình console, nếu chương trình chạy thành công sẽ xuất ra n dòng tương ứng với n dòng trong tệp tin *input.* Mỗi dòng sẽ thông báo ra rằng dòng đó sẽ thực hiện việc làm gì (chuyển đổi các hệ số; toán tử hay phép dịch, xoay vòng). Đồng thời, nếu dòng lệnh đó thực thi bị lỗi về mặt logic thì sẽ báo lỗi sau dòng thông báo trên màn hình console.

*Ví dụ*: Phép chia cho 0 sẽ báo lỗi *Divide by zero.*

* Kết quả tệp tin *output*:

Khi chương trình thực hiện xong, thì tệp tin *output* sẽ được xuất ra.



*Hình B2.4: Kết quả file output*

Tệp tin *output* sẽ có n dòng kết quả tương ứng với n dòng yêu cầu của tệp tin *input*. Nếu dòng nào phép tính lỗi hoặc không xử lý được sẽ ghi ra tệp tindòng NaN (Not a Number).

# **ĐÁNH GIÁ**

## **Tổng quát:**

* Sau 2 tuần làm việc, cả nhóm đã hoàn thành tất cả những yêu cầu của đồ án đưa ra về mặt nội dung và hình thức.
* Nhóm tự đánh giá đã hoàn thành 100% đồ án được giao.
* Chương trình chuyển đổi số nguyên lớn đã hoàn thành tốt 100% tất cả những chức năng yêu cầu:
  + Chuyển đổi số nhị phân sang thập phân và ngược lại.
  + Chuyển đổi số nhị phân sang thập lục phân và ngược lại.
  + Toán tử số học: +, - , \*, /
  + Toán tử logic: and &, or |, xor ^, not ~
  + Dịch trái và dịch phải: << ; >>
  + Xoay vòng trái và phải: ror ; rol
* Ngoài những chức năng yêu cầu ở trên, nhóm đã xây dựng thêm chức năng *chuyển đổi số nhị phân sang bát phân và ngược lại.* Chức năng này thực hiện tốt và đảm bảo kết quả 100%.
* Về việc xuất ra kết quả, nhóm đã sử dụng hiệu quả việc quản lý bộ nhớ để kết quả dãy nhị phân hiển thị đúng nhưng vẫn đảm bảo với số byte ít nhất có thể và chỉ hiển thị số chữ số có nghĩa mà không phải xuất đủ 128 chữ số nhị phân (hay 32 chữ số thập lục phân).

## **Chi tiết:**

|  |  |  |
| --- | --- | --- |
| STT | Chức năng | Mức độ tự đánh giá |
| 1 | Chuyển đổi số nhị phân sang thập phân và ngược lại. | 100% |
| 2 | Chuyển đổi số nhị phân sang thập lục phân và ngược lại. | 100% |
| 3 | Toán tử số học: +, - , \*, / | 100% |
| 4 | Toán tử logic: and &, or |, xor ^, not ~ | 100% |
| 5 | Dịch trái và dịch phải: << ; >> | 100% |
| 6 | Xoay vòng: ror ; rol | 100% |
| 7 | Chuyển đổi số nhị phân sang bát phân và ngược lại. (Làm thêm) | 100% |

# **TÀI LIỆU THAM KHẢO:**

* Thuật toán nhân Booth: [*http://tinyurl.com/thuattoanBooth*](http://tinyurl.com/thuattoanBooth)
* Thuật toán nhân và chia từ bài giảng lý thuyết: [*http://tinyurl.com/bieudiensonguyen*](http://tinyurl.com/bieudiensonguyen)