Module 03-面向对象编程

一、选择题：

Question 1
Given:

```
public class CreditCard {

private String cardlD;
private Integer limit;
public String ownerName;

public void setCardlnformation(String cardlD,
String ownerName,
Integer limit) {
this.cardlD = cardlD;
this.ownerName = ownerName;
this.limit = limit;
}
}
```

Which is true?
A. The class is fully encapsulated.
B. The code demonstrates polymorphism.
C. The ownerName variable breaks encapsulation.
D. The cardlD and limit variables break polymorphism.
E. The setCardlnformation method breaks encapsulation.

Answer: C

Question 2
Which two are true? (Choose two.)
A. An encapsulated, public class promotes re-use.
B. Classes that share the same interface are always tightly encapsulated.
C. An encapsulated class allows subclasses to overload methods, but does NOT allow overriding methods.
D. An encapsulated class allows a programmer to change an implementation without affecting outside code.

Answer: AD

Question 3
Assume that country is set for each class.
Given:

```
10. public class Money {
11. private String country, name;
12. public getCountry() { return country; }
13.}
```
and:
```
24. class Yen extends Money {
25. public String getCountry() { return super.country; }
26. }
27.
28. class Euro extends Money {
29. public String getCountry(String timeZone) {
30. return super.getCountry();
31. }
32. }
```
Which two are correct? (Choose two.)
A. Yen returns correct values.
B. Euro returns correct values.
C. An exception is thrown at runtime.
D. Yen and Euro both return correct values.
E. Compilation fails because of an error at line 25.
F. Compilation fails because of an error at line 30.

Answer: BE

Question 4
Given:
```
10. interface A { void x(); }
11. class B implements A { public void x() { } public void y() { } }
12. class C extends B { public void x() {} }
```
And:
```
20. java.util.List<A> list = new java.util.ArrayList<A>();
21. list.add(new B());
22. list.add(new C());
23. for (A a:list) {
24. a.x();
25. a.y();
26. }
```
What is the result?
A. The code runs with no output.
B. An exception is thrown at runtime.
C. Compilation fails because of an error in line 20.
D. Compilation fails because of an error in line 21.
E. Compilation fails because of an error in line 23.
F. Compilation fails because of an error in line 25.

Answer: F

Question 5
Given:
```
1. class SuperClass {
2. public A getA() {
3. return new A();
4. }
5. }
6. class SubClass extends SuperClass {
7. public B getA() {
8. return new B();
9. }
10. }
```
Which is true?
A. Compilation will succeed if A extends B.
B. Compilation will succeed if B extends A.
C. Compilation will always fail because of an error in line 7.
D. Compilation will always fail because of an error in line 8.

Answer: B

Question 6
Given:
```
1. interface A { public void aMethod(); }
2. interface B { public void bMethod(); }
3. interface C extends A,B { public void cMethod(); }
4. class D implements B {
5. public void bMethod() { }
6. }
7. class E extends D implements C {
8. public void aMethod() { }
9. public void bMethod() { }
10. public void cMethod() { }
11. }
```
What is the result?
A. Compilation fails because of an error in line 3.
B. Compilation fails because of an error in line 7.
C. Compilation fails because of an error in line 9.
D. If you define D e = new E(), then e.bMethod() invokes the version of bMethod() defined in Line 5.
E. If you define D e = (D)(new E()), then e.bMethod() invokes the version of bMethod() defined in Line 5.
F. If you define D e = (D)(new E()), then e.bMethod() invokes the version of bMethod() defined in Line 9.

Answer: F

Question 7
Given:

```
public class Base {
public static final String FOO ="foo";
public static void main(String[] args) {
Base b = new Base();
Sub s = new Sub();
System.out.print(Base.FOO);
System.out.print(Sub.FOO);
System.out.print(b.FOO);
System.out.print(s.FOO);
System.out.print(((Base)s).FOO);
} }
class Sub extends Base {public static final String FOO="bar";}
```

What is the result?
A. foofoofoofoofoo
B. foobarfoobarbar
C. foobarfoofoofoo
D. foobarfoobarfoo
E. barbarbarbarbar
F. foofoofoobarbar
G. foofoofoobarfoo

Answer: D

Question 8
Given:

```
class Pizza {
java.util.ArrayList toppings;
public final void addTopping(String topping) {
toppings.add(topping);
}
}
public class PepperoniPizza extends Pizza {
public void addTopping(String topping) {
System.out.println("Cannot add Toppings");
}
public static void main(String[] args) {
Pizza pizza = new PepperoniPizza();
pizza.addTopping("Mushrooms");
}
}
```

What is the result?
A. Compilation fails.
B. Cannot add Toppings
C. The code runs with no output.

