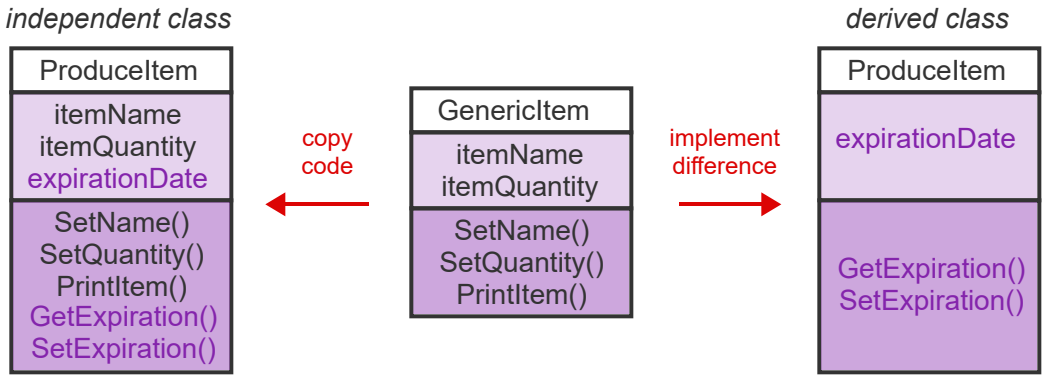
10.1 Derived classes

Derived class concept

Commonly, one class is similar to another class but with some additions or variations. Ex: A store inventory system might use a class called GenericItem that has itemName and itemQuantity data members. But for produce (fruits and vegetables), a ProduceItem class with data members itemName, itemQuantity, and expirationDate may be desired.

PARTICIPATION ACTIVITY 10.1.1: Creating a ProduceItem from GenericItem.

Start 2x speed



Feedback?

PARTICIPATION ACTIVITY 10.1.2: Derived class concept.

1) Creating an independent class that has similar members to an existing class requires copying and pasting code from one class to another.

- (●) True
- ( )

Correct

The new class may reuse the members by duplicating existing code, but creating duplicate code should generally be avoided when possible.

False

2) Creating a derived class is generally less work than creating an independent class.

- (•) True
- ( ) False

Correct

A derived class only needs to implement members that add additional functionality.

Feedback?

Inheritance

A ***derived class*** (or ***subclass***) is a class that is derived from another class, called a ***base class*** (or ***superclass***). Any class may serve as a base class. The derived class is said to inherit the properties of the base class, a concept called ***inheritance***. An object declared of a derived class type has access to all the public members of the derived class as well as the public members of the base class.

A derived class is declared by placing a colon ":" after the derived class name, followed by a member access specifier like public and a base class name. Ex: `class DerivedClass: public BaseClass { ... };`. The figure below defines the base class GenericItem and derived class ProduceItem that inherits from GenericItem.

Figure 10.1.1: Class ProduceItem is derived from class GenericItem.

```
// Base class
class GenericItem {
   public:
      void SetName(string newName) {
          itemName = newName;
      }

      void SetQuantity(int newQty) {
          itemQuantity = newQty;
      }

      void PrintItem() {
          cout << itemName << " " << itemQuantity << endl;
      }

   private:
      string itemName;
      int itemQuantity;
};

// Derived class inherits from GenericItem
class ProduceItem : public GenericItem {
   public:
      void SetExpiration(string newDate) {
          expirationDate = newDate;
      }

      string GetExpiration() {
          return expirationDate;
      }

   private:
      string expirationDate;
};
```

Feedback?

PARTICIPATION ACTIVITY 10.1.3: Using GenericItem and ProduceItem objects.

Start 2x speed

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

// See figure above for class details
class GenericItem { ... };
class ProduceItem : public GenericItem { ... };

int main() {
   GenericItem miscItem;
   ProduceItem perishItem;
```

miscItem

Crunchy Cereal
9

itemName
itemQuan

SetName(
SetQuant
PrintItem(

```
    miscItem.SetName("Crunchy Cereal");
    miscItem.SetQuantity(9);
    miscItem.PrintItem();

    perishItem.SetName("Apples");
    perishItem.SetQuantity(40);
    perishItem.SetExpiration("Dec 5, 2019");
    perishItem.PrintItem();
    cout << "  (Expires: " << perishItem.GetExpiration()
         << ")" << endl;

    return 0;
}
```

perishItem

Apples
40
Dec 5, 2019

itemNam
itemQuar
expiratior

SetName
SetQuan
PrintItem
SetExpira
GetExpira

