> My Courses > OCAJP 8 Training Course -1Z0 808 > Quiz : Working with Inheritance > **Report**

Search Courses

Quiz : Working with Inheritance

Attempt

1

Completed on

Wednesday , 04 September 2019 , 06:35 PM

Marks Obtained

0 / 13

Time Taken

00 H 00 M 07 S

Your score is

0.0%

Result

Fail

Mode

Exam

Domains wise Quiz Performance Report

No	1
Domain	Other
Total Question	13
Correct	0
Incorrect	0
Unattempted	13
Marked for review	0
Total	Total
All Domain	All Domain
Total Question	13
Correct	0
Incorrect	0
Unattempted	13
Marked for review	0

Review the Answers

Sorting by

All

Question 1

Unattempted

Domain : Other

Given :

```
1.      class Bird extends Animal implements Fly{
2.            // lots of code goes here
3.      }
4.
5.       interface Fly{
6.            // lots of code goes here
7.      }
8.
9.      class Animal{
10.          // lots of code goes here
11.    }
12.
13.   class Whiz{
14.          static Animal a = new Bird();
15.          public static void main(String []args){
16.                  System.out.print(a instanceof Fly);
17.          }
18.   }
```

What of the following is true?

A. Code compiles and prints true ✓

B. Code compiles and prints false

C. “a” has a Bird

D. “a” has a Animal

E. Compilation fails

Explanation:

Explanation:
Option A is the correct answer.

Option A is correct as the code compiles fine and produces the output as true. Since the class “Bird” has implemented the interface “Fly” Bird object can be considerd as a “Fly”, simply a “Bird” class object passes the “is-a” test with “Fly”. Therefore output is true.

Option B is incorrect as explained above.

Option D is incorrect since the class “Bird” is a subclass of the “Animal” class so we can consider “a” as an Animal.

Option C is incorrect there is no “Bird” object in the “Bird” class.

Option E is incorrect as the code compiles fine.

Reference : http://docs.oracle.com/javase/tutorial/java/IandI/subclasses.html

Ask our Experts

Rate this Question? ☺ ☹

Question 2 Unattempted

Domain : Other

Given :

```
1.  public class Whiz {
2.          public static void main(String[] args) {
3.                  I i = new I() { };
4.                  System.out.println(I.x + i.getValue() +""+ i);
5.          }
6.  }
7.
8.  interface I {
9.          int x = 10;
10.
11.         public default int getValue() {
12.                 return 5;
13.         }
14.
15.         public default String toString() {
16.                 return "I";
17.         }
18. }
```

What is the output?

A. 10I

B. 15I

C. Compilation fails due to error at line 11.

D. Compilation fails due to error at line 15. ✓

E. Compilation fails due to multiple errors

Explanation:

Explanation:
Option D is the correct answer

From Java SE 8, we are allowed to have non-abstract default and static methods but there are some limitations. Such limitation is that we cannot override methods of Objects class. So the code fails to compile due to line 15 since toString is Object class method. Hence option D is correct.

Reference : http://docs.oracle.com/javase/tutorial/java/IandI/abstract.html

Ask our Experts

Rate this Question? ☺ ☹

Question 3

Unattempted

Domain : Other

Which of the following is a full signature of a valid default method?

A. public default String toString()

B. default static void print()

C. abstract default void help()

D. public default void method() ✓

E. None of the above.

Explanation:

Explanation :

Option D is the correct answer.

From java 8 we can define non abstract default method in interfaces. Option A is incorrect since it is illegal to override Objects class methods using default methods.

Option B is incorrect as the default methods can’t be static. Option C is incorrect as they should be non abstract.

Option D is correct since it shows correct signature for the default method.

Reference : http://docs.oracle.com/javase/tutorial/java/IandI/defaultmethods.html

Ask our Experts

Rate this Question? ☺ ☹

Question 4

Unattempted

Domain : Other

Given :

```
1.      public class Whizlab{
2.          public static void main(String[] args) {
3.                  System.out.print(In2.print());
4.          }
5.      }
6.
7.      interface In1{
8.          public static void main print(){
9.                  System.out.println("In1");
10.       }
11.   }
12.
13.   interface In2 extends In1{
14.       static String print(){
15.               return "In2";
16.       }
17.   }
```

What is the output?

A. In1

B. In2 ✓

C. An Exception.

D. Compilation fails due to error at line 14.

E. Compilation fails due to multiple errors.

Explanation:

Explanation :
Option B is the correct answer.

Java 8 includes support for static methods within interfaces. These methods are defined explicitly with the static keyword and function nearly identically to static methods defined in classes. In fact, there is really only one distinction between a static method in a class and an interface. A static method defined in an interface is not inherited in any classes that implement the interface and even not inherited to it's sub interfaces. So there won't be any compile time error at line 14. Hence option D is incorrect.

