Started on	Saturday, 4 April 2020, 12:25 AM
State	Finished
Completed on	Saturday, 4 April 2020, 12:28 AM
Time taken	3 mins 17 secs
Marks	8.00/9.00
Grade	**88.89** out of 100.00
Feedback	Congratulations!! You have passed by securing more than 80%

Question 1

Correct

Mark 1.00 out of 1.00

Predict the output

```
class Car implements Insurance
{
   public int calcPremium(int i)
   {
      return i = i * i;
   }
}
interface Insurance
{
   int calcPremium(int i);
}
public class MainClass
{
   public static void main(String[] args)
   {
      Insurance b = new Car();
      System.out.println(b.calcPremium(2));
   }
}
```

Select one:

- a. Compile time error because you cannot create an object of type interface Insurance
- b. Run time Error
- c. The output will be 4 ✔
- d. Compile time error because you must create interface before implementing it.

b.calcPremium(2) will look for the method within the interface - b being the reference of interface type.

The definition of this method is given in the Car class that has implemented this interface. With i=i*i, the output is 4.

The correct answer is: The output will be 4

Correct

Mark 1.00 out of 1.00

```
interface DoStuff2
{
        float getRange(int low, int high);
}
interface DoMore
{
        float getAvg(int a, int b, int c);
}
abstract class DoAbstract implements DoStuff2, DoMore
{ }
class DoStuff implements DoStuff2
{
        public float getRange(int x, int y)
        {
                return 3.14f;
        }
}
interface DoAll extends DoMore
{
        float getAvg(int a, int b, int c, int d);
}
```

Select one:

- a. Compile time Error
- b. The file will compile without error. ✔
- c. Runtime Error

Your answer is correct.

The correct answer is: The file will compile without error.

Question 3

Correct

Mark 1.00 out of 1.00

The type Vehicle has drive functionality. The classes Car and Bike implements the drive functionality and can be further subclassed. Fill in the given code with appropriate access specifier so that the subclasses of Car and Bike do not modify the Drive functionality.

```
interface Vehicle{
  void drive();
}
class Car implements Vehicle{
```

public ✔ final ✔ void drive() {

```
      //drive   }
}
```

Your answer is correct.

The correct answer is:
The type Vehicle has drive functionality. The classes Car and Bike implements the drive functionality and can be further subclassed. Fill in the given code with appropriate access specifier so that the subclasses of Car and Bike do not modify the Drive functionality.

```
interface Vehicle{
  void drive();
}
class Car implements Vehicle{
   [public]  [final] void drive()   {
      //drive   }
}
```

Correct

Mark 1.00 out of 1.00

```
abstract class Vehicle
{
   abstract void calcPremium(Number N);
}
interface Insurance
{
   abstract void calcPremium (Object O);
}
class Car extends Vehicle implements Insurance
{
            public void calcPremium (Object O)
              {
                 System.out.println("Object");
              }
   void calcPremium (Number N)
   {
      System.out.println("Number");
   }
 }
public class Test
{
   public static void main(String[] args)
   {
      Vehicle a = new Car();
       a. calcPremium (new Integer(121));
        Insurance b = new Car();
         b. calcPremium (new Integer(121));
         Car c = new Car();
          c. calcPremium (new Integer(121));
   }
}
```

Select one:

- a. Number
  Object
  Number ✔
- b. Number
  Number
  Object
- c. Run time error
- d. Compile time error

a. calcPremium () with an integer object invokes this method within the Car class that takes Number argument. This is because a is the Vehicle type reference and Vehicle class has calcPremium () with a Number argument declared abstract. Hence, Number.

b. calcPremium () with an integer object invokes this method within the Car class that takes Object argument. This is because b is the Insurance type reference and Insurance interface has calcPremium () with an Object argument declared abstract. Hence, Object.

c. calcPremium () with an integer object invokes this method within the Car class that takes Number argument. Hence, Number.

The correct answer is: Number
Object
Number

Correct

Mark 1.00 out of 1.00

```
interface Employee
{
    int a=90;
}
class PermanentEmployee implements Employee
{
    public void f1()
    {
        a=10;
    }
}
```

Select one:

- a. error, since interfaces Employee is not public
- b. error, since variable a is default
- c. error, since variable a is assigned a value ✔
- d. no error

Variables within interface are static and final by default. They can not be assigned a value in the classes that implement this interface.

The correct answer is: error, since variable a is assigned a value

Question 6

Incorrect

Mark 0.00 out of 1.00

If a class implements two interfaces and they both have a default method with same name and signature but different implementation, then a conflict will arise because the compiler will not able to link a method call due to ambiguity. State true or false.

Select one:

- True
- False ✖

The correct answer is 'True'.

Question 7

Correct

Mark 1.00 out of 1.00

11. public interface Status {

12. public static final ✔ double PI = 3.14;

13. }

Fill the correct choice.

Your answer is correct.

The correct answer is:
11. public interface Status {

12. [public static final] double PI = 3.14;

13. }

Fill the correct choice.

Correct

Mark 1.00 out of 1.00

public void insuranceDescription(String s);
}
Which is the correct class?

Select one:

a. public class Car implements Insurance
{
public void insuranceDescription (Integer i) { }
}

b. public abstract class Car implements Insurance
{
public abstract void insuranceDescription (String s) { }
}

c. public class Car extends Insurance
{
public void insuranceDescription (Integer i) { }
}

d. public abstract class Car implements Insurance { } ✔

Your answer is correct.

The correct answer is: public abstract class Car implements Insurance { }

Question 9

Correct

Mark 1.00 out of 1.00

An interface can contain public, static, final fields (i.e., constants) default and static methods with bodies True ✔

An instance of interface can be created. False ✔

A class can implement multiple interfaces. True ✔

Many classes can implement the same interface. True ✔

Your answer is correct.

The correct answer is:

An interface can contain public, static, final fields (i.e., constants) default and static methods with bodies [True]

An instance of interface can be created.[False]

A class can implement multiple interfaces. [True]

Many classes can implement the same interface.[True]