



ĐẠI HỌC KHOA HỌC TỰ NHIÊN- ĐHQG TP.HCM

BÁO CÁO ĐỒ ÁN 1

# BIỂU DIỄN VÀ TÍNH TOÁN TRÊN SỐ NGUYÊN LỚN

**Thành viên nhóm:**

STT	Họ và tên	MSSV	Nhiệm vụ
1	Huỳnh Minh Hiếu	18120371	Lên ý tưởng, chuyển đổi cơ số 2 sang 10, chuyển đổi cơ số 2 sang 16, và ngược lại .
2	Đặng Văn Hiến	18120363	Viết hàm tính toán QInt, toán tử logic, ror, rol. Phụ viết báo cáo.
3	Trần Anh Quân	18120073	Viết báo cáo. Viết hàm tính toán với toán tử + - * /.

- Môi trường lập trình:** Microsoft Visual Studio 2019 bằng ngôn ngữ C++.
- Ý tưởng thiết kế:**
  - Sử dụng 2 phần tử kiểu long long để biểu diễn số QInt sắp xếp kiểu little endian.
  - Xây dựng những hàm tính toán của QInt: các toán tử + - \* / =, các toán tử so sánh , các phép xử lý bit & | ^ ~ << >> ROR ROL
    - Chuyển đổi từ 10 -> 2:** chia dần số thập phân cho 2 cho đến khi kết quả bằng 0, chữ số cuối lẻ là số dư phép chia gán 1/0 cho bit theo thứ tự. Nếu là số âm thì bù 2 kết quả.
    - Chuyển đổi từ 2 -> 10:** chuyển sang hệ 10 theo công thức  $\text{+= } 2^i$ ; thêm dấu - nếu số âm.

- **Chuyển đổi từ 16 -> 2:** 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. Nếu là số âm thì bù 2 kết quả.
- **Chuyển đổi từ 2 ->16:** xét mỗi 4 bit trong số hệ 2 rồi chuyển thành mã hexa tương ứng. Lặp lại cho đến hết; thêm dấu - nếu số âm.
- **Phép gán = :** Tính toán dựa trên các toán tử bitwise.
- **Phép cộng:** Tính toán dựa trên các toán tử bitwise.
- **Phép trừ:** Ta coi  $a - b$  như  $a + (-b)$ . Bù 2 cho  $b$  rồi thực hiện cộng.
- **Phép nhân:** Tính toán dựa trên các toán tử bitwise.
- **Phép chia:** Chuyển sang số dương, rồi tính toán dựa trên các toán tử bitwise.
- **Toán tử AND &:** AND các bit của 2 số với nhau.
- **Toán tử OR |:** OR các bit của 2 số với nhau.
- **Toán tử XOR ^:** XOR các bit của 2 số với nhau.
- **Toán tử NOT ~:** NOT các bit của số đó gán vào kết quả.
- **Phép toán dịch trái <<:** dịch bit sang bên trái  $n$  bit với 2 phần tử long long trên mảng, chú ý k để mất bit giữa 2 phần tử mảng long long.
- **Phép toán dịch phải >>:** tương tự toán tử << .

- **Phép toán ROL:** tương tự toán tử  $\ll$  nhưng chú ý lưu phần bit mất sẽ được gán về sau này.
- **Phép toán ROR:** tương tự toán tử  $\gg$  nhưng chú ý lưu phần bit mất sẽ được gán về sau này.

- Phạm vi biểu diễn:  $-2^{127} \rightarrow 2^{127}-1$

#### 4. Chạy và kiểm tra:

The image shows two Notepad windows side-by-side. The top window, titled 'input.txt - Notepad', contains assembly code for a program that reads a grade from 'input.txt', multiplies it by 10, and adds a bonus of 10. The code uses instructions like 'mov', 'mul', 'add', 'inc', 'ror', and 'rol'. The bottom window, titled '18120371\_18120363\_grade.txt - Notepad', shows the output of the program, which is 'Sum: 100'. The output is circled in red.

