7.8 Separate files for classes

Two files per class

Programmers typically put all code for a class into two files, separate from other code.

- ***ClassName.h*** contains the class definition, including data members and member function declarations.
- ***ClassName.cpp*** contains member function definitions.

A file that uses the class, such as a main file or ClassName.cpp, must include ClassName.h. The .h file's contents are sufficient to allow compilation, as long as the corresponding .cpp file is eventually compiled into the program too.

The figure below shows how all the .cpp files might be listed when compiled into one program. Note that the .h file is not listed in the compilation command, due to being included by the appropriate .cpp files.

Figure 7.8.1: Using two separate files for a class.

StoreItem.h

```
#ifndef STOREITEM_H
#define STOREITEM_H

class StoreItem {
   public:
      void SetWeightOunces(int
ounces);
      void Print() const;
   private:
      int weightOunces;
};

#endif
```

StoreItem.cpp

```
#include <iostream>
using namespace std;

#include "StoreItem.h"

void StoreItem::SetWeightOunces(int ounces) {
   weightOunces = ounces;
}

void StoreItem::Print() const {
   cout << "Weight (ounces): " << weightOunces <<
endl;
}
```

main.cpp

Compilation example

```
% g++ -Wall StoreItem.cpp main.cpp
% a.out
Weight (ounces): 16
```

```
#include <iostream>
using namespace std;

#include "StoreItem.h"

int main() {
   StoreItem item1;

   item1.SetWeightOunces(16);
   item1.Print();

   return 0;
}
```

Feedback?

Good practice for .cpp and .h files

Sometimes multiple small related classes are grouped into a single file to avoid a proliferation of files. But for typical classes, good practice is to create a unique .cpp and .h file for each class.

PARTICIPATION ACTIVITY | 7.8.1: Separate files.

1) Commonly a class definition and associated function definitions are placed in a .h file.

True [✓ False]

Correct

Just the class definition appears in the .h. All the function definitions appear in a .cpp file.

2) The .cpp file for a class should #include the associated .h file.

[✓ True] False

Correct

Otherwise, the .cpp file cannot be compiled on its own. The class definition is needed by the functions.

3) A drawback of the separate file approach is longer compilation times.

Correct

Actually, faster compilation results. Consider making a change to a single file in a program with 20 files; separate

True [✓ False]

files means only the changed file needs recompiling, not all 20.

Feedback?

Ex: Restaurant review classes

The restaurant review program, introduced in an earlier section, declared the Review, Reviews, and Restaurant classes in main.cpp. Each of the 3 classes should instead be implemented in .h/.cpp files, thus making for cleaner code in main.cpp.

Figure 7.8.2: .h and .cpp files for Review, Reviews, and Restaurant classes.

Review.h

```
#ifndef REVIEW_H
#define REVIEW_H

#include <string>

class Review {
   public:
      void SetRatingAndComment(
         int revRating,
         std::string revComment);
      int GetRating() const;
      std::string GetComment() const;

   private:
      int rating = -1;
      std::string comment =
"NoComment";
};

#endif
```

Review.cpp

```
#include "Review.h"
using namespace std;

void Review::SetRatingAndComment(int
revRating, string revComment) {
   rating = revRating;
   comment = revComment;
}

int Review::GetRating() const {
   return rating;
}

string Review::GetComment() const {
   return comment;
}
```

Reviews.h

Reviews.cpp

```
#ifndef REVIEWS_H
#define REVIEWS_H

#include <vector>
#include "Review.h"

class Reviews {
   public:
      void InputReviews();
      void PrintCommentsForRating(int
currRating) const;
      int GetAverageRating() const;

   private:
      std::vector<Review> reviewList;
};

#endif
```

```
#include <iostream>
#include "Reviews.h"
using namespace std;

// Get rating comment pairs, add each to list.
-1 rating ends.
void Reviews::InputReviews() {
   Review currReview;
   int currRating;
   string currComment;

   cin >> currRating;
   while (currRating >= 0) {
      getline(cin, currComment); // Gets rest
of line

currReview.SetRatingAndComment(currRating,
currComment);
      reviewList.push_back(currReview);
      cin >> currRating;
   }
}

// Print all comments for reviews having the
given rating
void Reviews::PrintCommentsForRating(int
currRating) const {
   Review currReview;
   unsigned int i;

   for (i = 0; i < reviewList.size(); ++i) {
      currReview = reviewList.at(i);
      if (currRating ==
currReview.GetRating()) {
         cout << currReview.GetComment() <<
endl;
      }
   }
}

int Reviews::GetAverageRating() const {
   int ratingsSum;
   unsigned int i;

   ratingsSum = 0;
   for (i = 0; i < reviewList.size(); ++i) {
      ratingsSum +=
reviewList.at(i).GetRating();
   }
   return (ratingsSum / reviewList.size());
}
```

