7.7 Classes and vectors/classes

Vector of objects: A reviews program

Combining classes and vectors is powerful. The program below creates a Review class (reviews might be for a restaurant, movie, etc.), then manages a vector of Review objects.

Figure 7.7.1: Classes and vectors: A reviews program.

```
Type rating + 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

Type rating. To end: -1
5
 Great place!
 Loved the food.
1
4
 New owners are nice.
 What a gem.
-1
```

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

class Review {
   public:
      void SetRatingAndComment(int revRating, string
revComment) {
         rating = revRating;
         comment = revComment;
      }
      int GetRating() const { return rating; }
      string GetComment() const { return comment; }

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

int main() {
   vector<Review> reviewList;
   Review currReview;
   int currRating;
   string currComment;
   unsigned int i;

   cout << "Type rating + comments. To end: -1" << endl;
   cin >> currRating;
   while (currRating >= 0) {
      getline(cin, currComment); // Gets rest of line
      currReview.SetRatingAndComment(currRating,
currComment);
      reviewList.push_back(currReview);
      cin >> currRating;
   }

   // Output all comments for given rating
   cout << endl << "Type rating. To end: -1" << endl;
   cin >> currRating;
   while (currRating != -1) {
      for (i = 0; i < reviewList.size(); ++i) {
         currReview = reviewList.at(i);
         if (currRating == currReview.GetRating()) {
            cout << currReview.GetComment() << endl;
         }
      }
      cin >> currRating;
   }

   return 0;
}
```

Feedback?

PARTICIPATION ACTIVITY | 7.7.1: Reviews program.

Consider the reviews program above.

1) How many member functions

does the Review class have?

Check Show answer

Answer

3

The class definition has one setter and two getter functions. All three are short and are thus defined as inlined. The one setter sets both the review rating and review comment.

2) When currReview is declared, what is the initial rating?

Check Show answer

Answer

-1

In the Review class, rating is declared as:

```
int rating = -1;
```

Thus, any Review object will initially have a rating of -1, as well as a comment of "NoComment".

3) As rating and comment pairs are read from input, what function adds them to vector reviewList? Type the name only, like: append.

Check Show answer

Answer

push_back

The reviewList vector is declared with no elements. The push_back(currReview) creates a new Review object element at the end of reviewList, and copies currReview's data into that object. A real reviewing app might have thousands of elements in the list.

4) How many comments were output for reviews having a rating of 5?

Check Show answer

Answer

2

The two comments were "Great place!" and "Loved the food." The vector allowed easy retrieval of all reviews having a rating of 5. Many other operations could be applied on that vector, like computing the average rating, or removing reviews with certain curse words.

Feedback?

A class with a vector: The Reviews class

A class' private data often involves vectors. The program below redoes the example above, creating a Reviews class for managing a vector of Review objects.

The Reviews class has functions for reading reviews and printing comments. The resulting main() is clearer than above.

The Reviews class has a "getter" function returning the average rating. The function computes the average rather than reading a private data member. The class user need not know how the function is implemented.

Figure 7.7.2: Improved reviews program with a Reviews class.

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

class Review {
   public:
      void SetRatingAndComment(int revRating, string
revComment) {
         rating = revRating;
         comment = revComment;
      }
      int GetRating() const { return rating; }
      string GetComment() const { return comment; }

   private:
      int rating = -1;
      string comment = "NoComment";
};
// END Review class


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

   private:
      vector<Review> reviewList;
};

// 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;
```

```
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

Average rating: 3

Type rating. To end: -1
5
 Great place!
 Loved the food.
1
4
 New owners are nice.
 What a gem.
-1
```

```
      }
   }
}

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());
}
// END Reviews class

int main() {
   Reviews allReviews;
   string currName;
   int currRating;

   cout << "Type ratings + comments. To end: -1" << endl;
   allReviews.InputReviews();

   cout << endl << "Average rating: ";
   cout << allReviews.GetAverageRating() << endl;

   // Output all comments for given rating
   cout << endl << "Type rating. To end: -1" << endl;
   cin >> currRating;
   while (currRating != -1) {
      allReviews.PrintCommentsForRating(currRating);
      cin >> currRating;
   }

   return 0;
}
```

Feedback?

