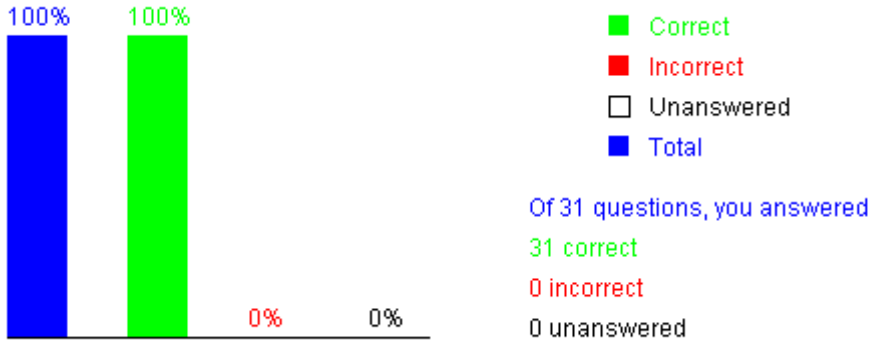
This quiz is for students to practice. A large number of additional quiz is available for instructors from the Instructor's Resource Website.

Chapter 15 Inheritance and Polymorphism



Please send suggestions and errata to Dr. Liang at y.daniel.liang@gmail.com. Indicate which book and edition you are using. Thanks!

Section 15.2 Base classes and Derived classes

15.1 Object-oriented programming allows you to derive new classes from existing classes. This is called _____.

- ( ) A. encapsulation
- (•) B. inheritance
- ( ) C. abstraction
- ( ) D. generalization

Your answer is correct ✓

15.2 Which of the following statements are true?

- [ ] A. A derived class is a subset of a base class.
- [x] B. A derived class is usually extended to contain more functions and more detailed information than its base class.
- [x] C. "class A: public B" means A is a derived class of B.
- [ ] D. "class A: public B" means B is a derived class of A.

Your answer is correct ✓

15.3 What is the output of the following code?

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

class ParentClass
{
public:
  int id;

  ParentClass(int id)
  {
    this->id = id;
  }

  void print()
  {
    cout << id << endl;
  }
};

class ChildClass: public ParentClass
{
public:
  int id;

  ChildClass(int id): ParentClass(1)
  {
```

```
    this->id = id;
  }
};

int main()
{
  ChildClass c(2);
  c.print();

  return 0;
}
```

- A. 0
- B. 1
- C. 2
- D. Nothing

Your answer is correct

Section 15.3 Generic Programming

15.4 Suppose Circle and Rectangle classes are derived from GeometricObject and you declared

```
void displayGeometricObject(GeometricObject shape)
{
  cout << shape.toString() << endl;
}
```

Which of the following function call is correct?

- [x] A. displayGeometricObject(GeometricObject("black", true));
- [x] B. displayGeometricObject(Circle(5));
- [x] C. displayGeometricObject(Rectangle(2, 3));
- [ ] D. displayGeometricObject(string());

Your answer is correct

Section 15.4 Constructors and Destructors

15.5 Are the constructors inherited by the derived class?

- A. Yes
- B. No

Your answer is correct

15.6 Are the destructors inherited by the derived class?

- A. Yes
- B. No

Your answer is correct

15.7 Suppose class A is derived from B and both A and B have no-arg constructors. To invoke B's constructor from A, use _____.

- A. A(): B() { ... }
- B. A(): { B(); ... }
- C. B(): A() { ... }
- D. B(): { A(); ... }

Your answer is correct

15.8 What is the output of the following code?

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

class B
{
public:
  ~B()
```

```
  {
    cout << "B";
  }
};

class A: public B
{
public:
  ~A()
  {
    cout << "A";
  }
};

int main()
{
  A a;
  return 0;
}
```

- (•) A. AB
- ( ) B. BA
- ( ) C. A
- ( ) D. B
- ( ) E. AA

Your answer is correct

15.9 What is wrong in the following code?

```
class Fruit
{
public:
  Fruit(int id)
  {
  }
};

class Apple: public Fruit
{
public:
  Apple()
  {
  }
};
```

- [x] A. The program will compile if you add a no-arg constructor for Fruit.
- [x] B. The program has a compile error because Fruit does not have a no-arg constructor.
- [x] C. The program will compile if you delete the constructor in Fruit.
- [x] D. The program will compile if you replace Apple() by Apple(): Fruit(4).

