CST209

Object-oriented Programming C++

(Week 13)

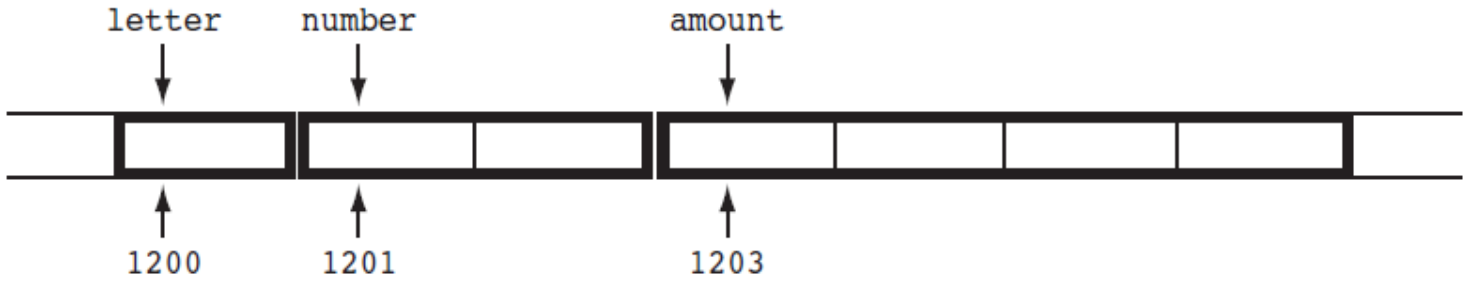
Dr. Teo Bee Guan

Content

- Pointer

Introduction to Pointer

- Every variable is allocated a section of memory large enough to hold a value of the variable's data type.
- Each byte of memory has a unique address .
- Suppose the following variables are defined in a program:
  - char letter;
  - short number;
  - float amount;



Introduction to Pointer

- Getting the address of a variable is accomplished with an operator in C++.

- When the address operator (&) is placed in front of a variable name, it returns the address of that variable.

In-class Practice – Example 1

Pointer Variables

- A pointer variable , which often is just called a pointer , is a special variable that holds a memory address.

- Because a pointer variable holds a memory address, it can be used to hold the location of some other piece of data.

In-class Practice – Example 2, 3

Pointer as Function Parameters

- A pointer can be used as a function parameter.

- It gives the function access to the original argument, much like a reference parameter does.

In-class Practice – Example 4

Pointer to C++ Class

- Just like pointers to normal variables and functions, we can have pointers to class member functions and member variables.

- We can define pointer of class type, which can be used to point to class objects.

In-class Practice – Example 5

In-Class Exercise 1

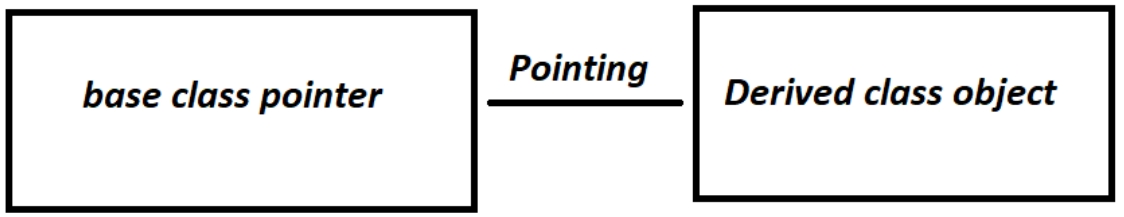
Create a class named Company with only one member variable, name. In the class, create the setter and getter function.

In the main function, create a Company object and a pointer to the object. Use both object and pointer to display the company name.

Pointer to C++ Class

- Pointer can also be used by a base class object to point to a derived class object.

- A pointer to the object of the derived class and a pointer to the object of the base class are type-compatible



Used to point the derived class object and pointer can still use aspects of base class

In-class Practice – Example 6

In-Class Exercise 2

Modify your Company class from Exercise 1 by having one more pure abstract function, displayInfo().

Create a HumanResource class that extends the Company class and has two member variables: numEmployee and workingHours. It should have one displayInfo() to display the information of number of employee and working hours. This function should override the displayInfo function in Company class.

Create a Marketing class that extends the Company class and has three member variables: numEmployee, workingHours and travelAllowance. It should have one displayInfo() to display the information of number of employee, working hours and travel allowance. This function should override the displayInfo function in Company class.

In the main function, create a HumanResource object and a Marketing object and a pointer to Company object. Use the Company object to point to the HumanResource object and Marketing object to display each of the department info.

See you next class