1. What is the difference between a (binary) operator and a function?

A binary operator like operator+ allows objects to be passed in and added together through the use of overloading. Whereas a function has a name, function arguments and is called usning syntax as opposed to a simple addition of two objects in the operator+ example.

1. Is it possible using operator overloading to change the effect of + on integers? Why or why not?

If the operator is a non-member function it can alter the value of integers. However in most cases it is not necessary to convert to int.

1. Why can’t we overload << or >> as member operators?

We can’t overload << or >> as member operators as they are used as the left hand side of non-member operators. We can’t overload the member operators because they are not symmetric and the compiler cannot perform the same amount of operations on the left and right sides.

4. Below is the definition for a class called Percent. Objects of type Percent represent percentages such as 10% or 99%. Give the definitions of the overloaded operators >> and << so that they can be used for input and output with objects of the class Percent. Assume that input always consists of an integer followed by the character ’%’, such as 25%. All percentages are whole numbers and are stored in an int member variable named value. You do not yet need to define the other overloaded operators and do not yet need to define the constructor. Firstly, you only have to define the overloaded operators >> and <<.

#include <iostream>

using namespace std;

class Percent

{

public:

friend bool operator ==(const Percent& first,

const Percent& second);

friend bool operator <(const Percent& first,

const Percent& second);

Percent( );

friend istream& operator >>(istream& inputStream,

Percent& aPercent);

friend ostream& operator <<(ostream& outputStream,

const Percent& aPercent);

//There will be other members and friends.

private:

int value;

};

1. Add to the class overloaded operators to add, subtract and multiply 2 percentages. If you have to re-write the code so that wherever possible the overloaded operators are made member functions of the class.
2. Implement all of the overloaded operators in the class (and any stand-alone ones still declared as friend functions of the class)

*N.B. Think carefully about the multiplication: 50% \* 50% should mean 50% OF 50%, and therefore provide the answer 25% not 2500%*

1. Write a program which fully tests your class.

5. Write a Money class, with data members euro and cent. Add overloaded operators (as member functions where this is possible) to do the following:

* Subtract 2 Money amounts.
* Multiply a Money amount by an integer to facilitate code like this:

myMoney = yourMoney \* 2;

* Divide a Money amount by a decimal
* Output a Money amount using the insertion operator
* Compare 2 Money amounts (with <, > and ==)