Your answer is correct

Section 15.5 Redefining Functions

15.10 Which of the following statements are true?

- [x] A. To redefine a function, the function must be defined in the derived class using the same signature and return type as in its base class.
- [x] B. Overloading a function is to provide more than one function with the same name but with different signatures to distinguish them.
- [x] C. It is a compilation error if two functions differ only in return type.
- [x] D. A private function cannot be redefined. If a function defined in a derived class is private in its base class, the two functions are completely unrelated.

Your answer is correct

15.11 Which of the following statements are true?

- [x] A. A function can be overloaded in the same class.
- [ ] B. A function can be redefined in the same class.
- [ ] C. If a function overloads another function, these two functions must have the same signature.

- [x] D. If a function redefines another function, these two functions must have the same signature.

Your answer is correct 

15.12 To invoke the toString() function defined in GeometricObject from a Circle object c, use _____.

- [ ] A. super.toString()
- [ ] B. c.super.toString()
- [x] C. c.GeometricObject::toString()
- [ ] D. c->GeometricObject::toString()

Your answer is correct 

Sections 15.7-15.8

15.13 What will be displayed by the following code?

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

class C
{
public:
  string toString()
  {
    return "C";
  }
};

class B: public C
{
  string toString()
  {
    return "B";
  }
};

class A: public B
{
  string toString()
  {
    return "A";
  }
};

void displayObject(C* p)
{
  cout << p->toString();
}

int main()
{
  displayObject(&A());
  displayObject(&B());
  displayObject(&C());
  return 0;
}
```

- [ ] A. ABC
- [ ] B. CBA
- [ ] C. AAA
- [ ] D. BBB
- [x] E. CCC

Your answer is correct 

15.14 What will be displayed by the following code?

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

class C
```

```
{
public:
  virtual string toString()
  {
    return "C";
  }
};

class B: public C
{
  string toString()
  {
    return "B";
  }
};

class A: public B
{
  string toString()
  {
    return "A";
  }
};

void displayObject(C* p)
{
  cout << p->toString();
}

int main()
{
  displayObject(&A());
  displayObject(&B());
  displayObject(&C());
  return 0;
}
```

- (●) A. ABC
- ( ) B. CBA
- ( ) C. AAA
- ( ) D. BBB
- ( ) E. CCC

Your answer is correct 

15.15 What will be displayed by the following code?

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

class C
{
public:
  string toString()
  {
    return "C";
  }
};

class B: public C
{
  string toString()
  {
    return "B";
  }
};

class A: public B
{
  virtual string toString()
  {
    return "A";
  }
};

void displayObject(C* p)
{
```

```
  cout << p->toString();
}

int main()
{
  displayObject(&A());
  displayObject(&B());
  displayObject(&C());
  return 0;
}
```

- A. ABC
- B. CBA
- C. AAA
- D. BBB
- E. CCC

Your answer is correct 

15.16 What will be displayed by the following code?

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

class C
{
public:
  virtual string toString()
  {
    return "C";
  }
};

class B: public C
{
  string toString()
  {
    return "B";
  }
};

class A: public B
{
  string toString()
  {
    return "A";
  }
};

void displayObject(C p)
{
  cout << p.toString();
}

int main()
{
  displayObject(A());
  displayObject(B());
  displayObject(C());
  return 0;
}
```

- A. ABC
- B. CBA
- C. AAA
- D. BBB
- E. CCC

Your answer is correct 

15.17 What is the output of the following code?

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

```
class Person
{
public:
  void printInfo()
  {
    cout << getInfo() << endl;
  }

  virtual string getInfo()
  {
    return "Person";
  }
};

class Student: public Person
{
public:
  virtual string getInfo()
  {
    return "Student";
  }
};

int main()
{
  Person().printInfo();
  Student().printInfo();
}
```

