1. Operator overloading (25 points) 
Answer the following about the program operators.cc in the next sheet.

(a) Why does the addition operator for class A at line 11 have two operands but the addition operator for class B at line 26 has only one operand?

(b) Why does the prefix increment operator for class B at line 36 have zero operands?

(c) Notice that in the prefix increment operator at lines 36-40 uses the postfix increment on member variable \( x \). Is this right or wrong? Explain your answer.

(d) Notice that the arguments to the class A addition operator are passed by value but the argument to class B addition operator are passed by reference. Are they both correct? Explain your answer.

(e) What is the value of \( b3.x \) printed at line 55?
// operators program for 2036 exam
#include <iostream>
class A {
public:
    A(); // Default constructor
    A(int); // int Constructor
public:
    int x; // Single data member
};
A operator+(A lhs, A rhs); // Addition operator
class B {
public:
    B(); // Default Constructor
    B(int); // int Constructor
    B operator+(const B& rhs) const; // Addition operator
    B operator=(const B& rhs);
    B& operator++(); // Prefix increment
public:
    int x; // Single data member
};
// Implementation of all constructors and destructor omitted for brevity.
// Addition operator implementations
B B::operator+(const B& rhs) const
{
    return B(x + rhs.x);
}
A operator+(A lhs, A rhs)
{
    return A(lhs.x + rhs.x);
}
// Prefix increment implementation
B& B::operator++()
{
    // This is the implementation of the PREFIX ++ operator for class B
    x++;
    return *this;
}
// Assignment operator implementation
B& B::operator=(const B& rhs)
{
    x = rhs.x;
    return *this;
}
int main()
{
    A a(1);
    B b1(1);
    B b2(2);
    B b3;
    b3 = ++b1 + b2 + b3;
    std::cout << "b3.x is " << b3.x << std::endl;
}
2. What is printed by the program subroutines on the next pages? (25 Points).

Hint: There are eight outputs.
// Subroutine parameter passing example.
// ECE2036, Fall 2012

#include <iostream>
using namespace std;

int Sub2(int arg0, int& arg1)
{
    // Subroutine with two args; does some manipulation of the arguments
    // and then multiplies the results and returns the product.
    return ++arg0 * arg1++;
}

int Sub1(int* pInt0, int* pInt1)
{
    return (*pInt0++) * (*pInt1++);
}

int main()
{
    // Two arrays for manipulating
    int A[10] = { 0, 10, 20, 30, 40, 50, 60, 70, 80, 90 }; 
    int B[10];

    int* pAzero = A;
    int* pAone = A+1;
    int sub1Return = Sub1(pAzero, pAone);
    cout << "Sub1 Return is " << sub1Return << endl;
    cout << "*pAzero is " << pAzero << endl;
    cout << "*pAone is " << pAone << endl;
    int sub2Return = Sub2(A[1], A[2]);
    cout << "Sub2 Return is " << sub2Return << endl;
    int* pA = A;
    int* pB = B;
    *pB++ = *pA++;
    *pB++ = pA[2];
    cout << "B[0] is " << B[0] << endl;
}

Program subroutines.cc
3. Constructors and destructors  (25 Points)

In the program `constructors-destructors.cc` attached, identify where (what line number) each of the default constructors, int constructors, copy constructors, and destructors are called for each class A and B. Specify which line of code causes each of the above and a brief explanation of why the constructor was called. Be sure to note that the addition operator for classes A and B are both defined differently and implemented differently.

As an example of how to fill in the table, one entry for the A int constructor is filled in.

<table>
<thead>
<tr>
<th>A Default Constructor</th>
<th>Line Number</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>A int Constructor</th>
<th>Line Number</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>38</td>
<td>Declaration of local variable “a” with int argument</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>A Copy Constructor</th>
<th>Line Number</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>A Destructor</th>
<th>Line Number</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>B Default Constructor</th>
<th>Line Number</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>B int Constructor</th>
<th>Line Number</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>B Copy Constructor</th>
<th>Line Number</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>B Destructor</th>
<th>Line Number</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
class A {
    public:
        A(); // Default constructor
        A(int); // int Constructor
        A(const A&); // Copy constructor
        ~A(); // Destructor
        A operator+(const A& rhs) const; // Addition operator
    public:
        int x; // Single data member
    }

A A::operator+(const A& rhs) const
{
    A r(x + rhs.x);
    return r;
}

class B {
    public:
        B(); // Default Constructor
        B(int); // int Constructor
        B(const B&); // Copy constructor
        ~B(); // Destructor
        B operator+(B rhs) const; // Addition operator
    public:
        int x; // Single data member
    }

B B::operator+(B rhs) const
{
    return B(x + rhs.x);
}

int main()
{
    A a(1);
    B b(2);
    a = a + a;
    b = b + b;
}

Program constructors-destructors.cc