- Chuyển đổi số QInt từ hệ thập phân sang hệ nhị phân ( dạng bù 2) và chuyển đổi từ hệ nhị phân sang hệ thập phân:

```

}

deleteFirstZeroChar(bin);

return bin;
}

// Ham chuyen chuoai thap phan sang nhi phan
string DecToBin(string dec)
{
    // chuyen doi theo cach thong thuong
    string bin;
    while (dec != "")
    {
        switch (dec[dec.length() - 1] - 48)
        {
            case 1:
            case 3:
            case 5:
            case 7:
            case 9:
                bin.insert(0, "1");
                break;
            default:
                bin.insert(0, "0");
        }
        DividedBy2(dec);
    }
    return bin;
}

```

```

QInt.cpp  main.cpp
us Files  (Global Scope)

temp.resize(temp.length() + i + j, '0');
result = PlusDecString(result, temp);
}
return result;
}

#pragma endregion

#pragma region Convert_String_By_Base

// Ham chuyen chuoai nhi phan sang thap phan
string BinToDec(string bin)
{
    // chuyen nhi phan sang thap phan theo cach thong
    string dec = "0";
    int len = bin.length();
    for (int i = 0; i < len; i++)
    {
        dec = MultipleDecString(dec, "2");
        dec = PlusDecString(dec, string(1, bin[i]));
    }
    return dec;
}

// Ham chuyen chuoai nhi phan sang thap luc phan
string BinToHex(string bin)
{
    string Bin_Hex[2][2][2][2] = {
        {

```

- Chuyển đổi số QInt từ hệ nhị phân (dạng bù 2) sang hệ thập lục phân và ngược lại

```

Visual Studio
Project Build Debug Team Tools Test Analyze Window H
Debug x86 Local Window

QInt.cpp  main.cpp
us Files  (Global Scope)

// Ham chuyen chuoai nhi phan sang thap luc phan
string BinToHex(string bin)
{
    string Bin_Hex[2][2][2][2] = {
        {
            {"0", "1"},
            {"2", "3"}
        },
        {
            {"4", "5"},
            {"6", "7"}
        },
        {
            {"8", "9"},
            {"A", "B"}
        },
        {
            {"C", "D"},
            {"E", "F"}
        }
    };
    string hex = "";
    // Dam bao chuoai nhi phan co do dai chia het cho 4
    int offset = 4 - bin.length() % 4;
    if (offset) {
        for (int i = 0; i < offset; i++)
        {
            bin.insert(0, "0");
        }
    }
    for (int i = 0; i < bin.length(); i++)
    {
        string temp = bin.substr(i, 4);
        hex += Bin_Hex[temp[0]][temp[1]][temp[2]][temp[3]];
    }
    return hex;
}

```

```

neous Files
string HexToBin(string hex) {
    map<char, string> Hex_Bin;
    Hex_Bin['0'] = "0000";
    Hex_Bin['1'] = "0001";
    Hex_Bin['2'] = "0010";
    Hex_Bin['3'] = "0011";
    Hex_Bin['4'] = "0100";
    Hex_Bin['5'] = "0101";
    Hex_Bin['6'] = "0110";
    Hex_Bin['7'] = "0111";
    Hex_Bin['8'] = "1000";
    Hex_Bin['9'] = "1001";
    Hex_Bin['A'] = "1010";
    Hex_Bin['B'] = "1011";
    Hex_Bin['C'] = "1100";
    Hex_Bin['D'] = "1101";
    Hex_Bin['E'] = "1110";
    Hex_Bin['F'] = "1111";

    map<char, string>::iterator it;
    string bin, temp;

    int len = hex.length();
    for (int i = 0; i < len; i++)
    {
        it = Hex_Bin.find(hex[i]);
        bin += it->second;
    }
    deleteFirstZeroChar(bin);
    return bin;
}

