창의적 소프트웨어 프로그래밍 Lab 12

Handed out : Thu, Nov 3, 2022

Due : Mon, Nov 7, 2022, 23:59 (NO SCORE for late submissions!)

Submit your file on LMS.

1. Write a program that works as follows:
   A. Class C inherits from class B, class B inherits from class A.
   B. Each class has a public member function test().
      i. A::test() returns a string "A::test()".
      ii. B::test() returns a string "B::test()".
      iii. C::test() returns a string "C::test()".
   C. Create objects of class A, B, and C by new operator and put them into std::vector<A*> arr.
   D. Call the test() function of each element of arr to show the execution result as shown below. Each element of arr must be deallocated after use.
   E. Do not use the type casting operator throughout the code.
   F. Input: None
   G. Output: The result for calling test() functions
   H. Files to submit:
      i. A C++ source file

```
$ ./classes
A::test()
B::test()
C::test()
$
```

2. Write a program that works as follows:

   A. Class C inherits from class B, class B inherits from class A.

   B. Each class has a public member function getTypeInfo().

      i. A::getTypeInfo() returns a string of "This is an instance of class A".

      ii. B::getTypeInfo() returns a string of "This is an instance of class B".

      iii. C::getTypeInfo() returns a string of "This is an instance of class C".

   C. Define the following two functions in the global scope. Both functions print out strings obtained by calling getTypeInfo() of the object passed as argument.

      i. void printObjectTypeInfo1(A* object)

      ii. void printObjectTypeInfo2(A& object)

   D. Create objects of class A, B, and C by new operator and put them into std::vector<A*> arr.

   E. Call printObjectTypeInfo1() and printObjectTypeInfo2() by passing each element of arr as an argument. Each element of arr must be deallocated after use.

   F. Do not use the type casting operator throughout the code.

   G. Input: None

   H. Output: The result for printObjectTypeInfo1() and printObjectTypeInfo2()

   I. Files to submit:

      i. A C++ source file

```
$ ./print_info
This is an instance of class A
This is an instance of class A
This is an instance of class B
This is an instance of class B
This is an instance of class C
This is an instance of class C
$
```

3. Write a program that works as follows:

   A. Class C inherits from class B, class B inherits from class A.

   B. Add member variables.

      i. Add memberA, a private member variable of type int* to class A.

      ii. Add memberB, a private member variable of type double* to class B.

      iii. Add memberC, a private member variable of type std :: string* to class C.

   C. Constructors

      i. The constructor of class A takes an [int] argument, allocates memberA with new, stores the int argument value in the allocated space, and prints the string "new memberA".

      ii. The constructor of class B takes a [double] argument, allocates memberB using new, stores the double argument value in the allocated space, and prints out the string "new memberB". And it calls the constructor of class A from the initialization list, passing an integer 1 as the argument, to initialize memberA.

      iii. The constructor of class C takes a [const std::string&] type argument, allocates memberC with new, stores the string in the allocated space, and prints the string "new memberC". And it calls the constructor of class B from the initialization list, passing a double value 1.1 as the argument, to initialize memberB.

   D. Destructors

      i. The destructor of class A uses delete to free memberA and prints "delete memberA".

      ii. The destructor for class B uses delete to free memberB and prints "delete memberB".

      iii. The destructor of class C uses delete to free memberC and prints "delete memberC".

   E. A, B, and C all have member functions void print().

      i. A::print() prints out the data stored in the space pointed to by memberA.

ii. B::print() calls A::print() first, and then prints out the data stored in the space pointed to by memberB.

iii. C::print() calls B::print() first, and then prints out the data stored in the space pointed to by memberC.

F. Take an integer, real number, and string from the user, and then create objects of class A, B, and C by new operator with user inputs and put them into std::vector<A*> arr.

G. Call the print() function of each element of arr. Each element of arr must be deallocated after use.

H. Do not use the type casting operator throughout the code.

I. Input: None

J. Output: The result for print(), and creating and destructing objects.

K. Files to submit:

i. A C++ source file

```
$ ./print_member
20 3.14 test
new memberA
new memberA
new memberB
new memberA
new memberB
new memberC
*memberA 20
*memberA 1
*memberB 3.14
*memberA 1
*memberB 1.1
*memberC test
delete memberA
delete memberB
delete memberA
delete memberC
delete memberB
delete memberA
$
```