You can access interface static method using it's interface name only. So, At Line 3 statement prints "In2". Hence option B is correct

Reference : http://docs.oracle.com/javase/tutorial/java/IandI/defaultmethods.html

Ask our Experts

Rate this Question? ☺ ☹

Question 5 Unattempted

Domain : Other

Given :

```
1.  class Sup {
2.          protected void method() {
3.                  System.out.print("Sup");
4.          }
5.  }
6.  class Sub extends Sup {
7.          //override method () here
8.  }
```

Which of the following methods, inserted independently in the class Sub, correctly override method() from the class Sup?

A. public final void method(){ } ✓

B. private void method(){ }

C. void method(){ }

D. public void method(int i){ }

E. protected int method(){}

Explanation:

Explanation :
Option A is the correct answer.

Option A is correct as there we have used less restrictive access modifier and it is legal to make overriding method final therefore that method correctly override the “method()”.

Option E is incorrect since we can’t change void or primitive return types when overriding methods.

Option B and C are incorrect as we use more restrictive access level there (private and default access levels).

Option D is incorrect as it is overloading not overriding since we have changed the argument list.

Reference : http://docs.oracle.com/javase/tutorial/java/IandI/override.html

Ask our Experts

Rate this Question? ☺ ☹

Question 6

Unattempted

Domain : Other

Given :

```
1.  class A extends B implements C {
2.          // lots of code goes here
3.  }
4.
5.  interface C {
```

```
6.          // lots of code goes here
7.    }
8.
9.    abstract class B{
10.         // lots of code goes here
11.   }
12.
13.   class Whiz {
14.         public static void main(String []args) {
15.               A a = new A();
16.         }
17.   }
```

Which of the following statement will produce the output as true when inserted at line 16? *(Choose 2)*

A. System.out.print(a.instanceof(A));

B. System.out.print(a.instanceof(B));

C. System.out.print(a instance of A);

D. System.out.print(a instanceof C); ✓

E. System.out.print(a instanceof Object); ✓

Explanation:

Explanation:
Options D and E are the correct answer.

In this code class "A" has implemented the interface C and extended the class B. So class 'A' object can be considered as a "C" or "B". So if we test the object "a" for is-a relationship with "C" or "B" then the output will be true. And instanceof is an operator. Therefore option D is correct.

As any class in java is an indirect or direct subclass of Object, So option E is correct.

Options A, B and C are incorrect as “instanceof” has been used incorrectly there.

Reference : http://docs.oracle.com/javase/tutorial/java/IandI/subclasses.html

Ask our Experts

Rate this Question? ☺ ☹

Question 7 Unattempted

Domain : Other

Given following method :

void print(){ System.out.println(“default”); }

Which are correct overriding of print() method?

A. public final void print(){ } ✓

B. private final void print(){ }

C. int print(){ return 0; }

D. public void print(int i){ }

E. protected void print() throws Exception { }

Explanation:

Explanation:

Option A is the correct answer

Option A is correct as there we have used less restrictive access modifier and it is legal therefore that method correctly overrides the “print()”.

Option B is incorrect as we use more restrictive access level there (private access level).

Option C is incorrect since we can’t change void or primitive return types when overriding methods.

Option D is incorrect as it is overloading not overriding since we have changed the argument list.

Option E is incorrect as it is declared to throw a checked exception which is not thrown in the super class method. It is illegal to throw a new checked exception in overriding method.

Reference : http://docs.oracle.com/javase/tutorial/java/IandI/override.html

Ask our Experts

Rate this Question? ☺ ☹

Question 8

Unattempted

Domain : Other

Given :

```
class Ab {
    private final void print() {
        System.out.print("class Ab");
    }
}

class Sub extends Ab {
    private void print() {
        System.out.print("class Sub");
    }
}

class Whiz {
    public static void main(String [] args) {
        Ab ab = new Sub();
        ab.print();
    }
}
```

Which is true?

A. The output will be class Ab

B. The output will be class Sub

C. Compilation fails due to error on line 8 since final methods can’t be overridden.

D. Compilation fails due to error on line 16. ✓

E. An exception will be thrown at runtime

Explanation:

Explanation:

Option D is the correct answer.

Option D is correct as code fails due to error on line 16. Trying to access private method from the outside class causes a compile time error.

Options A, B and E are incorrect as the code fails to compile.

Option C is incorrect. The “print()” method can’t be seen from any other class since it is declared as private. So it is impossible to overide them because the method is not inherited. So at line 8 there is new “print()” method not overridden method. Therefore there is no error due to final at line 8.

Reference :http://docs.oracle.com/javase/tutorial/java/IandI/polymorphism.html

Ask our Experts

Rate this Question? ☺ ☹

Question 9 Unattempted

Domain : Other

Given :

```
1.  class Bird extends Animal {
2.          public void print(){ System.out.print("Bird"); }
3.          void fly(){ System.out.print("Bird flies");   }
4.  }
5.
```

```
6.   class Animal {
7.       public void print(){ System.out.print("Animal"); }
8.   }
9.
10.  class Whiz {
11.
12.      public static void main(String []args) {
13.          Animal b = new Bird();
14.          b.print();
15.          b.fly();
16.      }
17.   }
```

What is the output?