D. A NullPointerException is thrown in Line 4.

Answer: A

Question 9
Given:
```
10. public class Foo {
11. public int a;
12. public Foo() { a = 3; }
13. public void addFive() { a += 5; }
14. }
```
and:
```
20. public class Bar extends Foo {
21. public int a;
22. public Bar() { a = 8; }
23. public void addFive() { this.a +=5; }
24. }
```
invoked with:
```
30. Foo foo = new Bar();
31. foo.addFive();
32. System.out.println("Value: "+ foo.a);
```
What is the result?
A. Value: 3
B. Value: 8
C. Value: 13
D. Compilation fails.
E. The code runs with no output.
F. An exception is thrown at runtime.

Answer: A

Question 10
Given:
```
10. public class SuperCalc {
11. protected static int multiply(int a, int b) { return a * b; }
12. }
```
and:
```
20. public class SubCalc extends SuperCalc {
21. public static int multiply(int a, int b) {
22. int c = super.multiply(a, b);
23. return c;
24. }
25. }
```
and:
```
30. SubCalc sc = new SubCalc();
31. System.out.println(sc.multiply(3,4));
```

32. System.out.println(SubCalc.multiply(2,2));
What is the result?

A. 12   4
B. The code runs with no output.
C. An exception is thrown at runtime.
D. Compilation fails because of an error in line 21.
E. Compilation fails because of an error in line 22.
F. Compilation fails because of an error in line 31.

Answer: E

Question 11
Given:
1. public class Team extends java.util.LinkedList {
2. public void addPlayer(Player p) {
3. add(p);
4. }
5. public void compete(Team opponent) { /* more code here */ }
6. }
7. class Player { /* more code here */ }
Which two are true? (Choose two.)

A. This code will compile.
B. This code demonstrates proper design of an is-a relationship.
C. This code demonstrates proper design of a has-a relationship.
D. A Java programmer using the Team class could remove Player objects from a Team object.

Answer: AD

Question 12
Given:
11. class ClassA {}
12. class ClassB extends ClassA {}
13. class ClassC extends ClassA {}
and:
21. ClassA p0 = new ClassA();
22. ClassB p1 = new ClassB();
23. ClassC p2 = new ClassC();
24. ClassA p3 = new ClassB();
25. ClassA p4 = new ClassC();
Which three are valid? (Choose three.)
A. p0 = p1;
B. p1 =p2;
C. p2 = p4;

D. p2 = (ClassC)p1;
E. p1 = (ClassB)p3;
F. p2 = (ClassC)p4;

Answer: AEF

Question 13
Given:

```
11. class Animal { public String noise() { return "peep"; } }
12. class Dog extends Animal {
13. public String noise() { return "bark"; }
14. }
15. class Cat extends Animal {
16. public String noise() { return "meow"; }
17. }
.....
30. Animal animal = new Dog();
31. Cat cat = (Cat)animal;
32. System.out.printIn(cat.noise());
```

What is the result?
A. peep
B. bark
C. meow
D. Compilation fails.
E. An exception is thrown at runtime.

Answer: E

Question 14
Given:

```
11. class Cup { }
12. class PoisonCup extends Cup { }
21. public void takeCup(Cup c) {
22. if(c instanceof PoisonCup) {
23. System.out.println("Inconceivable!");
24. } else if(c instanceof Cup) {
25. System.out.println("Dizzying intellect!");
26. } else {
27. System.exit(0);
28. }
29. }
```

And the execution of the statements:

```
Cup cup = new PoisonCup();
takeCup(cup);
```

What is the output?

A. Inconceivable!
B. Dizzying intellect!
C. The code runs with no output.
D. An exception is thrown at runtime.
E. Compilation fails because of an error in line 22.

Answer: A

Question 15
Click the Exhibit button.

```
1. public class SimpleCalc {
2. public int value;
3. public void calculate() { value += 7; }
4. }
```

And:

```
1. public class MultiCalc extends SimpleCalc {
2. public void calculate() { value -= 3; }
3. public void calculate(int multiplier) {
4. calculate();
5. super.calculate();
6. value *=multiplier;
7. }
8. public static void main(String[] args) {
9. MultiCalc calculator = new MultiCalc();
10. calculator.calculate(2);
11. System.out.println("Value is: "+ calculator.value);
12. }
13. }
```

What is the result?
A. Value is: 8
B. Compilation fails.
C. Value is: 12
D. Value is: -12
E. The code runs with no output.
F. An exception is thrown at runtime.

