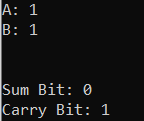
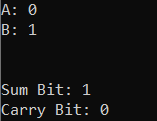
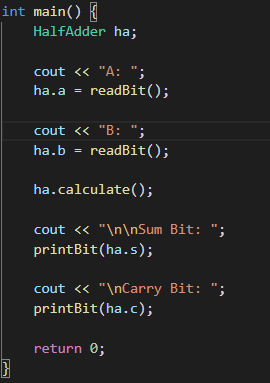
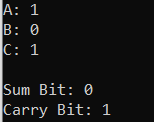
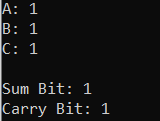
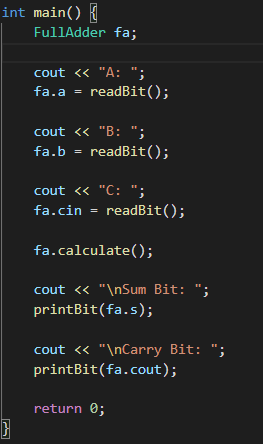
**P5 – Digital Objects**

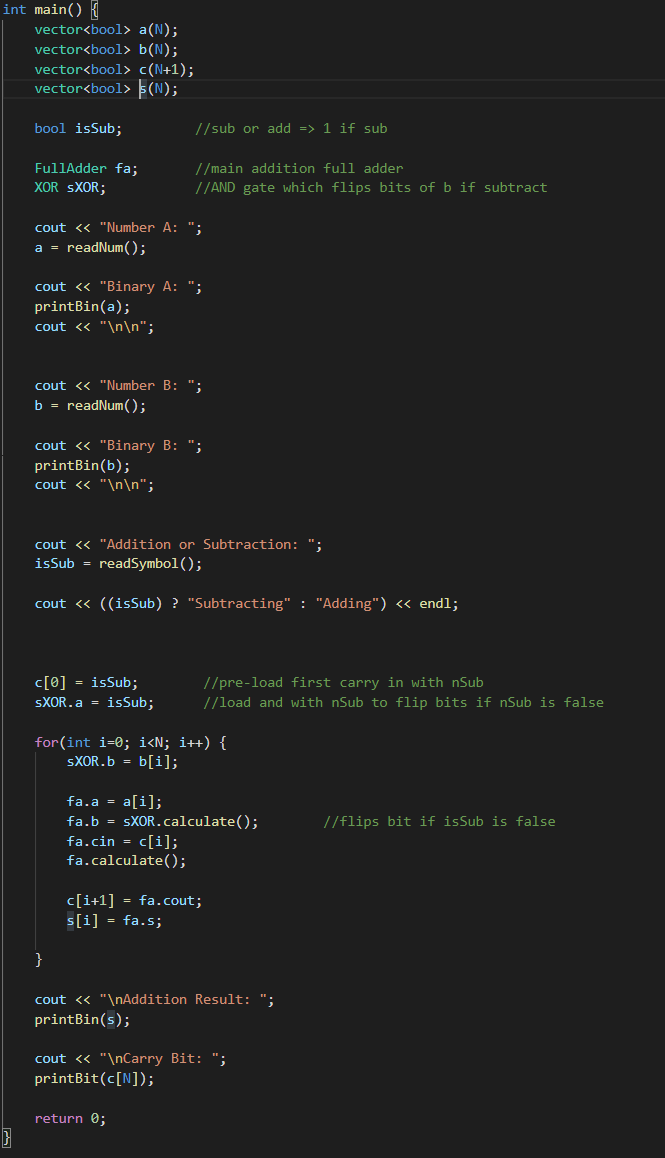
**Final**

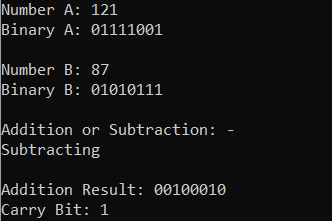
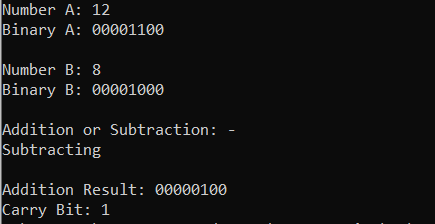
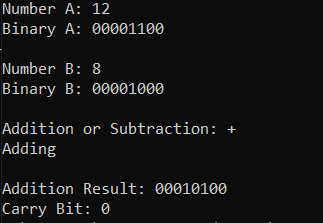
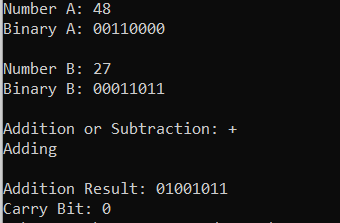
Reusing the “readBit” and “printBit” I had created in the lab P2, I was able to test all of my various gates and adders.



