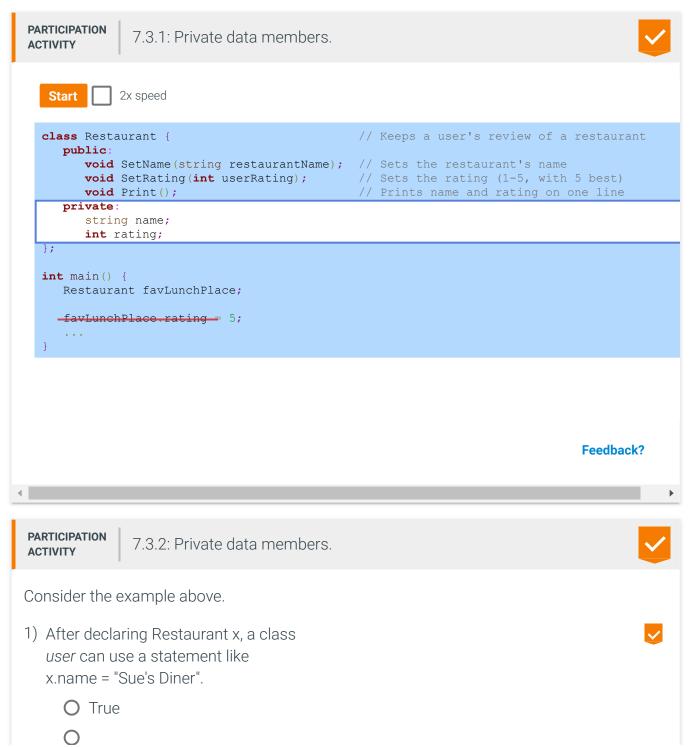
7.3 Defining a class

Private data members

In addition to public member functions, a class definition has **private data members**: Variables that member functions can access but class users cannot. Private data members appear after the word "private:" in a class definition.

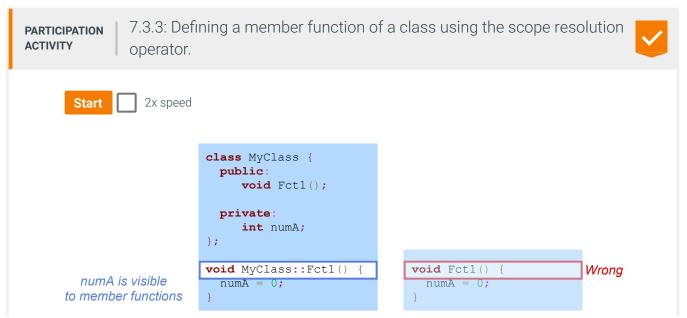


False	
 After declaring a Restaurant object x, a class user can use a statement like myString = x.name. 	
O True	
O False	
3) A class definition should provide comments along with each private data member so that a class user knows how those data members are used.	
O True	
O False	
	Feedback?

Defining a class' public member functions

A programmer defining a class first *declares* member functions after the word "public:" in the class definition. A *function declaration* provides the function's name, return type, and parameter types, but not the function's statements.

The programmer must also *define* each member function. A *function definition* provides a class name, return type, parameter names and types, and the function's statements. A member function definition has the class name and two colons (::), known as the *scope resolution operator*, preceding the function's name. A member function definition can access private data members.



Feedback?

Figure 7.3.1: A complete class definition, and use of that class.

```
#include <iostream>
#include <string>
using namespace std;
class Restaurant {
                                              // Info about a
restaurant
   public:
      void SetName(string restaurantName); // Sets the
restaurant's name
      void SetRating(int userRating);  // Sets the rating (1-
5, with 5 best)
      void Print();
                                             // Prints name and
rating on one line
   private:
      string name;
      int rating;
};
// Sets the restaurant's name
void Restaurant::SetName(string restaurantName) {
   name = restaurantName;
// Sets the rating (1-5, with 5 best)
void Restaurant::SetRating(int userRating) {
   rating = userRating;
}
// Prints name and rating on one line
void Restaurant::Print() {
  cout << name << " -- " << rating << endl;</pre>
int main() {
   Restaurant favLunchPlace;
   Restaurant favDinnerPlace;
   favLunchPlace.SetName("Central Deli");
   favLunchPlace.SetRating(4);
   favDinnerPlace.SetName("Friends Cafe");
   favDinnerPlace.SetRating(5);
   cout << "My favorite restaurants: " << endl;</pre>
   favLunchPlace.Print();
   favDinnerPlace.Print();
   return 0;
```

My favorite restaurants: Central Deli -- 4 Friends Cafe -- 5

Feedback?

PARTICIPATION ACTIVITY	7.3.4: Class definition.	<u> </u>
Consider the	example above.	
1) How is the declared?	Print() member function	<u>~</u>
O Prin	nt();	
O void	l Print();	
O void	<pre>! Restaurant::Print();</pre>	
	the Print() member definition begin?	<u> </u>
O void	Print();	
O void	Restaurant::Print();	
O void	Restaurant::Print()	
begin as fo	Print() function's definition pllows? urant::Print(int x)	✓
O Yes		
O No		
class Resta	rate data members of aurant do the Print() efinition's statements	✓
O SetN	Name	
	unchPlace and DinnerPlace	
O nam	ne and rating	
		Feedback?

Example: RunnerInfo class

The RunnerInfo class below maintains information about a person who runs, allowing a class user to set the time run and the distance run, and to get the runner's speed. The subsequent question set asks for the missing parts to be completed.

