KHOA CÔNG NGHỆ THÔNG TIN
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
227 Nguyễn Văn Cừ, Phường 4, Quận 5, TP.HCM
Điện thoại: (08) 38.354.266 – Fax: (08) 38.350.096

BÁO CÁO BÀI TẬP

HỆ THỐNG MÁY TÍNH

ĐỀ TÀI

TÍNH TOÁN TRÊN SỐ NGUYÊN

Giảng viên hướng dẫn: Thầy Lê Viết Long

STT	Họ và tên	MSSV
1	Trương Tiến Anh	22120017

TP. Hồ Chí Minh, tháng 3 năm 2024

PHẦN ĐÁNH GIÁ

1. Bảng đánh giá

Yêu cầu	Trạng thái	Mức độ hoàn thành (%)
1. Bài tập 1	Hoàn thành	100%
2. Bài tập 2	Hoàn thành	100%

=> Tổng thể mức độ hoàn thành của bài nộp: Hoàn thành 100%

2. Đánh giá tổng kết

Bài nộp đã hoàn thành đầy đủ yêu cầu đề ra, bao gồm việc xử lý bits, áp dụng các phép toán trên bits

KẾT QUẢ LÀM BÀI

1. Chuyển đổi số từ hệ 10 về hệ 2.

Nhập vào số nguyên X (4 byte) có dấu hãy "đọc" dãy bit nhị phân của X và xuất ra màn hình

```
//a)
void printBinary(int x)
{
    cout << "Day bits (4 bytes) cua X: ";
    for (int i = 31; i >= 0; --i)
    {
        cout << ((x >> i) & 1);
    }
}

//b)

void printBinaryArr(int Arr[])
{
    cout << "Day bits tuong ung voi mang Arr[]: ";
    for (int i = 0; i < 32; i++)
    {
        cout << ((Arr[i] >> 0) & 1);
    }
    cout << endl;
}
```

```
int main()
{

    //a)
    cout << "Number: ";
    long long number; cin >> number;
    printBinary(number);
    cout << endl;
    //b)
    srand(time(NULL));
    int Arr[32];
    cout << "Mang Arr[]: ";
    for (int i = 0; i < 32; ++i) {
        Arr[i] = rand() % 2;
        cout << Arr[i];
    }
    cout << endl;
    printBinaryArr(Arr);
    return 0;
}
```

2. Viết chương trình Nhập vào 2 dãy bit 8 bit (ở dạng bù 2). Hãy thực hiện các phép tính cộng, trừ,nhân, chia trên 2 dãy bit đã nhập

Phép cộng:

```
string addBits(string bitA, string bitB)
{
    string result = "";
    int carry = 0;
    int i = bitA.length() - 1, j = bitB.length() - 1;
    while (i >= 0 || j >= 0 || carry) {
        int sum = carry;
        if (i >= 0) sum += bitA[i--] - '0';
        if (j >= 0) sum += bitB[j--] - '0';

        result = to_string(sum % 2) + result;
        carry = sum / 2;
    }
    if (result.length() > bitA.length())
        result = result.substr(1);
    return result;
}
```

Phép trừ:

```
string TwoFromOne(string bit)
{
    string bit1 = "00000001";
    for (int i = 0; i < bit.size(); i++)
    {
        bit[i] = (bit[i] == '0') ? '1' : '0';
    }
    string result = addBits(bit, bit1);
    return result;
}


string subtractBits(string bitA, string bitB)
{
    bitB = TwoFromOne(bitB);//Bu B
    return addBits(bitA, bitB);
}
```

Phép nhân:

```
string mulBits(string bitA, string bitB)
{
    int numA = binaryToDecimal(bitA);
    int numB = binaryToDecimal(bitB);

    if (numA * numB > 0)
    {
        return decimalToBinary(abs(numA) * abs(numB));
    }
    else if (numA * numB < 0)
    {
        string res = decimalToBinary(abs(numA) * abs(numB));
        return TwoFromOne(res);
    }
}
```

Phép chia:

```
pair<string,string> divBits(string bitA, string bitB)
{
    int Q = binaryToDecimal(bitA);
    int M = binaryToDecimal(bitB);

    bool num = Sign(Q);
    bool de = Sign(M);

    Q = abs(Q);
    M = abs(M);
    int A = 0;

    int k = bitA.length();
    while (k > 0)
    {
        A = A << 1;
        Q = Q << 1;
        A = A - M;

        if (A < 0)
        {
            string temp = decimalToBinary(Q);
            temp[temp.size() - 1] = '0';
            Q = binaryToDecimal1(temp);
            A = A + M;
        }
        else
        {
            string temp = decimalToBinary(Q);
            temp[temp.size() - 1] = '1';
            Q = binaryToDecimal1(temp);
        }
        k--;
    }
    string res1, res2;
    if (num && de == false)
    {
        res1 = TwoFromOne(decimalToBinary(Q));
    }
    if (num < 0)
    {
        res2 = TwoFromOne(decimalToBinary(A));
    }
    return { res1,res2 };
}
```