Answer: A

Question 16
Given:

```
1. public class Blip {
2. protected int blipvert(int x) { return 0; }
3. }
4. class Vert extends Blip {
5. // insert code here
6. }
```

Which five methods, inserted independently at line 5, will compile?
(Choose five.)
A. public int blipvert(int x) { return 0; }
B. private int blipvert(int x) { return 0; }
C. private int blipvert(long x) { return 0; }
D. protected long blipvert(int x) { return 0; }
E. protected int blipvert(long x) { return 0; }
F. protected long blipvert(long x) { return 0; }
G. protected long blipvert(int x, int y) { return 0; }

Answer: ACEFG

Question 17
Given:

```
11. abstract class Vehicle { public int speed() { return 0; } }
12. class Car extends Vehicle { public int speed() { return 60; } }
13. class RaceCar extends Car { public int speed() { return 150; }}
......
21. RaceCar racer = new RaceCar();
22. Car car = new RaceCar();
23. Vehicle vehicle = new RaceCar();
24. System.out.println(racer.speed() + ", " + car.speed()
25. + ", "+ vehicle.speed());
```

What is the result?
A. 0, 0,0
B. 150, 60, 0
C. Compilation fails.
D. 150, 150, 150
E. An exception is thrown at runtime.

Answer: D

Question 18
Given:

```
10. interface A { public int getValue() }
11. class B implements A {
12. public int getValue() { return 1; }
13. }
14. class C extends B {
15. // insert code here
16. }
```

Which three code fragments, inserted individually at line 15, make use of polymorphism? (Choose three.)
A. public void add(C c) { c.getValue(); }
B. public void add(B b) { b.getValue(); }
C. public void add(A a) { a.getValue(); }

D. public void add(A a, B b) { a.getValue(); }
E. public void add(C c1, C c2) { c1.getValue(); }

Answer: BCD

Question 19
Which three statements are true? (Choose three.)
A. A final method in class X can be abstract if and only if X is abstract.
B. A protected method in class X can be overridden by any subclass of X.
C. A private static method can be called only within other static methods in class X.
D. A non-static public final method in class X can be overridden in any subclass of X.
E. A public static method in class X can be called by a subclass of X without explicitly referencing the class X.
F. A method with the same signature as a private final method in class X can be implemented in a subclass of X.
G. A protected method in class X can be overridden by a subclass of A only if the subclass is in the same package as X.

Answer: BEF

Question 20
Given:
```
abstract class A {
abstract void al();
void a2() { }
}
class B extends A {
void a1() { }
void a2() { }
}
class C extends B { void c1() { } }
```
and:
A x = new B(); C y = new C(); A z = new C();
Which four are valid examples of polymorphic method calls? (Choose four.)
A. x.a2();
B. z.a2();
C. z.c1();
D. z.a1();
E. y.c1();
F. x.a1();

Answer: ABDF

Question 21
Click the Exhibit button.

```
1. public class GoTest {
2. public static void main(String[] args) {
3. Sente a = new Sente(); a.go();
4. Goban b = new Goban(); b.go();
5. Stone c = new Stone(); c.go();
6. }
7. }
8.
9. class Sente implements Go {
10. public void go() { System.out.println("go in Sente."); }
11. }
12.
13. class Goban extends Sente {
14. public void go() { System.out.println("go in Goban"); }
15. }
16.
17. class Stone extends Goban implements Go { }
18.
19. interface Go { public void go(); }
```

What is the result?

A. go in Goban
go in Sente
go in Sente

B. go in Sente
go in Sente
go in Goban

C. go in Sente
go in Goban
go in Goban

D. go in Goban
go in Goban
go in Sente

E. Compilation fails because of an error in line 17.

Answer: C

Question 22
Given:

```
1. class ClassA {
```

```
2. public int numberOfinstances;
3. protected ClassA(int numberOfinstances) {
4. this.numberOflnstances = numberOfinstances;
5. }
6. }
7. public class ExtendedA extends ClassA {
8. private ExtendedA(int numberOfInstances) {
9. super(numberOfiInstances);
10. }
11. public static void main(String[] args) {
12. ExtendedA ext = new ExtendedA(420);
13. System.out.print(ext.numberOfInstances);
14. }
15. }
```

Which is true?
A. 420 is the output.
B. An exception is thrown at runtime.
C. All constructors must be declared public.
D. Constructors CANNOT use the private modifier.
E. Constructors CANNOT use the protected modifier.