```

- Các operator =, operator +, operator -, operator \*, operator / trên các hệ cơ số:

```

QInt QInt::operator+ (const QInt& x) const {
    QInt a = *this, b = x, temp;
    QInt zero;

    while (1)
    {
        temp = a;
        a = a ^ b;
        b = temp & b;
        if (b == zero) {
            return a;
        }
        b << 1;
    }
}

QInt QInt::operator- (const QInt& x) const {
    QInt b = x;
    b.TwosCompliment();
    return *this + b;
}

QInt QInt::operator* (const QInt& x) const {
    QInt a = *this, b = x, c; // copy QInt -> a

    for (int i = 0; i < 128; i++)
    {
        c << 1;
        if (b.arrayBits[1] < 0) { // <0 thì bit đầu tiên
            c = c + a;
        }
    }
}

```

```

QInt b = x;
b.TwosCompliment();
return *this + b;
}

QInt QInt::operator* (const QInt& x) const {
    QInt a = *this, b = x, c; // copy QInt -> a

    for (int i = 0; i < 128; i++)
    {
        c << 1;
        if (b.arrayBits[1] < 0) { // <0 thì bit đầu tiên
            c = c + a;
        }
        b << 1;
    }
    return c;
}

// Hàm chia lấy nguyên của 2 số QInt dương
QInt dividePositiveQInt(const QInt& x, const QInt& y) {
    QInt quotient(0, 1);
    QInt b = y;

    if (b == x) {
        return QInt(0, 1);
    }
    else if (x < b) {
        return QInt(0, 0);
    }
}

```

```

quotient = quotient + ((x - b) / y);
return quotient;
}

QInt QInt::operator/(const QInt& x) const {
    QInt a = *this;
    QInt b = x;
    QInt c;

    bool neg = false;
    if (a.arrayBits[1] < 0) {
        neg = !neg;
        a.TwosCompliment();
    }
    if (x.arrayBits[1] < 0) {
        neg = !neg;
        b.TwosCompliment();
    }

    // chia c = a / b dương
    c = dividePositiveQInt(a, b);

    if (neg) {
        c.TwosCompliment();
    }
    return c;
}

#pragma endregion

```

```

do {
    b << 1;
    quotient << 1;
} while (b <= x);
b >> 1;
quotient >> 1;

// 1 chut de quy
quotient = quotient + ((x - b) / y);

return quotient;
}

QInt QInt::operator/(const QInt& x) const {
    QInt a = *this;
    QInt b = x;
    QInt c;

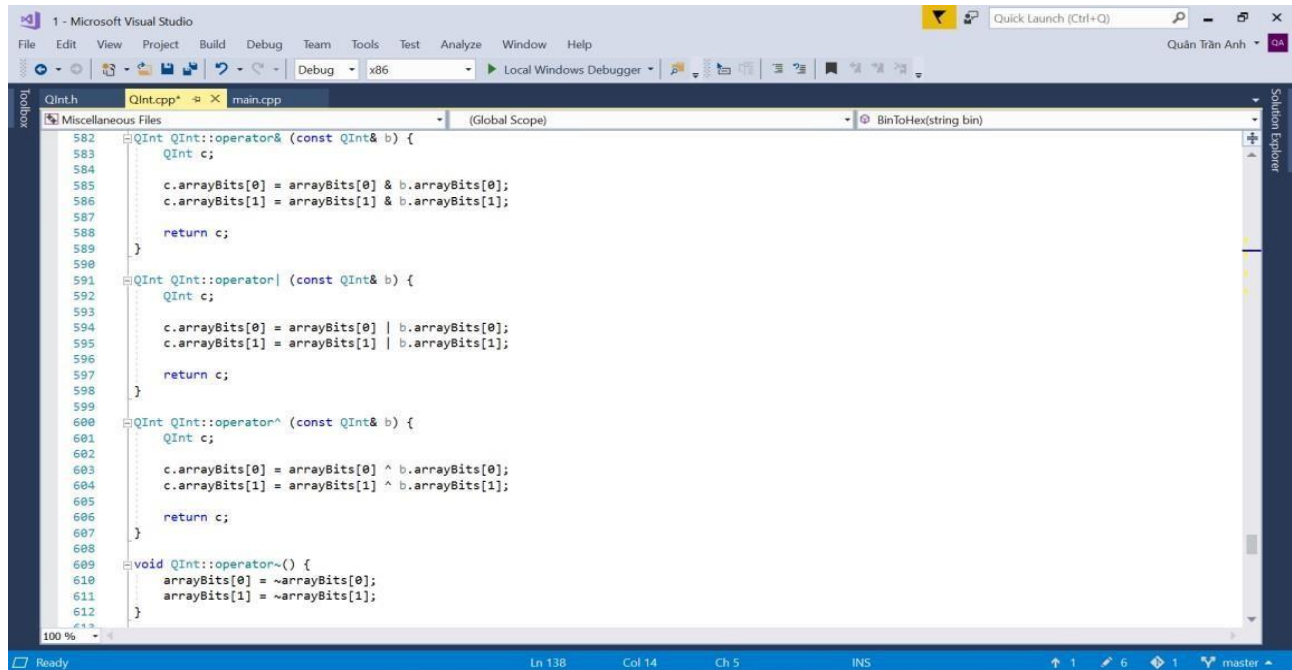
    bool neg = false;
    if (a.arrayBits[1] < 0) {
        neg = !neg;
        a.TwosCompliment();
    }
    if (x.arrayBits[1] < 0) {
        neg = !neg;
        b.TwosCompliment();
    }

    // chia c = a / b dương
    c = dividePositiveQInt(a, b);
}