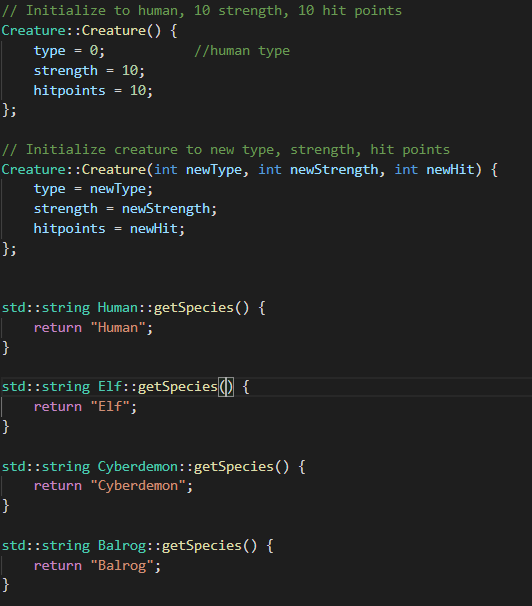


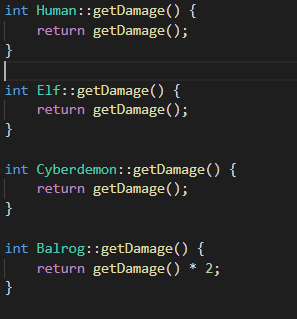
With this completed I then moved on to creating the multi-bit adder and subtracter. For the adder to become a subtractor we must flip all the bits (for b inputs) and set the input carry equal to 1.

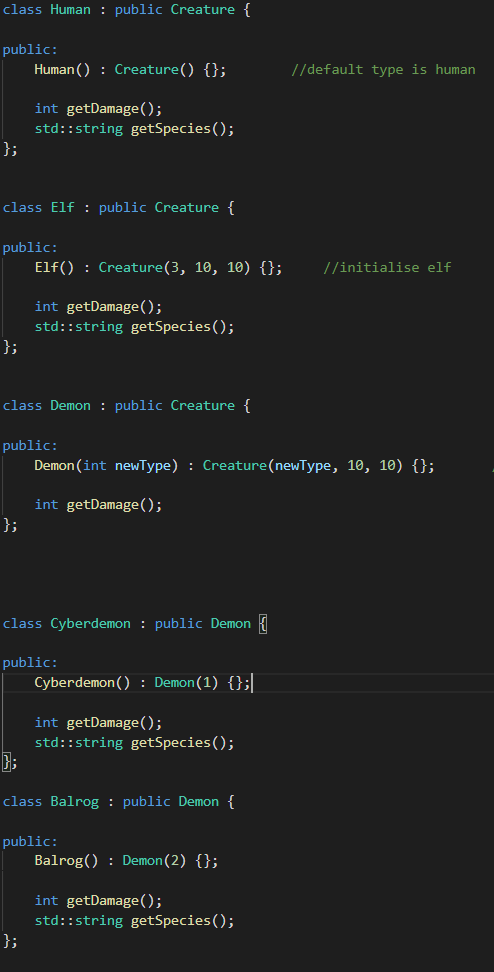


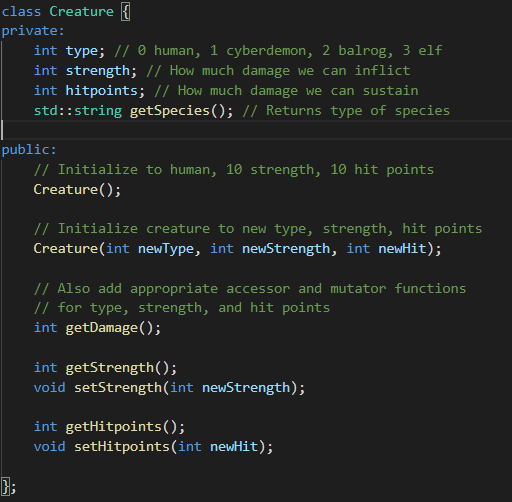


Additional Work









Complete LogicGates

class Gate {

public:

    bool a = false;

    bool b = false;

    Gate() {};

    Gate(bool A, bool B) {

        a = A;

        b = B;

    };

    ~Gate() {delete &a; delete &b;};

};

class AND : public Gate {

public:

    AND() : Gate() {};

    ~AND() {};

    bool calculate() {

        return a && b;

    }

};

class OR : public Gate {

public:

    OR() : Gate() {};

    ~OR() {};

    bool calculate() {

        return a || b;

    }

};

class XOR : public Gate {

public:

    XOR() : Gate() {};

    ~XOR() {};

    bool calculate() {

        return (a != b);

    }

};

class HalfAdder : public XOR, AND  {

public:

    bool a = false;

    bool b = false;

    bool s = false;

    bool c = false;

    HalfAdder() : lXOR(), lAND() {};

    ~HalfAdder() {delete &lXOR; delete &lAND;};

    void calculate() {

        lAND.a = a;

        lAND.b = b;

        lXOR.a = a;

        lXOR.b = b;

        c = lAND.calculate();

        s = lXOR.calculate();

    }

private:

    XOR lXOR;

    AND lAND;

};

class FullAdder : HalfAdder, OR {

public:

    bool cin = false;

    bool a = false;

    bool b = false;

    bool s = false;

    bool cout = false;

    FullAdder() : addr(), lOR() {};

    ~FullAdder() {delete &addr; delete &lOR;};

    void calculate() {

        //using the same HalfAdder twice rather than two seperate

        addr.a = a;

        addr.b = b;

        addr.calculate();

        bool cTemp = addr.c;

        addr.a = cin;

        addr.b = addr.s;

        addr.calculate();

        lOR.a = addr.c;

        lOR.b = cTemp;

        s = addr.s;

        cout = lOR.calculate();

    }

private:

    HalfAdder addr;

    OR lOR;

};