```
Crunchy Cereal 9
Apples 40
 (Expires: Dec 5, 2019)
```

Feedback?

PARTICIPATION ACTIVITY 10.1.4: Derived classes.

1) A class that can serve as the basis for another class is called a _____ class.

Check Show answer

Answer

base

A derived class refers to a class that is derived from another class that is known as the base class.

2) In the figure above, how many total class members does GenericItem contain?

Check Show answer

Answer

5

GenericItem has 2 data members + 3 member functions = 5 total class members.

3) In the figure above, how many total class members are unique to ProduceItem?

Answer

3

Check Show answer

ProduceItem inherits GenericItem's 5 class members and implements 3 more: expirationDate, SetExpiration(), and GetExpiration().

4) Class Dwelling has data members door1, door2, door3. A class House is derived from Dwelling and has data members wVal, xVal, yVal, zVal. How many data members does `House h;` create?

Check Show answer

Answer

7

4 data members are defined in House, and 3 more data members are inherited from the base class Dwelling.

Feedback?

Inheritance scenarios

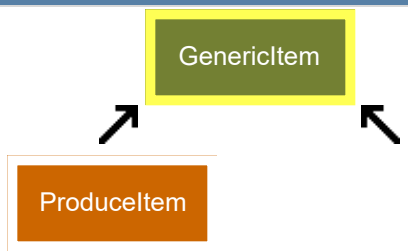
Various inheritance variations are possible:

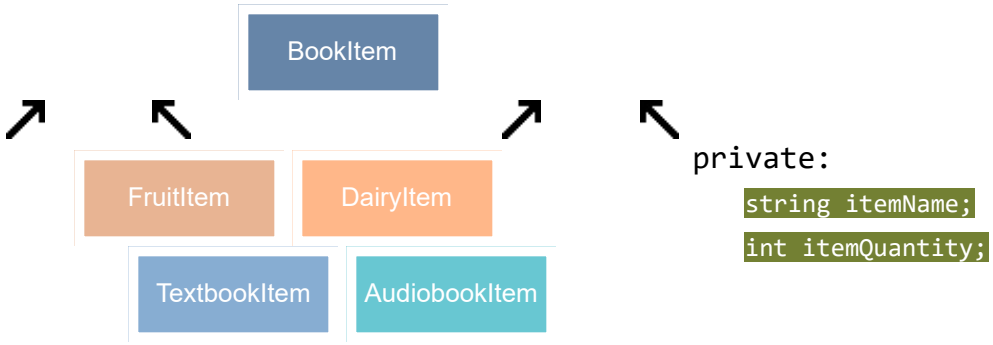
- A derived class can serve as a base class for another class. Ex: `class FruitItem: public ProduceItem {...}` creates a derived class FruitItem from ProduceItem, which was derived from GenericItem.
- A class can serve as a base class for multiple derived classes. Ex: `class FrozenFoodItem: public GenericItem {...}` creates a derived class FrozenFoodItem that inherits from GenericItem, just as ProduceItem inherits from GenericItem.
- A class may be derived from multiple classes. Ex: `class House: public Dwelling, public Property {...}` creates a derived class House that inherits from base classes Dwelling and Property.

PARTICIPATION ACTIVITY 10.1.5: Interactive inheritance tree.

Click a class to see available functions and data for that class.

Inheritance tree	Selected class pseudocode

	`public:`
	`void SetName(string newName)`
	`void SetQuantity(int newQty)`
	`void PrintItem()`



Selected class code

```
class GenericItem {
   public:
      void SetName(string newName)
         { itemName = newName; };
      void SetQuantity(int newQty)
         { itemQuantity = newQty; };
      void PrintItem() {
         cout << itemName << " "
              << itemQuantity
              << endl;
      };

   private:
      string itemName;
      int itemQuantity;
};
```

Feedback?

PARTICIPATION ACTIVITY 10.1.6: Inheritance scenarios.

Refer to the interactive inheritance tree above.

1) The BookItem class acts as a derived class and a base class.

   - (•) True
   - ( ) False

Correct

BookItem is derived from the GenericItem class. BookItem is also a base class when TextbookItem and AudiobookItem are derived from BookItem.

2) ProduceItem and

Correct

BookItem share some of the same class members.

- (•) True
- ( ) False

ProduceItem and BookItem are both derived from GenericItem and share the class members defined in GenericItem.

3) DairyItem and TextbookItem share some of the same class members.

- (•) True
- ( ) False

Correct

DairyItem inherits class members from ProduceItem, which inherits class members from GenericItem. TextbookItem inherits class members from BookItem, which also inherits class members from GenericItem. So class members from GenericItem are shared by DairyItem and TextbookItem.

4) AudiobookItem inherits the data member called reader from BookItem.

- ( ) True
- (•) False

Correct

The reader data member is declared in AudiobookItem, not in BookItem.

5) AudiobookItem inherits the member function GetTitle() from BookItem.

- (•) True
- ( ) False

Correct

The member function GetTitle() is declared in BookItem.

Feedback?

Example: Business and Restaurant

The example below defines a Business class with data members name and address. The Restaurant class is derived from Business and adds a rating data member with a getter and setter.

Figure 10.1.2: Inheritance example: Business and Restaurant classes.

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

class Business {
   public:
      void SetName(string busName) {
         name = busName;
      }

      void SetAddress(string busAddress) {
         address = busAddress;
      }

      string GetDescription() const {
         return name + " -- " + address;
      }

   private:
      string name;
      string address;
};

class Restaurant : public Business {
   public:
      void SetRating(int userRating) {
         rating = userRating;
      }

      int GetRating() const {
         return rating;
      }

   private:
      int rating;
};

int main() {
   Business someBusiness;
   Restaurant favoritePlace;

   someBusiness.SetName("ACME");
   someBusiness.SetAddress("4 Main St");

   favoritePlace.SetName("Friends Cafe");
   favoritePlace.SetAddress("500 W 2nd Ave");
   favoritePlace.SetRating(5);

   cout << someBusiness.GetDescription() << endl;
   cout << favoritePlace.GetDescription() << endl;
   cout << "  Rating: " << favoritePlace.GetRating() << endl;

   return 0;
}
```

```
ACME -- 4 Main St
Friends Cafe -- 500 W 2nd Ave
  Rating: 5
```

