Started on	Friday, 27 March 2020, 1:34 AM
State	Finished
Completed on	Friday, 27 March 2020, 1:35 AM
Time taken	1 min 27 secs
Marks	7.00/7.00
Grade	**100.00** out of 100.00
Feedback	Congratulations!! You have secured more than 80%

Question 1

Correct

Mark 1.00 out of 1.00

Predict the output.

```
abstract class Sample {
    public int x;
    Sample()     {
        x = 10;
    }
    abstract final public void display();
}

class Test extends Sample {
    final public void display()   {
        System.out.println("x = " + x);
    }
    public static void main(String[] args)   {
        Test t = new Test();
        t.display();
    }
}
```

Select one:

○ a. x = 10

○ b. x = 0

○ c. Runtime error

● d.
Compile Time error ✓

Your answer is correct.

The correct answer is:
Compile Time error

Correct

Mark 1.00 out of 1.00

Statement 1 : An abstract class cannot have non abstract methods

Statement 2 : An abstract class should have a minimum of one abstract method in its class.

Select one:

○ a. Statement 2 alone is true

○ b. Statement 1 alone is true

○ c. Both Statement 1 and Statement 2 are true

● d. Statement 1 and Statement 2 are false ✓

Your answer is correct.

The correct answer is: Statement 1 and Statement 2 are false

Question 3

Correct

Mark 1.00 out of 1.00

Observe the below code.

```
interface A1 {
       public abstract void  a11();
       public abstract void  a12();
}
interface A2 extends A1 {
        void  a21();
        void  a22();
}
public class C  implements A2  {
}
```

In the above scenario, which methods should class C override?

Select one:

○ a. void a21() { }
void a22(){ }

○ b. public void a11() { }
public void a12() { }
void a21() { }
void a22(){ }

○ c. public void a21() { }
public void a22(){ }

● d. public void a11() { }
public void a12() { }
public void a21() { }
public void a22(){ } ✓

Your answer is correct.
In an interface when a method is just declared, by default they are public and abstract. So when the methods in an interface are overridden they need to be public (access level cannot be more restrictive). If not it leads to compilation error.

The correct answer is: public void a11() { }
public void a12() { }
public void a21() { }
public void a22(){ }

Correct

Mark 1.00 out of 1.00

keyword. State true or false.

Select one:

○ True

● False ✓

The correct answer is 'False'.

Question 5

Correct

Mark 1.00 out of 1.00

Predict the output of the following program

```
class Parent  {
   public void display()    {
      System.out.println("In Parent");
   }
}
class Test extends Parent  {
   public void display()  {
      System.out.println("In child");
   }
   public void testMethod()  {
      System.out.println("In test method");
   }
   public static void main(String[] args)  {
      Parent p = new Test();
      p.display();
      p.testMethod();
   }
}
```

Select one:

● a. Compile Time error ✓

○ b. In Parent
In child
In test method

○ c. Runtime error

○ d. In child
In test method

Your answer is correct.

When a parent class reference holds a child class object, using that reference we can invoke the methods in the parent and also the overridden methods in child. To invoke child specific method (testMethod) downcasting needs to be done. If it is accessed directly using parent class reference, it leads to compilation error.

The correct answer is: Compile Time error

Correct

Mark 1.00 out of 1.00

Select one:

○ a. Method Overloading

● b. Method Overriding ✓

○ c. Both the options

Your answer is correct.
When parent class reference holds a child class object and we invoke the overridden method, which method is invoked, depends on the object it holds and not on the reference. Object is created at run time. Hence it is an example for run time polymorphism.

The correct answer is: Method Overriding

Question **7**

Correct

Mark 1.00 out of 1.00

If the access specifier of the display method in super class is protected, then what could be the valid specifier for the overriding display method in sub class?

Select one:

● a.
protected and public ✓

○ b.
protected, default and public

○ c. protected and default

○ d.
private and protected

Your answer is correct.
The subclass overridden method cannot have weaker access than super class method. Which means, when overriding a method, the access level cannot be more restrictive than the overridden methods access level. In parent class the method is protected, then in child class it can be protected or public.

The correct answer is:
protected and public