```



- Các toán tử AND “&”, OR “|”, XOR “^”, NOT “~”:

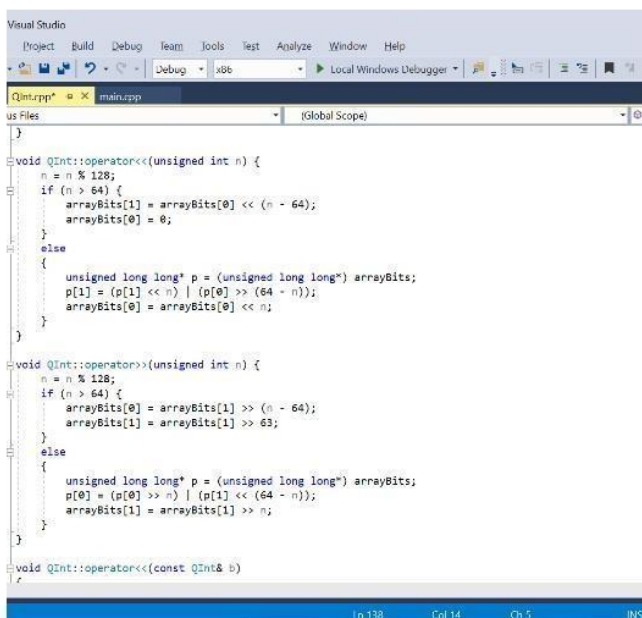


```

1 - Microsoft Visual Studio
File Edit View Project Build Debug Team Tools Test Analyze Window Help
Debug x86 Local Windows Debugger
Solution Explorer
Miscellaneous Files (Global Scope) BinToHex(string bin)
582 QInt QInt::operator& (const QInt& b) {
583     QInt c;
584
585     c.arrayBits[0] = arrayBits[0] & b.arrayBits[0];
586     c.arrayBits[1] = arrayBits[1] & b.arrayBits[1];
587
588     return c;
589 }
590
591 QInt QInt::operator| (const QInt& b) {
592     QInt c;
593
594     c.arrayBits[0] = arrayBits[0] | b.arrayBits[0];
595     c.arrayBits[1] = arrayBits[1] | b.arrayBits[1];
596
597     return c;
598 }
599
600 QInt QInt::operator^ (const QInt& b) {
601     QInt c;
602
603     c.arrayBits[0] = arrayBits[0] ^ b.arrayBits[0];
604     c.arrayBits[1] = arrayBits[1] ^ b.arrayBits[1];
605
606     return c;
607 }
608
609 void QInt::operator~() {
610     arrayBits[0] = ~arrayBits[0];
611     arrayBits[1] = ~arrayBits[1];
612 }
100 %
Ready Ln 138 Col 14 Ch 5 INS

