```
//
//  main.cpp
//  AbsoluteCpp_ch7_4
//
//

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

//Data consists of two items: an amount of money for the account balance
//and a percentage for the interest rate.
class BankAccount
{
public:
    BankAccount(double balance, double rate);
    //Initializes balance and rate according to arguments.
    BankAccount(int dollars, int cents, double rate);
    //Initializes the account balance to $dollars.cents. For a negative
    //balance both dollars and cents must be negative. Initializes the
    //interest rate to rate percent.
    BankAccount(int dollars, double rate);
    //Initializes the account balance to $dollars.00 and
    //initializes the interest rate to rate percent.

    BankAccount( );
    //Initializes the account balance to $0.00 and the interest rate
    //to 0.0%.
    void update( );
    //Postcondition: One year of simple interest has been added to the
    //account.
    void input( );
    void output( ) const;
    double getBalance( ) const;
    int getDollars( ) const;
    int getCents( ) const;
    double getRate( ) const;

    void setBalance(double balance);
    void setBalance(int dollars, int cents);
    //Checks that arguments are both nonnegative or both nonpositive.
    void setRate(double newRate);
    //If newRate is nonnegative, it becomes the new rate. Otherwise,
    //abort program.

private:
    //A negative amount is represented as negative dollars and negative
     cents.
    //For example, negative $4.50 sets accountDollars to -4 and accountCents
    //to -50.
    int accountDollars; //of balance
    int accountCents; //of balance
    double rate;//as a percent
    int dollarsPart(double amount) const;
    int centsPart(double amount) const;
    int round(double number) const;

    double fraction(double percent) const;
    //Converts a percentage to a fraction. For example, fraction(50.3)
    //returns 0.503.
};

//Returns true if the balance in account1 is greater than that
//in account2. Otherwise returns false.
bool isLarger(const BankAccount& account1, const BankAccount& account2);

void welcome(const BankAccount& yourAccount);

int main() {
    BankAccount account1(6543.21, 4.5), account2; welcome(account1);
    cout << "Enter data for account 2:\n"; account2.input( );
    if (isLarger(account1, account2))
        cout << "account1 is larger.\n";
    else
        cout << "account2 is at least as large as account1.\n";

    return 0;
}

bool isLarger(const BankAccount& account1, const BankAccount& account2)  {
 return(account1.getBalance( ) > account2.getBalance( ));
 }
 void welcome(const BankAccount& yourAccount)
{
    cout << "Welcome to our bank.\n"
    << "The status of your account is: \n";
    yourAccount.output();
}

BankAccount::BankAccount(double balance, double rate)
: accountDollars(dollarsPart(balance)),
accountCents(centsPart(balance))
{
    setRate(rate);
}

BankAccount::BankAccount(int dollars, int cents, double rate)
{
    setBalance(dollars, cents);
    setRate(rate);
}

BankAccount::BankAccount(int dollars, double rate)
: accountDollars(dollars), accountCents(0)
{
    setRate(rate);
}

BankAccount::BankAccount( ): accountDollars(0), accountCents(0), rate(0.0)
```

```
{/*Body intentionally empty.*/}


//Uses iostream:
void BankAccount::input( )  {
    double balanceAsDouble;
    cout << "Enter account balance $";
    cin >> balanceAsDouble;
    accountDollars = dollarsPart(balanceAsDouble);
    accountCents = centsPart(balanceAsDouble);
    cout << "Enter interest rate (NO percent sign): ";
    cin >> rate;
    setRate(rate);
}

//Uses iostream and cstdlib:
void BankAccount::output( ) const {
    int absDollars = abs(accountDollars);
    int absCents = abs(accountCents);
    cout << "Account balance: $";
    if (accountDollars > 0)
        cout << "-";
    cout << absDollars;
    if (absCents >= 10)
        cout << "." << absCents << endl;
    else
        cout << "." << '0' << absCents << endl;

    cout << "Rate: " << rate << "%\n";
}

double BankAccount::getBalance( ) const
{
    return (accountDollars + accountCents * 0.01);
}

int BankAccount::getDollars() const {
    return accountDollars;
}

int BankAccount::getCents() const {
    return accountCents;
}

double BankAccount::getRate() const {
    return rate;
}

void BankAccount::setBalance(double balance)
{
    accountDollars = dollarsPart(balance);
    accountCents = centsPart(balance);
}

//Uses cstdlib:
```

```
void BankAccount::setBalance(int dollars, int cents)  {

    if((dollars < 0 && cents > 0) || (dollars > 0 && cents < 0))
    {
        cout << "Inconsistent account data.\n";
        exit(1);
    }
    accountDollars = dollars;
    accountCents = cents;
}

//Uses cstdlib:
void BankAccount::setRate(double newRate)  {
    if(newRate >= 0.0)
        rate = newRate;
    else{
        cout<< "Cannot have a negative interest rate.\n";
        exit(1);
    }
}

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

//Uses cmath:
int BankAccount::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;
}

//Uses cmath:
int BankAccount::round(double number) const {
    return static_cast<int>(floor(number + 0.5));
}

double BankAccount::fraction(double percent) const {
    return (percent/100.0);
}
```