CSCI 1370

Instructor: Emmett Tomai

The point of these exercises is to allow you to evaluate whether you have learned the material from the past week, and to direct you in additional studying outside of class. These are exactly the types of questions that will show up on your final. My advice is to figure it out on paper, using your book or other resources, and then verify your answer by running the code. If you really want to learn the material, try variations and make sure you understand why the behavior changes the way it does.

Review exercises: classes

1. The familiar `string` class has a method `substr` with the following declaration:

   ```
   string string::substr( int pos, int n );
   ```

   This method returns the part of the string starting at pos and n characters long.

   Declare a string object `phrase` and assign it the value "The pen is blue". Using the `substr` method, print out the first 5 characters in `phrase`.

   Answer:

   ```
   string phrase = "The pen is blue.";
   cout << phrase.substr( 0, 5 );
   ```

2. Define a class `Bank` that has one data member, a double called `amount`. It should also have a default constructor (that takes no arguments). It should also have two public methods: `Deposit` which takes a double and returns nothing, and `Withdraw` which takes a double and returns a double. You are not implementing these methods here, only defining the class.

   Answer:

   ```
   class Bank
   {
   public:
      double amount;

      Bank();
      void Deposit( double money );
      double Withdraw( double requested );
   };
   ```

3. Based on the class definition in question 2, define the default constructor for the `Bank` class so that it sets `amount` to 0.

   Answer:

```
Bank::Bank()
{
    amount = 0;
}
```

4. Based on the class definition in question 2, write the method `Bank::Deposit` to increase the amount in the `Bank` object by the specified amount.

   Answer:

```
void
Bank::Deposit( double money )
{
    amount += money;
}
```

5. Based on the class definition in question 2, write the method `Bank::Withdraw`. If the amount in the Bank object is less that the amount specified, it should print "Overdrawn!", return only the amount possible, and reduce the amount in the object to zero. If the amount in the Bank object is equal or greater to the amount specified, it should return the amount requested and reduce the amount in the Bank object accordingly.

   Answer:

```
double
Bank::Withdraw( double requested )
{
    if( requested > amount )
    {
            cout << "Overdrawn!" << endl;
            requested = amount;
            amount = 0;
            return requested;
    }

    amount = amount - requested;
    return requested;
}
```

6. Based on the class definition in question 2, write C++ statements to declare a Bank object, make two deposits of $25 and $32.50, and a withdrawal of $10.

   Answer:

```
Bank bank;
bank.Deposit( 25.0 );
bank.Deposit( 32.50 );
bank.Withdraw( 10 );
```