PARTICIPATION ACTIVITY 7.7.2: Reviews program. ✓

Consider the reviews program above.

1) The first class is named Review. What is the second class named? ✓

[✓ Reviews]
reviewList
allReviews

Correct

The only difference from Review is the "s" at the end, suggesting this class manages a set of reviews.

2) How many private data members does the Reviews class have? ✓

Correct

○ 0
[✓ 1]
○ 2

The private data member is reviewsList, which happens to be a vector.

3) Which function reads all reviews?

○ GetReviews()
[✓ InputReviews()]

Correct

The function outputs a message and then reads rating/comment pairs until -1 is reached. This functionality is no longer needed in main().

4) What does PrintCommentsForRating() do?

○ Prints ratings sorted by rating level.
○ Print all ratings above a rating level.
[✓ Print all ratings having a particular rating level.]

Correct

The function prints just those reviews whose ratings equal a particular level, such as 2.

5) Does main() declare a vector?

○ Yes
[✓ No]

Correct

main() declares a variable of type Reviews. Reviews itself has a vector, but main() doesn't declare a vector. In general, main() is simpler, since much of the functionality is now in the Reviews class.

Feedback?

Using Reviews in the Restaurant class

Programmers commonly use classes within classes. The program below improves the Restaurant class by having a Reviews object rather than a single rating.

Figure 7.7.3: Improved reviews program with a Restaurant class.

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

// Review and Reviews classes omitted from figure
// ...


class Restaurant {
   public:
      void SetName(string restaurantName) {
         name = restaurantName;
      }
      void ReadAllReviews();
      void PrintCommentsByRating() const;

   private:
      string name;
      Reviews reviews;
};

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);
   }
}

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;
}
```

```
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 | 7.7.3: Restaurant program with reviews.

Consider the Restaurant program above.

1) How many private data members does the Restaurant class have?

- ○ 0
- ○ 1
- [✓ 2]

Correct

The class has private data members name and reviews. Reviews is a class containing its own data, but that data is hidden to the Restaurant class.

2) Which Restaurant member function reads all reviews?

- ○ GetReviews()
- ○ InputReviews()
- [✓ ReadAllReviews()]

Correct

The Restaurant class uses this name, and then calls the Reviews class' InputReviews() function. The two functions are similar, and could have the same or different names.

3) What does PrintCommentsByRating() do?

- [✓ Prints ratings sorted by rating level.]
- ○ Print all ratings having a particular rating level.

Correct

The Restaurant class designer decided this functionality would be useful to a class user. The Reviews class lacked this function, but fortunately had a PrintCommentsForRating() function that helped.

4) Does main() declare a Reviews object?

- ○ Yes
- [✓ No]

Correct

main() declares a Restaurant object. Within that object are many other things, like a Reviews object, but those things are not visible in main(). Classes can greatly simplify the main() function.

Feedback?

CHALLENGE ACTIVITY 7.7.1: Enter the output of classes and vectors.

Jump to level 1

Type the program's output.

```
// "New" means new compared to previous level
#include <iostream>
#include <string>
```

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

class Product {
   public:
      void SetPriceAndName(int productPrice, string productName) {
         price = productPrice;
         name = productName;
      };
      int GetPrice() const { return price; };
      string GetName() const { return name; };
   private:
      int price; // in dollars
      string name;
};
// END Product class

class Products {
   public:
      void InputProducts();
      void PrintAfterDiscount(int discountPrice);
   private:
      vector<Product> productList;
};

void Products::InputProducts() {
   Product currProduct;
   int currPrice;
   string currName;

   cin >> currPrice;
   while (currPrice > 0) {
      cin >> currName;
      currProduct.SetPriceAndName(currPrice, currName);
      productList.push_back(currProduct);
      cin >> currPrice;
   }
}

void Products::PrintAfterDiscount(int discountPrice) {
   unsigned int i;
   int currDiscountPrice;

   for (i = 0; i < productList.size(); ++i) {
      currDiscountPrice = productList.at(i).GetPrice() - discountPrice;
      cout << "$" << currDiscountPrice << " " << productList.at(i).GetName() << endl;
   }
}
// END Products class