Answer: A

Question 23
Given:

```
1. class Super {
2. private int a;
3. protected Super(int a) { this.a = a; }
4. }
.....
11. class Sub extends Super {
12. public Sub(int a) { super(a); }
13. public Sub() { this.a= 5; }
14. }
```

Which two, independently, will allow Sub to compile? (Choose two.)

A. Change line 2 to:
```
     public int a;
```

B. Change line 2 to:
```
     protected int a;
```

C. Change line 13 to:
```
     public Sub() { this(5); }
```

D. Change line 13 to:

```
public Sub() { super(5); }
```

E. Change line 13 to:
```
public Sub() { super(a); }
```

Answer: CD

Question 24
Given:
```
10. public class Hello {
11. String title;
12. int value;
13. public Hello() {
14. title += "World";
15. }
16. public Hello(int value) {
17. this.value = value;
18. title = "Hello";
19. Hello();
20. }
21. }
```
and:
```
30. Hello c = new Hello(5);
31. System.out.println(c.title);
```
What is the result?
A. Hello
B. Hello World
C. Compilation fails.
D. Hello World 5
E. The code runs with no output.
F. An exception is thrown at runtime.

Answer: C

Question 25
Click the Exhibit button.
```
1. public class Car {
2. private int wheelCount;
3. private String vin;
4. public Car(String vin) {
5. this.vin = vin;
6. this.wheelCount = 4;
7. }
8. public String drive() {
9. return "zoom-zoom";
10. }
11. public String getInfo() {
12. return "VIN: "+ vin + "wheels: "+ wheelCount;
13. }
14. }
```
And:
```
1. public class MeGo extends Car {
2. public MeGo(String vin) {
3. this.wheelCount = 3;
4. }
5. }
```
What two must the programmer do to correct the compilation errors?
(Choose two.)
A. insert a call to this() in the Car constructor
B. insert a call to this() in the MeGo constructor
C. insert a call to super() in the MeGo constructor
D. insert a call to super(vin) in the MeGo constructor
E. change the wheelCount variable in Car to protected
F. change line 3 in the MeGo class to super.wheelCount = 3;

Answer: DE

Question 26
Click the Exhibit button.
```
1. public class Employee {
2. String name;
3. double baseSalary;
4. Employee(String name, double baseSalary) {
5. this.name = name;
6. this.baseSalary = baseSalary;
7. }
8. }
```
And:
```
1. public class Salesperson extends Employee {
2. double commission;
3. public Salesperson(String name, double baseSalary,
4. double commission) {
5. // insert code here
6. }
7. }
```
Which code, inserted at line 7, completes the Salesperson constructor?

A. this.commission = commission;
B. superb();
    commission = commission;
C. this.commission = commission;
    superb();

D. super(name, baseSalary);
    this.commission = commission;

E. super();
    this.commission = commission;

F. this.commission = commission;
    super(name, baseSalary);

Answer: D

Question 27
Which Man class properly represents the relationship “Man has a best friend who is a Dog”?
A. class Man extends Dog { }
B. class Man implements Dog { }
C. class Man { private BestFriend dog; }
D. class Man { private Dog bestFriend; }
E. class Man { private Dog<bestFriend> }
F. class Man { private BestFriend<dog> }

Answer: D

Question 28
Which four are true? (Choose four.)
A. Has-a relationships should never be encapsulated.
B. Has-a relationships should be implemented using inheritance.
C. Has-a relationships can be implemented using instance variables.
D. Is-a relationships can be implemented using the extends keyword.
E. Is-a relationships can be implemented using the implements keyword.
F. The relationship between Movie and Actress is an example of an is-a relationship.
G. An array or a collection can be used to implement a one-to-many has-a relationship.

Answer: CDEG

Question 29
Which two are true about has-a and is-a relationships? (Choose two.)
A. Inheritance represents an is-a relationship.
B. Inheritance represents a has-a relationship.
C. Interfaces must be used when creating a has-a relationship.
D. Instance variables can be used when creating a has-a relationship.