A. BirdBird flies

B. Bird

C. Bird followed by an exception

D. Compilation fails ✓

Explanation:

Explanation:
Option D is the correct answer.

Option D is correct as the code fails to compile. At line 15, we have tried to invoke the “fly()” method of the “Bird” class but it causes a compile time error because the reference type of the object is “Animal” and there is no such a method declared inside the Animal class.

Other options are incorrect as explained above.

Reference : http://docs.oracle.com/javase/tutorial/java/javaOO/objectcreation.html

http://docs.oracle.com/javase/tutorial/java/javaOO/usingobject.html

Ask our Experts

Rate this Question? ☺ ☹

Question 10

Unattempted

Domain : Other

Given

```
 interface I{
 void meth();
 }

 class A implements I{
     void A(String s){
         }
     public void meth(){
         System.out.print("A");
    }
}

class C extends A implements I{
    public void meth(){
        System.out.print("C");
    }
}

class Ex6{
    public static void main(String args[]){
        A a = new A();
22.                         C c1 = (C)a;
23.                         c1.meth();
24.              }
25.    }
```

What is the output?

A. A

B. C

C. Compilation fails due to error on line 6

D. Compilation fails due to multiple errors

E. An exception will be thrown at runtime ✓

Explanation:

Explanation:

Option E is the correct answer.

Option E is correct as here we try to cast a super class reference to lower class reference, but super class reference refers to super class object, so casting will cause a ClassCastException.

Option A and B are incorrect as the code throws an exception before producing any output.

Options C and D are incorrect as the code compiles fine.

Reference :http://docs.oracle.com/javase/tutorial/java/IandI/subclasses.html

Ask our Experts

Rate this Question? ☺ ☹

Question 11 Unattempted

Domain : Other

Given :

```
1.  class Person {
2.           Person(String s){ super(); }
3.  }
4.  class Employee extends Person {
5.           // insert code here
6.           // insert code here
7.  }
8.
9.  public class Whiz {
10.          public static void main(String [] args) {
11.                  Person  ab = new Employee();
12.          }
13. }
```

Which, inserted independently at line 5, will produce the output as “Kent”?

A. Employee(String s){ super();System.out.print(s); }   Employee(){ this("Kent");}

B. Employee(String s){ super(null);System.out.print(s); }   Employee(){ this("Kent"); } ✓

C. Employee(String s){ super(“ ”);System.out.print(s); }   Employee(){ super("Kent"); }

D. Employee(String s){ super(null);System.out.print(“Kent”); }

E. None of the above.

Explanation:

Explanation:
Option B is the correct answer.

Option B is correct as we have used the keyword “this” to invoke the “Employee(String s)” constructor. And print statement in that constructor produces the expected output.

Option A is incorrect as the super class has only one constructor which can take String as the argument so calling “super()” with empty parameters causes a compile time error.

Option C is incorrect as the expected output can only be achieved by invoking the “Employee(String s)” constructor so using super will skip the needed constructor and will invoke the super class constructor. Therefore the expected output will not be produced.

Option D will cause compile time error as we need to have “Employee()” constructor which doesn't take any parameter as we used “new Employee()” at line 11.

Reference : http://docs.oracle.com/javase/tutorial/java/javaOO/thiskey.html

Ask our Experts

Rate this Question? ☺ ☹

Question 12 Unattempted

Domain : Other

Which of the following can be inserted in to an interface?

A. **public abstract static void print();**

B. **default abstract void print();**

C. **public void print(){}**

D. **public default void print(){}** ✓

E. **None of above**

Explanation:

Option D is the correct answer

From java SE 8, we can have static and default non abstract method in interfaces. So Option D is correct.

Options A and B are incorrect since static and default methods should be non-abstract in interfaces.

Option C is incorrect since all non-abstract methods should be either static or default in interfaces.

Reference :http://docs.oracle.com/javase/tutorial/java/IandI/usinginterface.html

Ask our Experts

Rate this Question? ☺ ☹

Question 13 Unattempted

Domain : Other

Which of following represents a correct abstract class?

A. abstract class Whiz{ public void print(); }

B. abstract class Whiz{ public void print() { } } ✓

C. class Whiz{ public void print(); }

D. final abstract class Whiz{ public abstract void print(); }

E. None of above

Explanation:

Explanation:

Option B is the correct answer

Option B is correct. Abstract methods are optional for an abstract class, it is enough to mark a class with the abstract keyword to make it abstract.

Option A is incorrect as the abstract methods should be marked using the abstract keyword.

Option C is incorrect as the abstract class should be marked with the abstract keyword and also the abstract method should be marked with the abstract keyword.

Option D is incorrect as the abstract class can't be final.

REFERENCE :http://docs.oracle.com/javase/tutorial/java/IandI/abstract.html

Ask our Experts

Rate this Question? ☺ ☹

Finish Review

Certification

Cloud Certification

Java Certification

PM Certification

Big Data Certification

Company

Support

Discussions

Blog

Follow us