// New: Store class has been added
class Store {
   public:
      void SetName(string storeName) {
         name = storeName;
      }
      void ReadAllProducts();
      void PrintSale(int saleAmount);
   private:
      string name;
      Products products;
};

void Store::ReadAllProducts() {
   products.InputProducts();
}

void Store::PrintSale(int saleAmount) {
   cout << name << "'s sale:" << endl;
   products.PrintAfterDiscount(saleAmount);
}
// New: END Store class

int main() {

   // New: main() now uses Store class
   Store ourPlace;
   string currName;

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

   ourPlace.ReadAllProducts();
   ourPlace.PrintSale(2);

   return 0;
}
```

1 2

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

✓ The Store class has a Products data member. Store's ReadAllProducts() calls Products' Inp outputs name, string literal, then endl, and then calls Products' PrintAfterDiscount().

Yours

```
QMart's sale:
$3 Berries
$6 Paper
```

Expected

```
QMart's sale:
$3 Berries
$6 Paper
```

Feedback?

CHALLENGE ACTIVITY 7.7.2: Writing vectors with classes.

Jump to level 1

✓ 1
✓ 2

Write Album's PrintSongsLongerThan() to print all the songs from the album longer than the value of the parameter songDuration. Use Song's PrintSong() to print the songs.

```
   while (currName != "quit") {
      cin >> currDuration;
      currSong.SetNameAndDuration(currName, currDuration);
      albumSongs.push_back(currSong);
      cin >> currName;
   }
```

```
}

void Album::PrintSongsLongerThan(int songDuration) const {
   unsigned int i;
   Song currSong;

   cout << "Songs longer than " << songDuration << " seconds:" << endl;

   /* Your code goes here */
   for(i=0; i<albumSongs.size(); i++){

      if (albumSongs.at(i).GetDuration()>songDuration){
         albumSongs.at(i).PrintSong();
      }

   }


}

int main() {
   Album musicAlbum;
   string albumName;

   getline(cin, albumName);
   musicAlbum.SetName(albumName);
   musicAlbum.InputSongs();
   musicAlbum.PrintName();
```

1 **2**

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

✓ A for loop may iterate through all the songs in albumSongs. An if statement may check whether the current song's duration is longer than 150. If so, then PrintSong() is called for that song.

✓ 1: Compare output

Input

```
The Dark Side of the Moon
Breathe 163 Time 413 Money 383 Eclipse 123
```

Your output

```
The Dark Side of the Moon
Songs longer than 150 seconds:
Breathe - 163
Time - 413
Money - 383
```

✓ 2: Compare output

Input

```
The Endless River
Sum 288 Skins 157 Unsung 67 Anisina 196 Ca
```

```
The Endless River
```

Your output

```
Songs longer than 150 seconds:
Sum - 288
Skins - 157
Anisina - 196
Calling - 217
Surfacing - 166
```

✓ 3: Compare output

Input

```
Parachutes
Shiver 304 Spies 318 Sparks 227 Yellow 266
```

Your output

```
Parachutes
Songs longer than 150 seconds:
Shiver - 304
Spies - 318
Sparks - 227
Yellow - 266
Trouble - 273
```

✓ 4: Compare output

Input

```
Archangel
Archangel 154 Everlasting 169 Nero 207 Des
```

Your output

```
Archangel
Songs longer than 150 seconds:
Archangel - 154
Everlasting - 169
Nero - 207
Destructo - 151
Caradhras - 252
Aesir - 290
```

Feedback?