Answer: AD

Question 30
Given:

```
10. interface Jumper { public void jump(); }
......
20. class Animal {}
......
30. class Dog extends Animal {
31. Tail tail;
32. }
......
40. class Beagle extends Dog implements Jumper {
41. public void jump() { }
42. }
.......
50. class Cat implements Jumper {
51. public void jump() { }
52. }
```

Which three are true? (Choose three.)
A. Cat is-a Animal
B. Cat is-a Jumper
C. Dog is-a Animal
D. Dog is-a Jumper
E. Cat has-a Animal
F. Beagle has-a Tail
G. Beagle has-a Jumper

Answer: BCF

Question 31
Given:

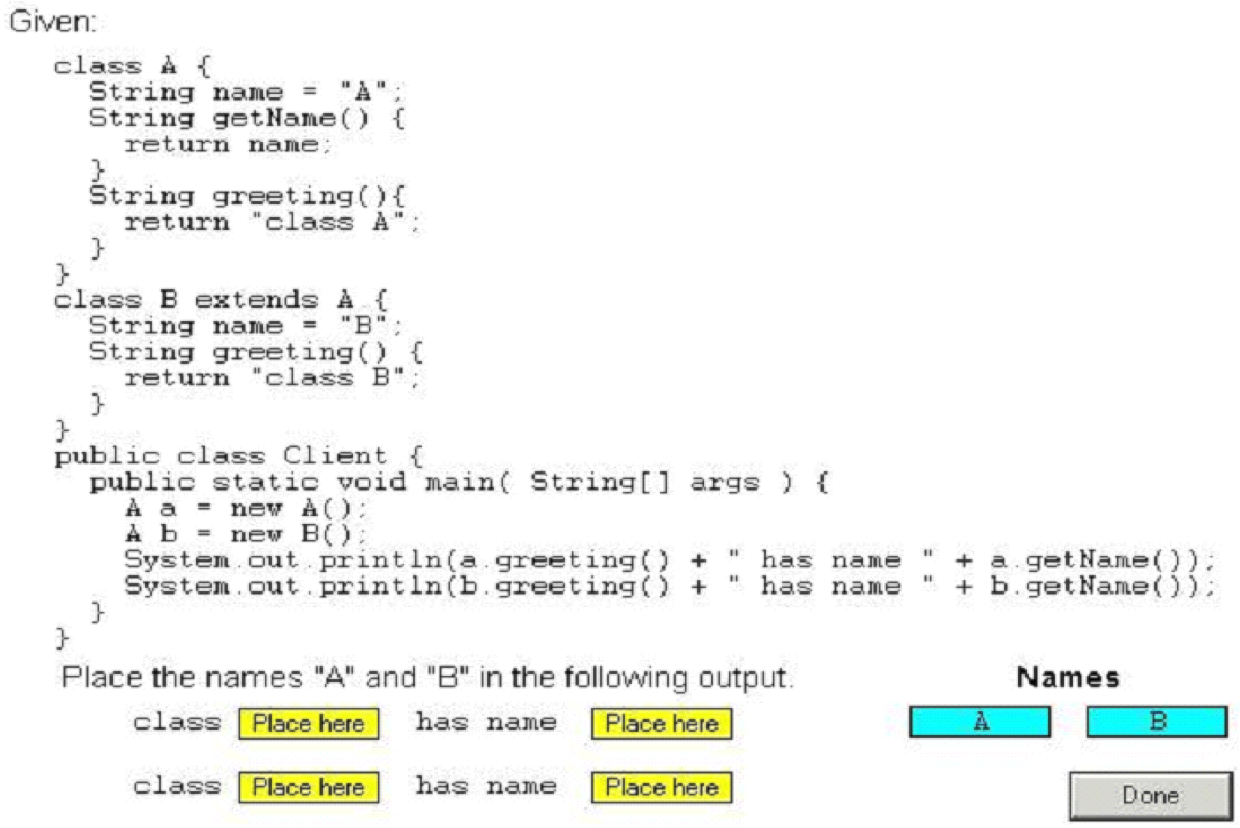
```
1. package geometry;
2. public class Hypotenuse {
3. public InnerTriangle it = new InnerTriangle();
4. class InnerTriangle {
5. public int base;
6. public int height;
7. }
8. }
```

Which is true about the class of an object that can reference the variable base?
A. It can be any class.
B. No class has access to base.
C. The class must belong to the geometry package.
D. The class must be a subclass of the class Hypotenuse.

Answer: C

二、拖拽题：

Question 1:



Given:

```
class A {
  String name = "A";
  String getName() {
    return name;
  }
  String greeting(){
    return "class A";
  }
}
class B extends A {
  String name = "B";
  String greeting() {
    return "class B";
  }
}
public class Client {
  public static void main( String[] args ) {
    A a = new A();
    A b = new B();
    System.out.println(a.greeting() + " has name " + a.getName());
    System.out.println(b.greeting() + " has name " + b.getName());
  }
}
```