Restaurant.h

Restaurant.cpp

```
#ifndef RESTAURANT_H
#define RESTAURANT_H

#include <string>
#include "Reviews.h"

class Restaurant {
   public:
      void SetName(std::string
restaurantName);
      void ReadAllReviews();
      void PrintCommentsByRating()
const;

   private:
      std::string name;
      Reviews reviews;
};

#endif
```

```
#include <iostream>
#include "Restaurant.h"
using namespace std;

void Restaurant::SetName(string
restaurantName) {
   name = restaurantName;
}

void Restaurant::ReadAllReviews() {
   cout << "Type ratings + comments. To end:
-1" << endl;
   reviews.InputReviews();
}

void Restaurant::PrintCommentsByRating() const
{
   int i;

   cout << "Comments for each rating level: "
<< endl;
   for (i = 1; i <= 5; ++i) {
      cout << i << ":" << endl;
      reviews.PrintCommentsForRating(i);
   }
}
```

Feedback?

PARTICIPATION ACTIVITY

7.8.2: Restaurant reviews program's main.cpp.

Start 2x speed

main.cpp

```
#include <iostream>
#include "Restaurant.h"
using namespace std;

int main() {
   Restaurant ourPlace;
   string currName;

   cout << "Type restaurant name: " << endl;
   getline(cin, currName);
   ourPlace.SetName(currName);
   cout << endl;

   ourPlace.ReadAllReviews();
   cout << endl;

   ourPlace.PrintCommentsByRating();

   return 0;
}
```

Console:

```
Type restaurant name:
Maria's Healthy Food

Type ratings + comments. To end: -1
5 Great place!
5 Loved the food.
2 Pretty bad service.
4 New owners are nice.
2 Yuk!!!
4 What a gem.
-1

Comments for each rating level:
1:
2:
Pretty bad service.
Yuk!!!
3:
4:
New owners are nice.
What a gem.
5:
Great place!
Loved the food.
```

Feedback?

PARTICIPATION ACTIVITY | 7.8.3: Restaurant review program .h and .cpp files.

Review.cpp

Restaurant.cpp	#includes the "Restaurant.h" header file. Restaurant.cpp is the only file that includes Restaurant.h.	**Correct**
Reviews.cpp	Uses cin and getline() statements to get ratings and comments from the user. The implementation of the InputReviews() member function contains cin and getline() statements for getting integer ratings and string comments.	**Correct**
Restaurant.h	Makes the Restaurant, Reviews, and Review classes available when being #included by another code file. Restaurant.h includes Reviews.h, which includes Review.h. So all 3 classes become available when Restaurant.h is included.	**Correct**
Review.h	Does not #include any of the 3 header files. Review.h defines the Review class, which does not depend on the Reviews or Restaurant classes, so none of the 3 header files are included.	**Correct**

Reviews.h

#includes the <vector> header file.

The Reviews class has a private member named reviewList, which is a vector of Review objects.

Correct

Implements class member functions, none of which use cin or cout.

Reset

Feedback?

CHALLENGE ACTIVITY 7.8.1: Enter the output of separate files.

Jump to level 1

Type the program's output.

main.cpp | Product.h | Product.cpp | Products.h | Products.cpp | Store.h | Store.cpp

```
#include <string>
#include <iostream>
#include "Store.h"
using namespace std;

int main() {
   Store ourPlace;
   string currName;

   cin >> currName;
   ourPlace.SetName(currName);

   ourPlace.ReadAllProducts();
   ourPlace.PrintSale(1);

   return 0;
}
```

Inp

To
5
6
-1

Ou

1 2

Check Next **Done**. Click any level to practice more. Completion is preserv

✓ The Store, Products, and Product classes are implemented in .h/.cpp files. The Store class
Store's ReadAllProducts() calls Products' InputProducts(). Store's PrintSale() outputs name, str

Products' PrintAfterDiscount().

Yours

```
Toogle's sale:
Tuna: 4
Flowers: 5
```

Expected

```
Toogle's sale:
Tuna: 4
Flowers: 5
```

Feedback?