Figure 7.3.2: Simple class example: RunnerInfo.

```
#include <iostream>
using namespace std;
class RunnerInfo {
  public:
      void SetTime(int timeRunSecs);
                                          // Time run in seconds
     void SetDist(double distRunMiles);  // Distance run in miles
     double GetSpeedMph();
                                           // Speed in miles/hour
     int timeRun;
      double distRun;
};
void __(B)__::SetTime(int timeRunSecs) {
   timeRun = timeRunSecs; // timeRun refers to data member
void __(C)__SetDist(double distRunMiles) {
  distRun = distRunMiles;
double RunnerInfo::GetSpeedMph() const {
   return distRun / (timeRun / 3600.0); // miles / (secs / (hrs / 3600 secs))
int main() {
   RunnerInfo runner1; // User-created object of class type RunnerInfo
   RunnerInfo runner2; // A second object
   runner1.SetTime(360);
   runner1.SetDist(1.2);
   runner2.SetTime(200);
   runner2.SetDist(0.5);
   cout << "Runner1's speed in MPH: " << runner1. (D) << endl;</pre>
   cout << "Runner2's speed in MPH: " << __(E)__ << endl;</pre>
   return 0;
```

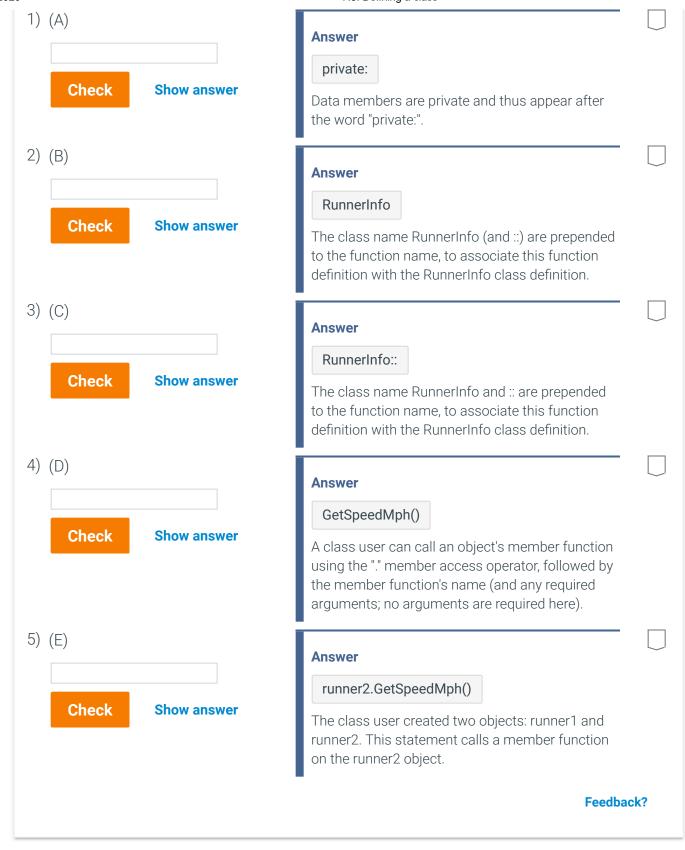
Runner1's speed in MPH: 12 Runner2's speed in MPH: 9

Feedback?

PARTICIPATION ACTIVITY

7.3.5: Class example: RunnerInfo.

Complete the missing parts of the figure above.



Exploring further:

- Classes from cplusplus.com
- Classes from msdn.microsoft.com



```
Kids: 3
New baby, kids now: 4
   19 }
   20
   21 int PersonInfo::GetNumKids() const {
   22
         return numKids;
   23 }
   24
   25 int main() {
   26
         PersonInfo person1;
   27
         int personsKids;
   28
   29
         cin >> personsKids;
   30
         person1.SetNumKids(personsKids);
   31
   32
   33
         /* Your solution goes here */
         cout << "Kids: " << person1.GetNumKids() << endl;</pre>
   34
         person1.IncNumKids();
   35
         cout << "New baby, kids now: " << person1.GetNumKids() << endl;</pre>
   36
   37
   38
   39
         return 0;
   40 }
           ✓ All tests passed
  Run

✓ Testing with input: 3

                            Kids: 3
            Your output
                            New baby, kids now: 4

✓ Testing with input: 6

                            Kids: 6
            Your output
                            New baby, kids now: 7
                                                                                  Feedback?
CHALLENGE
             7.3.3: Basic class definition.
ACTIVITY
Define the missing function. licenseNum is created as: (100000 * customID) +
licenseYear, where customID is a function parameter. Sample output with inputs 2014
777:
```

```
Dog license: 77702014
   22
      licenseNum | (100000 * customID) + licenseYear;
   23 }
   24
   25 int DogLicense::GetLicenseNum() const {
   26
         return licenseNum;
   27 }
   28
   29 int main() {
   30
         DogLicense dog1;
         int userYear;
   31
   32
         int userId;
   33
   34
         cin >> userYear;
   35
         cin >> userId;
   36
   37
         dog1.SetYear(userYear);
         dog1.CreateLicenseNum(userId);
cout << "Dog license: " << dog1.GetLicenseNum() << endl;</pre>
   38
   39
   40
   41
         return 0;
   42 }
            All tests passed
  Run
✓ Testing with inputs: 2014 777
            Your output
                             Dog license: 77702014

✓ Testing with inputs: 2009 321

            Your output
                             Dog license: 32102009
                                                                                     Feedback?
```