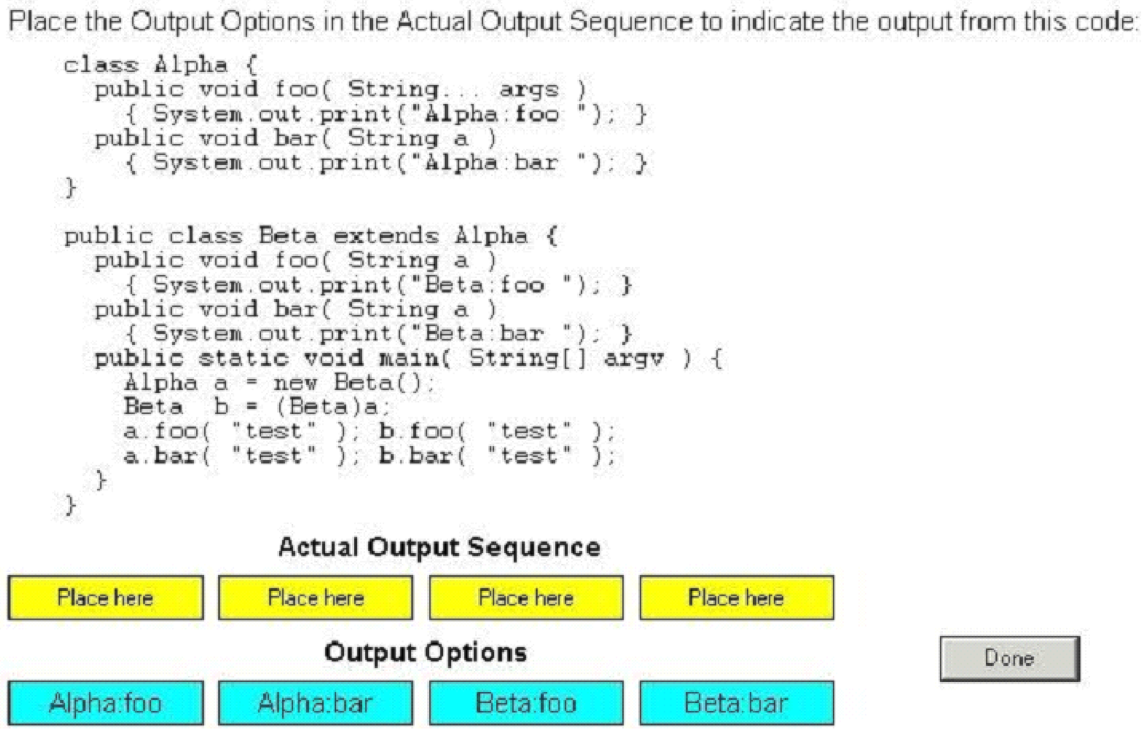
Place the names "A" and "B" in the following output.

```
class  Place here  has name  Place here

class  Place here  has name  Place here
```

Answer:

class A has name A
class B has name A

Question 2:



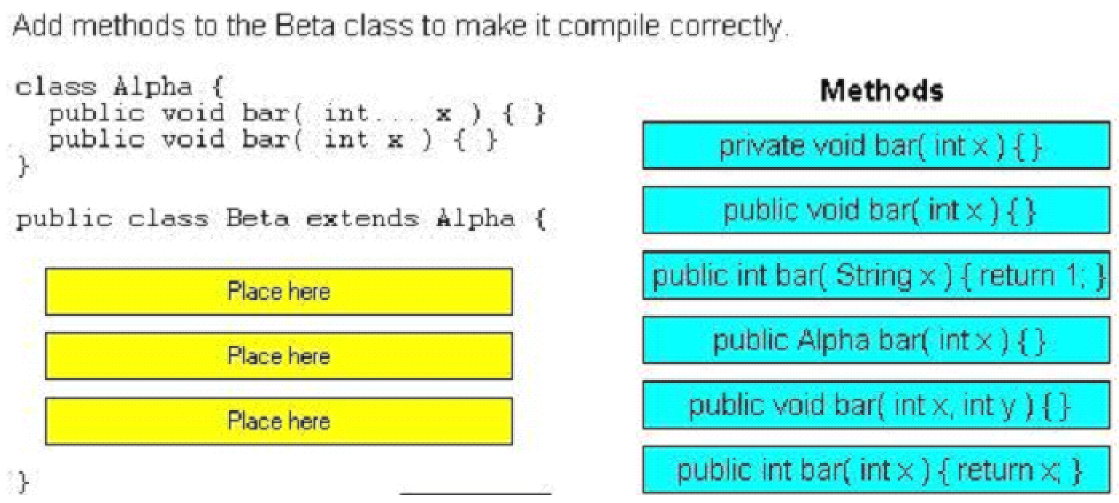
Place the Output Options in the Actual Output Sequence to indicate the output from this code:

```
class Alpha {
  public void foo( String... args )
    { System.out.print("Alpha:foo "); }
  public void bar( String a )
    { System.out.print("Alpha:bar "); }
}

public class Beta extends Alpha {
  public void foo( String a )
    { System.out.print("Beta:foo "); }
  public void bar( String a )
    { System.out.print("Beta:bar "); }
  public static void main( String[] argv ) {
    Alpha a = new Beta();
    Beta  b = (Beta)a;
    a.foo( "test" ); b.foo( "test" );
    a.bar( "test" ); b.bar( "test" );
  }
}
```