```

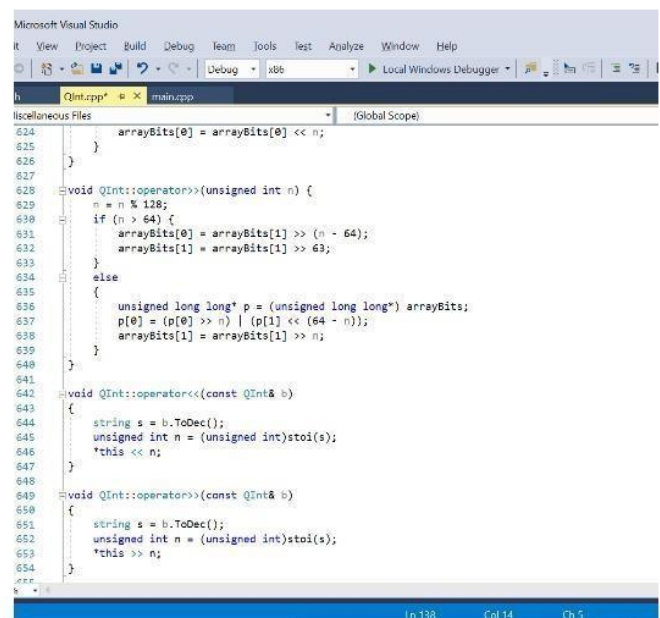
- Các toán tử: dịch trái “<<”, dịch phải “>>” số học



```

Visual Studio
Project Build Debug Team Tools Test Analyze Window Help
Debug x86 Local Windows Debugger
Solution Explorer
Miscellaneous Files (Global Scope)
624 void QInt::operator<< (unsigned int n) {
625     n = n % 128;
626     if (n > 64) {
627         arrayBits[1] = arrayBits[0] << (n - 64);
628         arrayBits[0] = 0;
629     }
630     else {
631         unsigned long long* p = (unsigned long long*) arrayBits;
632         p[1] = (p[1] << n) | (p[0] >> (64 - n));
633         arrayBits[0] = arrayBits[0] << n;
634     }
635 }
636
637 void QInt::operator>> (unsigned int n) {
638     n = n % 128;
639     if (n > 64) {
640         arrayBits[0] = arrayBits[1] >> (n - 64);
641         arrayBits[1] = arrayBits[1] >> 63;
642     }
643     else {
644         unsigned long long* p = (unsigned long long*) arrayBits;
645         p[0] = (p[0] >> n) | (p[1] << (64 - n));
646         arrayBits[1] = arrayBits[1] >> n;
647     }
648 }
649
650 void QInt::operator<< (const QInt& b)
651 {
652 }
653
654
655
Ln 138 Col 14 Ch 5 INS