- A. Person Person
- B. Person Student
- C. Stdudent Student
- D. Student Person

Your answer is correct 

15.18 What is the output of the following code?

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

class Person
{
public:
  void printInfo()
  {
    cout << getInfo() << endl;
  }

  string getInfo()
  {
    return "Person";
  }
};

class Student: public Person
{
public:
  string getInfo()
  {
    return "Student";
  }
};

int main()
{
  Person().printInfo();
  Student().printInfo();
}
```

- A. Person Person
- B. Person Student
- C. Stdudent Student
- D. Student Person

Your answer is correct

15.19 If A is derived from B, and B is derived from C, B has a virtual function, will this function be dynamically binded?

- ( ) A. Yes
- (•) B. No

Your answer is correct

15.20 If the variable that references the object for the function is not the address of the object, will this function be dynamically binded?

- ( ) A. Yes
- (•) B. No

Your answer is correct

15.21 Which of the following statements are true?

- [x] A. If a function is defined virtual in a base class, it is automatically virtual in all its derived classes. It is not necessary to add the keyword virtual in the function declaration in the derived class.
- [x] B. If a function will not be redefined, it is more efficient without declaring it virtual, because it takes more time and system resource to bind virtual functions dynamically at runtime.
- [x] C. A virtual function may be implemented in several derived classes. C++ dynamically binds the implementation of the function at runtime, decided by the actual class of the object referenced by the variable.
- [x] D. The compiler finds a matching function according to parameter type, number of parameters, and order of the parameters at compile time.

Your answer is correct

15.22 Which of the following statements are true?

- [x] A. Private members can only be accessed from the inside of the class and public members can be accessed from any other classes.
- [x] B. A protected data field or a protected function in a base class can be accessed by name in its derived classes.
- [x] C. A public data field or function in a class can be accessed by name by any other program.

Your answer is correct

15.23 Analyze the following code:

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

class A
{
public:
  A()
  {
    t();
    cout << "i from A is " << i << endl;
  }

  void t()
  {
    setI(20);
  }

  virtual void setI(int i)
  {
    this->i = 2 * i;
  }

  int i;
};
```

```
class B: public A
{
public:
  B()
  {
    // cout << "i from B is " << i << endl;
  }

  virtual void setI(int i)
  {
    this->i = 3 * i;
  }
};

int main()
{
  A* p = new B();

  return 0;
}
```

- A. The constructor of class A is not called.
- B. The constructor of class A is called and it displays "i from A is 0".
- C. The constructor of class A is called and it displays "i from A is 40".
- D. The constructor of class A is called and it displays "i from A is 60".

Your answer is correct 

15.24 Analyze the following code:

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

class A
{
public:
  A()
  {
    t();
    // cout << "i from A is " << i << endl;
  }

  void t()
  {
    setI(20);
  }

  virtual void setI(int i)
  {
    this->i = 2 * i;
  }

  int i;
};

class B: public A
{
public:
  B()
  {
    cout << "i from B is " << i << endl;
  }

  virtual void setI(int i)
  {
    this->i = 3 * i;
  }
};

int main()
{
  A* p = new B();

  return 0;
}
```

- A. The constructor of class A is not called.

B. The constructor of class A is called and it displays "i from B is 0".

C. The constructor of class A is called and it displays "i from B is 40".

D. The constructor of class A is called and it displays "i from B is 60".

Your answer is correct

Section 15.9 Abstract Classes and Pure Virtual Functions

15.25 Which of the following is an abstract function?

A. virtual double getArea();

B. virtual double getArea() = 0;

C. double getArea() = 0;

D. double getArea();

Your answer is correct

15.26 Which of the following statements are true?

A. An abstract class is declared using a keyword abstract.

B. A class is abstract if it contains a pure virtual function.

C. An abstract class is like a regular class except that you cannot create objects from it.

D. You can declare a class abstract even though it does not contain abstract functions.

Your answer is correct

Section 15.10 Casting: static_cast versus dynamic_cast

15.27 Which of the following statements are true?

A. Assigning a pointer of a derived class type to a pointer of its base class type is called upcasting.

B. Assigning a pointer of a base class type to a pointer of its derived class type is called downcasting.

C. Upcasting can be performed implicitly without using the dynamic_cast operator.

D. downcasting must be performed explicitly using the dynamic_cast operator.

Your answer is correct

15.28 Suppose you declared GeometricObject* p = &object. To cast p to Circle, use _____.

A. Circle* p1 = dynamic_cast<Circle>(p);

B. Circle* p1 = dynamic_cast<Circle*>(p);

C. Circle p1 = dynamic_cast<Circle*>(p);

D. Circle p1 = dynamic_cast<Circle>(p);

Your answer is correct

15.29 What will be displayed by the following code?

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

class A
{
public:
  string toString()
  {
    return "A";
  }

};

class B: public A
{
public:
  string toString()
  {
```

```
    return "B";
  }
};

int main()
{
  A* b = new B();

  cout << static_cast<A*>(b)->toString() << b->toString() << endl;

  return 0;
}
```