Feedback?

PARTICIPATION ACTIVITY 10.1.7: Inheritance example.

Refer to the code above.

1) How many member

functions are defined in Restaurant?

- 2
- 3
- 5

Correct

Restaurant defines member functions SetRating() and GetRating().

2) How many member functions can a Restaurant object call?

- 2
- 3
- 5

Correct

Restaurant inherits 3 member functions from Business and defines 2 more member functions. The favoritePlace object in the example above calls all 5 functions.

3) Which function call produces a syntax error?

- `someBusiness.SetRating(4);`
- `favoritePlace.GetRating();`
- `favoritePlace.SetRating(4);`

Correct

someBusiness is a Business object, and can only call member functions defined in Business: SetName(), SetAddress(), and GetDescription().

4) What is the best way to declare a new DepartmentStore class?

- `class DepartmentStore { ... };`
- `class DepartmentStore : public Restaurant { ... };`
- `class DepartmentStore : public Business { ... };`

Correct

Like Restaurant, DepartmentStore should derive from Business so DepartmentStore can inherit all the Business member. DepartmentStore could also add new members like employeeDiscount and storeHours.

Feedback?

Exploring further:

- Inheritance (C++) from msdn.microsoft.com.

CHALLENGE ACTIVITY 10.1.1: Derived classes.

Jump to level 1

Type the program's output.

```
Camel
3 seat
```

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

class Vehicle {
   public:
      void SetSpeed(int speedToSet) {
         speed = speedToSet;
      }

      void PrintSpeed() {
         cout << speed;
      }

   private:
      int speed;
};

class Carriage {
   public:
      void SetSeats(int seatsToSet) {
         seats = seatsToSet;
      }

      void PrintSeats() {
         cout << seats << " seats in wagon";
      }

   private:
      int seats;
};

class AnimalDrawnCarriage : public Vehicle, public Carriage {
   public:
      void SetAnimal(string animalToSet) {
         animal = animalToSet;
      }

      void PrintAnimalSpeed() {
         cout << animal << " speed: ";
         PrintSpeed();
      }

   private:
      string animal;
};

int main() {
   AnimalDrawnCarriage wagon;

   wagon.SetSpeed(10);
   wagon.SetSeats(3);
   wagon.SetAnimal("Camel");

   wagon.PrintAnimalSpeed();
   cout << endl;
   wagon.PrintSeats();

   return 0;
}
```

1 2 3

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

✔ AnimalDrawnCarriage inherits both Vehicle and Carriage. A class can inherit from multiple l

Yours
```
Camel speed: 10
3 seats in wagon
```

Expected
```
Camel speed: 10
3 seats in wagon
```

Feedback?

CHALLENGE ACTIVITY 10.1.2: Basic inheritance.

Assign courseStudent's name with Smith, age with 20, and ID with 9999. Use the PrintAll() member function and a separate cout statement to output courseStudents's data. End with a newline. Sample output from the given program:

```
Name: Smith, Age: 20, ID: 9999
```

```
33      int GetID() {
34         return idNum;
35      }
36
37   private:
38      int idNum;
39};
40
41int main() {
42   StudentData courseStudent;
43
44   /* Your solution goes here  */
45   courseStudent.SetName("Smith");
46   courseStudent.SetAge(20);
47   courseStudent.SetID(9999);
48
49   courseStudent.PrintAll();
50   cout<< ", ID: " << courseStudent.GetID() <<endl;
51
52   return 0;
53}
```

Run ✓ All tests passed

✓ Testing with Smith, 20, 9999.

Your output
```
Name: Smith, Age: 20, ID: 9999
```

✓ Testing again followed by calling PrintAll().

Your output

```
Name: Smith, Age: 20, ID: 9999
Name: Smith, Age: 20
```

✓ Testing ID assigned 9999

Your value `9999`

Feedback?