Ex11_10

Account.h

```
#ifndef ACCOUNT_H
#define ACCOUNT_H

class Account {
public:
   Account( double = 0.0);
   void credit( double );
   bool debit( double );
   void setBalance( double );
   double getBalance();
private:
   double balance;
};

#endif
```

Account.cpp

```
#include <stdexcept>
#include <iostream>
#include "Account.h"

using namespace std;

Account::Account(double bal) {
    setBalance(bal);
}

void Account::credit(double c) {
    if (c > 0.0) {
        balance += c;
    } else {
        throw invalid_argument("Credit must be > 0.\n");
    }
}
```

```
bool Account::debit(double d) {
    if (d <= getBalance()) {
        balance -= d;
        return true;
    } else {
        cout << "Debit amount exceeded account balance.\n";
        return false;
    }
}

void Account::setBalance(double b) {
    if (b >= 0.0) {
        balance = b;
    } else {
        throw invalid_argument("Balance must be >= 0.0.\n");
    }
}
double Account::getBalance() {
    return balance;
}
```

SavingsAccount.h

```
#ifndef SAVINGS_H
#define SAVINGS_H

#include "Account.h"

class SavingsAccount : public Account {
public:
   SavingsAccount( double = 0.0, double = 0.0);

   double calculateInterest();
private:
   double interestRate;
};

#endif
```

SavingsAccount.cpp

```cpp
#include <stdexcept>
#include "SavingsAccount.h"

using namespace std;

SavingsAccount::SavingsAccount(double bal, double i)
: Account(bal), interestRate{i} {
    credit(calculateInterest());
}

double SavingsAccount::calculateInterest() {
    return getBalance() * interestRate/100.0;
}
```

CheckingAccount.h

```cpp
#ifndef CHECKING_H
#define CHECKING_H

#include "Account.h"

class CheckingAccount : public Account {
public:
   CheckingAccount( double = 0.0, double = 0.0);

   void credit( double );
   bool debit( double );
private:
   double transactionFee;

   void chargeFee();
};

#endif
```

CheckingAccount.cpp

```cpp
#include <iostream>
#include <stdexcept>
#include "CheckingAccount.h"

using namespace std;

CheckingAccount::CheckingAccount(double bal, double tf)
: Account{bal}{
    if (tf >= 0.0) {
        transactionFee = tf;
    } else {
        throw invalid_argument("Transaction fee must be >= 0.\n");
    }
}

void CheckingAccount::credit(double c) {
    Account::credit(c);
    chargeFee();
}

bool CheckingAccount::debit(double d) {
    if (d + transactionFee <= getBalance()) {
        Account::debit(d);
        chargeFee();
        return true;
    } else {
        throw invalid_argument("Debit amount exceeded account
balance.\n");
        return false;
    }
}

void CheckingAccount::chargeFee() {
    setBalance(getBalance() - transactionFee);
}
```

11_10.cpp

```cpp
// Test program for Account hierarchy.
```

```cpp
#include <iostream>
#include <iomanip>
#include "Account.h" // Account class definition
#include "SavingsAccount.h" // SavingsAccount class definition
#include "CheckingAccount.h" // CheckingAccount class definition
using namespace std;

int main()
{
   Account account1( 50.0 ); // create Account object
   SavingsAccount account2( 25.0, .03 ); // create SavingsAccount object
   CheckingAccount account3( 80.0, 1.0 ); // create CheckingAccount
object

   cout << fixed << setprecision( 2 );

   // display initial balance of each object
   cout << "account1 balance: $" << account1.getBalance() << endl;
   cout << "account2 balance: $" << account2.getBalance() << endl;
   cout << "account3 balance: $" << account3.getBalance() << endl;

   cout << "\nAttempting to debit $25.00 from account1." << endl;
   account1.debit( 25.0 ); // try to debit $25.00 from account1
   cout << "\nAttempting to debit $30.00 from account2." << endl;
   account2.debit( 30.0 ); // try to debit $30.00 from account2
   cout << "\nAttempting to debit $40.00 from account3." << endl;
   account3.debit( 40.0 ); // try to debit $40.00 from account3

   // display balances
   cout << "\naccount1 balance: $" << account1.getBalance() << endl;
   cout << "account2 balance: $" << account2.getBalance() << endl;
   cout << "account3 balance: $" << account3.getBalance() << endl;

   cout << "\nCrediting $40.00 to account1." << endl;
   account1.credit( 40.0 ); // credit $40.00 to account1
   cout << "\nCrediting $65.00 to account2." << endl;
   account2.credit( 65.0 ); // credit $65.00 to account2
   cout << "\nCrediting $20.00 to account3." << endl;
```

```
   account3.credit( 20.0 ); // credit $20.00 to account3

   // display balances
   cout << "\naccount1 balance: $" << account1.getBalance() << endl;
   cout << "account2 balance: $" << account2.getBalance() << endl;
   cout << "account3 balance: $" << account3.getBalance() << endl;

   // add interest to SavingsAccount object account2
   double interestEarned = account2.calculateInterest();
   cout << "\nAdding $" << interestEarned << " interest to account2."
      << endl;
   account2.credit( interestEarned );

   cout << "\nNew account2 balance: $" << account2.getBalance() << endl;
} // end main
```

Result

```
PS C:\vscode\main program\進階程設\ex11_10> ./a
account1 balance: $50.00
account2 balance: $25.01
account3 balance: $80.00

Attempting to debit $25.00 from account1.

Attempting to debit $30.00 from account2.
Debit amount exceeded account balance.

Attempting to debit $40.00 from account3.

account1 balance: $25.00
account2 balance: $25.01
account3 balance: $39.00

Crediting $40.00 to account1.

Crediting $65.00 to account2.

Crediting $20.00 to account3.

account1 balance: $65.00
account2 balance: $90.01
account3 balance: $58.00

Adding $0.03 interest to account2.

New account2 balance: $90.03
```