- (•) A. AA
- ( ) B. AB
- ( ) C. BA
- ( ) D. BB

Your answer is correct 

15.30 What will be displayed by the following code?

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

class A
{
public:
  string toString()
  {
    return "A";
  }

};

class B: public A
{
public:
  string toString()
  {
    return "B";
  }
};

int main()
{
  B b;
  cout << static_cast<A>(b).toString() << b.toString() << endl;

  return 0;
}
```

- ( ) A. AA
- (•) B. AB
- ( ) C. BA
- ( ) D. BB

Your answer is correct 

15.31 Choose the correct answers to address the issues in the followin code:

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

class A
{
public:
  _Place1_____ string toString()
  {
    return "A";
  }

};
```

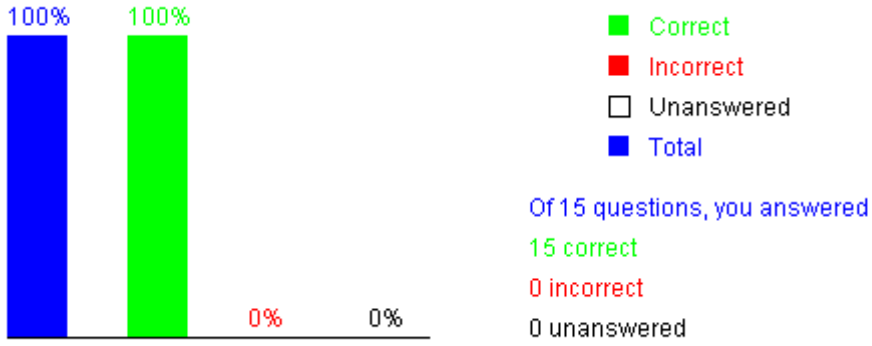
```
class B: public A
{
public:
  _Place2________ string toString()
  {
    return "B";
  }
};

int main()
{
  A* b = new B();

  cout << dynamic_cast<B*>(b)->toString() << endl;

  return 0;
}
```

- [ ] A. With blank for Place1 and Place2, the program will compile and run fine.
- [ ] B. With Place1 replaced by virtual and Place2 blank, the program will compile, but not run.
- [x] C. With Place1 replaced by virtual and Place2 blank, the program will compile and run.
- [x] D. With Place1 replaced by virtual and Place2 virtual, the program will compile and run.

Your answer is correct 

This quiz is for students to practice. A large number of additional quiz is available for instructors from the Instructor's Resource Website.

Chapter 16 Exception Handling



Please send suggestions and errata to Dr. Liang at y.daniel.liang@gmail.com. Indicate which book and edition you are using. Thanks!

Section 16.2 Exception-Handling Overview

16.1 Which of the following statements are correct?

- [x] A. The execution of a throw statement is called throwing an exception.
- [x] B. You can throw a value of any type.
- [x] C. The try block contains the code that are executed in normal circumstances.
- [x] D. The catch block contains the code that are executed when an exception occurs.

Your answer is correct

16.2 What is wrong in the following code?

```
vector<int> v;
v[0] = 2.5;
```

- ( ) A. The program has a compile error because there are no elements in the vector.
- ( ) B. The program has a compile error because you cannot assign a double value to v[0].
- (•) C. The program has a runtime error because there are no elements in the vector.
- ( ) D. The program has a runtime error because you cannot assign a double value to v[0].

Your answer is correct

16.3 If you are not interested in the contents of an exception object, the catch block parameter may be omitted.

- (•) A. true
- ( ) B. false

Your answer is correct

16.4 If you enter 1 0, what is the output of the following code?

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

int main()
{
  // Read two integers
  cout << "Enter two integers: ";
  int number1, number2;
  cin >> number1 >> number2;

  try
  {
    if (number2 == 0)
      throw number1;

    cout << number1 << " / " << number2 << " is "
```

```
        << (number1 / number2) << endl;

      cout << "C" << endl;
    }
    catch (int e)
    {
      cout << "A" << endl;
    }

    cout << "B" << endl;

