

```

//
//  main.cpp
//  AbsoluteCpp_ch8_2
//

#include <iostream>
#include <cstdlib>
#include <cmath>
using namespace std;

//Class for amounts of money in U.S. currency.
class Money
{
public:
    Money( );
    Money(double amount);
    Money(int dollars, int cents);
    Money(int dollars);
    double getAmount( ) const;
    int getDollars( ) const;
    int getCents( ) const;
    void input( ); //Reads the dollar sign as well as the amount number.
    void output( ) const;
    const Money operator +(const Money& amount2) const;
    const Money operator -(const Money& amount2) const;
    bool operator ==(const Money& amount2) const;
    const Money operator -( ) const;
private:
    int dollars; //A negative amount is represented as negative dollars and
    int cents; //negative cents. Negative $4.50 is represented as -4 and -50

    int dollarsPart(double amount) const;
    int centsPart(double amount) const;
    int round(double number) const;
};

int main( )
{
    Money yourAmount, myAmount(10, 9);
    cout << "Enter an amount of money: ";
    yourAmount.input( );

    cout << "Your amount is ";
    yourAmount.output( );
    cout << endl;
    cout << "My amount is ";
    myAmount.output( );
    cout << endl;

    if (yourAmount == myAmount)
        cout << "We have the same amounts.\n";
    else
        cout << "One of us is richer.\n";

    Money ourAmount = yourAmount + myAmount;

```

```

yourAmount.output( ); cout << " + "; myAmount.output( );
cout << " equals "; ourAmount.output( ); cout << endl;

Money diffAmount = yourAmount - myAmount;
yourAmount.output( ); cout << " - "; myAmount.output( );
cout << " equals "; diffAmount.output( ); cout << endl;

return 0;
}

const Money Money::operator +(const Money& secondOperand) const
{
    int allCents1 = cents + dollars*100;
    int allCents2 = secondOperand.cents + secondOperand.dollars*100;
    int sumAllCents = allCents1 + allCents2;
    int absAllCents = abs(sumAllCents); //Money can be negative.
    int finalDollars = absAllCents/100;
    int finalCents = absAllCents%100;

    if (sumAllCents < 0)
    {
        finalDollars = -finalDollars;
        finalCents = -finalCents;
    }

    return Money(finalDollars, finalCents);
}

const Money Money::operator -(const Money& secondOperand) const
{
    int allCents1 = cents + dollars*100;
    int allCents2 = secondOperand.cents
        + secondOperand.dollars*100;
    int diffAllCents = allCents1 - allCents2;
    int absAllCents = abs(diffAllCents);
    int finalDollars = absAllCents/100;
    int finalCents = absAllCents%100;

    if (diffAllCents < 0)
    {
        finalDollars = -finalDollars;
        finalCents = -finalCents;
    }

    return Money(finalDollars, finalCents);
}

bool Money::operator ==(const Money& secondOperand) const
{
    return ((dollars == secondOperand.dollars)
        && (cents == secondOperand.cents));
}

const Money Money::operator -( ) const
{
    return Money(-dollars, -cents);
}

```

```

}

Money::Money( ): dollars(0), cents(0)
{ /*Body intentionally empty*/}

Money::Money(double amount)
    : dollars(dollarsPart(amount)), cents(centsPart(amount))
{ /*Body intentionally empty*/}

Money::Money(int theDollars)
    : dollars(theDollars), cents(0)
{ /*Body intentionally empty*/}

//Uses cstdlib:
Money::Money(int theDollars, int theCents)
{
    if ((theDollars < 0 && theCents > 0) || (theDollars > 0 && theCents <
        0))
    {
        cout << "Inconsistent money data.\n";
        exit(1);
    }
    dollars = theDollars;
    cents = theCents;
}

double Money::getAmount( ) const
{
    return (dollars + cents*0.01);
}

int Money::getDollars( ) const
{
    return dollars;
}

int Money::getCents( ) const
{
    return cents;
}

//Uses iostream and cstdlib:
void Money::output( ) const
{
    int absDollars = abs(dollars);
    int absCents = abs(cents);
    if (dollars < 0 || cents < 0) //accounts for dollars == 0 or cents == 0
        cout << "$-";
    else
        cout << '$';
    cout << absDollars;

    if (absCents >= 10)
        cout << '.' << absCents;
    else

```

```

        cout << '.' << '0' << absCents;
    }

//Uses iostream and cstdlib:
void Money::input( )
{
    char dollarSign;
    cin >> dollarSign; //hopefully
    if (dollarSign != '$')
    {
        cout << "No dollar sign in Money input.\n";
        exit(1);
    }

    double amountAsDouble;
    cin >> amountAsDouble;
    dollars = dollarsPart(amountAsDouble);
    cents = centsPart(amountAsDouble);
}

int Money::dollarsPart(double amount) const
{
    return static_cast<int>(amount);
}

int Money::centsPart(double amount) const
{
    double doubleCents = amount*100;
    int intCents = (round(fabs(doubleCents))%100); //% can misbehave on
    negatives
    if (amount < 0)
        intCents = -intCents;
    return intCents;
}

int Money::round(double number) const
{
    return static_cast<int>(floor(number + 0.5));
}

```