Answer:

Alpha:foo Beta:foo Beta:bar Beta:bar

Question 3:



Add methods to the Beta class to make it compile correctly.

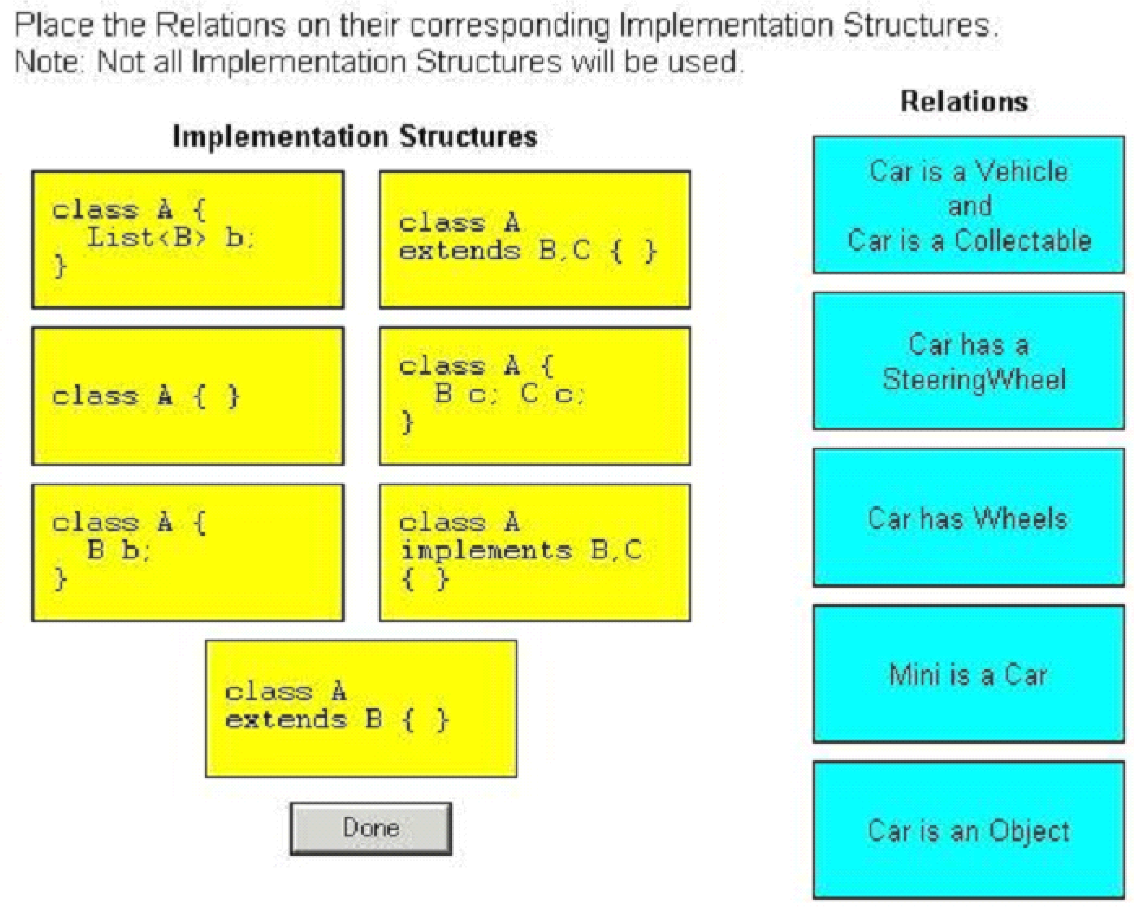
```
class Alpha {
  public void bar( int... x ) { }
  public void bar( int x ) { }
}

public class Beta extends Alpha {

        Place here

        Place here

        Place here

}
```

```
private void bar( int x ) { }
public void bar( int x ) { }
public int bar( String x ) { return 1; }
public Alpha bar( int x ) { }
public void bar( int x, int y ) { }
public int bar( int x ) { return x; }
```

Answer:

```
public class Beta extends Alpha{
        public void bar(int x){}
        public int bar(String x){return 1;}
        public void bar(int x,int y){}
}
```

Question 4:



Place the Relations on their corresponding Implementation Structures.
Note: Not all Implementation Structures will be used.