    return 0;
}
```

- A. A
- B. B
- C. C
- D. AB

Your answer is correct

16.5 catch (type p) acts very much like a parameter in a function. Once the exception is caught, you can access the thrown value from this parameter in the body of a catch block.

- A. true
- B. false

Your answer is correct

Section 16.4 Exception Classes

16.6 Which of the following classes are predefined in C++?

- [x] A. exception
- [x] B. runtime_error
- [x] C. overflow_error
- [x] D. underflow_error
- [x] E. bad_exception

Your answer is correct

16.7 Which of the following classes are in the header file <stdexcept>?

- [ ] A. exception
- [x] B. runtime_error
- [x] C. overflow_error
- [x] D. underflow_error
- [x] E. bad_exception

Your answer is correct

16.8 Which of the following classes are in the header file <stdexcept>?

- [x] A. logic_error
- [x] B. invalid_argument
- [x] C. length_error
- [x] D. out_of_range
- [x] E. bad_cast

Your answer is correct

16.9 The function what() is defined in _____.

- A. exception
- B. runtime_error
- C. overflow_error
- D. underflow_error
- E. bad_exception

Your answer is correct

Section 16.5 Custom Exception Classes

16.10 Which of the following statements are true?

- (•) A. A custom exception class is just like a regular class.
- ( ) B. A custom exception class must always be derived from class exception.
- ( ) C. A custom exception class must always be derived from a derived class of class exception.
- ( ) D. A custom exception class must always be derived from class runtime_error.

Your answer is correct

Section 16.6 Multiple Catches

16.11 Suppose Exception2 is derived from Exception1. Analyze the following code.

```
try {
  statement1;
  statement2;
  statement3;
}
catch (Exception1 ex1)
{
}
catch (Exception2 ex2)
{
}
```

- (•) A. If an exception of the Exeception2 type occurs, this exception is caught by the first catch block.
- ( ) B. If an exception of the Exeception2 type occurs, this exception is caught by the second catch block.
- ( ) C. The program has a compile error because these two catch blocks are in wrong order.
- ( ) D. The program has a runtime error because these two catch blocks are in wrong order.

Your answer is correct

Section 16.8 Rethrowing Exceptions

16.12 Suppose that statement2 throws an exception of type Exception2 in the following statement:

```
try {
  statement1;
  statement2;
  statement3;
}
catch (Exception1 ex1)
{
}
catch (Exception2 ex2)
{
}
catch (Exception3 ex3)
{
  statement4;
  throw;
}
statement5;
```

Which statements are executed after statement2 is executed?

- ( ) A. statement1
- ( ) B. statement2
- ( ) C. statement3
- ( ) D. statement4
- (•) E. statement5

Your answer is correct

16.13 Suppose that statement3 throws an exception of type Exception3 in the following statement:

```
try {
  statement1;
  statement2;
  statement3;
}
catch (Exception1 ex1)
{
}
catch (Exception2 ex2)
{
}
catch (Exception3 ex3)
{
  statement4;
  throw;
}
statement5;
```

Which statements are executed after statement3 is executed?

- ( ) A. statement1
- ( ) B. statement2
- ( ) C. statement3
- (•) D. statement4
- ( ) E. statement5

Your answer is correct 

Section 16.9 Exception Specification

16.14 Which of the following statements are true?

- [x] A. A function should warn the programmers that any exceptions it might throw, so the programmers can write robust program to deal with these potential exceptions in a try-catch block.
- [x] B. If a function is declared as returnType functionName(parameterList) throw (type), this function can only throw the exception of the specified type.
- [x] C. Placing throw() after a function header, known as an empty exception specification, declares that the function does not throw any exceptions.
- [x] D. Throwing an exception that is not declared in the throw list will causes function unexpected to be invoked.
- [x] E. A function without exception specification can throw any exception and will not cause unexpected to be invoked.

Your answer is correct 

Section 16.10 When to Use Exceptions

16.15 Which of the following statements are true?

- [x] A. C++ allows you to throw a primitive type value or any object-type value.
- [x] B. In general, common exceptions that may occur in multiple classes in a project are candidates for exception classes.
- [x] C. Simple errors that may occur in individual functions are best handled locally without throwing exceptions.
- [x] D. Exception handling is for dealing with unexpected error conditions. Do not use a try-catch block to deal with simple, expected situations.

Your answer is correct