```

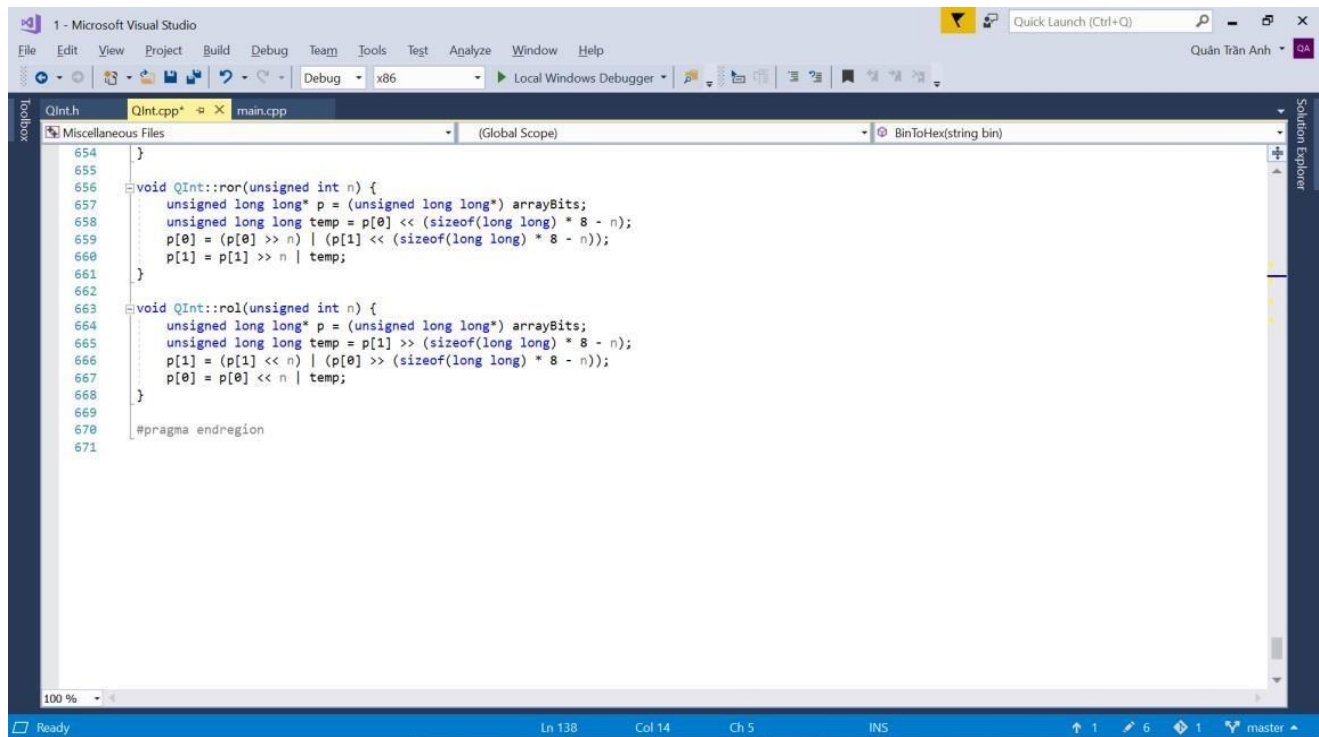


```

Microsoft Visual Studio
File Edit View Project Build Debug Team Tools Test Analyze Window Help
Debug x86 Local Windows Debugger
Solution Explorer
Miscellaneous Files (Global Scope)
624 arrayBits[0] = arrayBits[0] << n;
625 }
626
627 void QInt::operator>> (unsigned int n) {
628     n = n % 128;
629     if (n > 64) {
630         arrayBits[0] = arrayBits[1] >> (n - 64);
631         arrayBits[1] = arrayBits[1] >> 63;
632     }
633     else {
634         unsigned long long* p = (unsigned long long*) arrayBits;
635         p[0] = (p[0] >> n) | (p[1] << (64 - n));
636         arrayBits[1] = arrayBits[1] >> n;
637     }
638 }
639
640 void QInt::operator<< (const QInt& b)
641 {
642     string s = b.ToDec();
643     unsigned int n = (unsigned int)stoi(s);
644     *this << n;
645 }
646
647 void QInt::operator>> (const QInt& b)
648 {
649     string s = b.ToDec();
650     unsigned int n = (unsigned int)stoi(s);
651     *this >> n;
652 }
653
654
655
Ln 138 Col 14 Ch 5

```

- Các phép xoay trái ROL, xoay phải ROR.



```
654 }
655
656 void QInt::ror(unsigned int n) {
657     unsigned long long* p = (unsigned long long*) arrayBits;
658     unsigned long long temp = p[0] << (sizeof(long long) * 8 - n);
659     p[0] = (p[0] >> n) | (p[1] << (sizeof(long long) * 8 - n));
660     p[1] = p[1] >> n | temp;
661 }
662
663 void QInt::rol(unsigned int n) {
664     unsigned long long* p = (unsigned long long*) arrayBits;
665     unsigned long long temp = p[1] >> (sizeof(long long) * 8 - n);
666     p[1] = (p[1] << n) | (p[0] >> (sizeof(long long) * 8 - n));
667     p[0] = p[0] << n | temp;
668 }
669
670 #pragma endregion
671
```

## 5. Đánh giá mức độ hoàn thành:

- Hoàn thành đầy đủ những chức năng được yêu cầu
- Không có chức năng nào không làm được hoặc làm sai
- Hoàn thành 100% đề án