Implementation Structures

```
class A {
  List<B> b;
}
```

```
class A
extends B,C { }
```

```
class A { }
```

```
class A {
  B c; C c;
}
```

```
class A {
  B b;
}
```

```
class A
implements B,C
{ }
```

```
class A
extends B { }
```

Done

Relations

Car is a Vehicle and Car is a Collectable

Car has a SteeringWheel

Car has Wheels

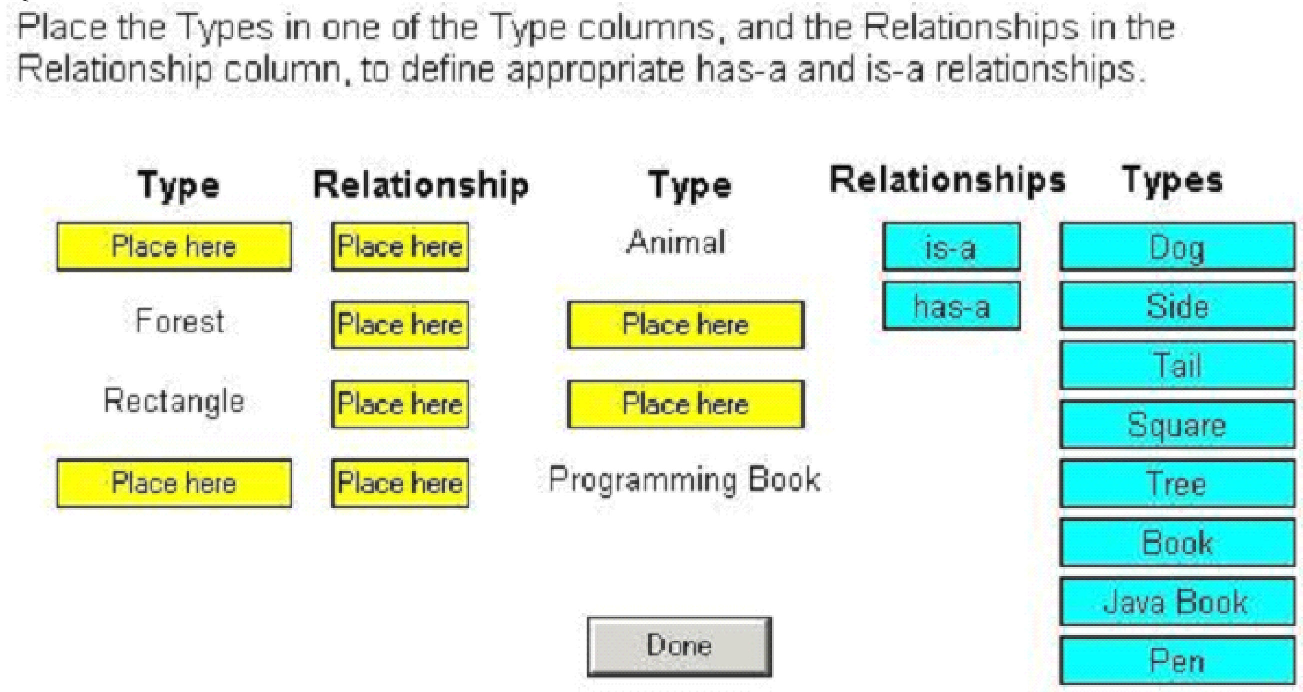
Mini is a Car

Car is an Object

Answer:

```
class A{ List<B> b;}            //Car has Wheels
class A extends B,C{}
class A{}                       //Car is an Object
class A{B c; C c;}
class A{B b;}                   //Car has a Steering Wheels
class A implements B,C{}        //Car is Vehicle And Car is a Collectable
class A extends B{}             //Mini is a Car
```

Question 5:



Place the Types in one of the Type columns, and the Relationships in the Relationship column, to define appropriate has-a and is-a relationships.

Type	Relationship	Type	Relationships	Types
Place here	Place here	Animal	is-a	Dog
Forest	Place here	Place here	has-a	Side
Rectangle	Place here	Place here		Tail
Place here	Place here	Programming Book		Square
				Tree
				Book
				Java Book
				Pen

Answer:

Dog	is-a	Animal
Forest	has-a	Tree
Rectangle	has-a	Square
Java Book	is-a	Programming Book