Question: 1

Given:

```
1  class Base {
2     // insert code here
3  }
4
5  public class Derived extends Base {
6
7  public static void main ( String [ ] args ) {
8        Derived obj = new Derived ( ) ;
9        obj . setNum ( 3 ) ;
10       System . out . println ( "Square = " + obj . getNum ( ) * obj . getNum ( ) ) ;
11    }
12 }
```

Whi c h two options, when inserted independently inside class Base, ensure that the c lass is being properly encapsulated and allow the program to execute and print the square of the number?

A.

Java

```
2  private int num ;
3
4  public int getNum ( ) {
5     return num ;
6  }
7
8  public void setNum ( int num ) {
9     this . num = num ;
10 }
```

B.

Java

```
2  public int num ;
3
4  protected public int getNum ( ) {
5     return num ;
6  }
7
8  protected public void setNum ( int num ) {
9     this . num = num ;
10 }
```

C.

Java

```
2  private int num ;
3
4  public int getNum ( ) {
5     return num ;
6  }
7
8  private void setNum ( int num ) {
9     this . num = num ;
```

```
10 }
```

D.

Java

```
2  protected  int  num ;
3
4  public  int  getNum ( )  {
5     return  num ;
6  }
7
8  public  void  setNum ( int  num )  {
9     this . num  =  num ;
10 }
```

E.

Java

```
2  protected  int  num ;
3
4  private  int  getNum ( )  {
5     return  num ;
6  }
7
8  public  void  setNum ( int  num )  {
9     this . num  =  num ;
10 }
```

answer

Option A, standard packaging practice.

Option B, prote c ted and publi c cannot be used at the same time, causing compilation errors.

For option C , you should change the visibility modifier of the setNum method to publi c .

Option D, a standard encapsulation method that supports field inheritance under different suites.

For option E, the visibility modifier of the getNum method should be changed to private.

Question: 2

Given:

Java

```
1  public class Whizlabs {
2
3     public static void main(String[] args) {
4        String s = "A";
5
6        switch (s) {
7           case "a":
8              System.out.println("simaple A ");
9           default:
10             System.out.print("default ");
11          case "A":
12             System.out.print("Capital A ");
```

```
13       }
14    }
15 }
```

What is the result?

A. simaple A **B.** Capital A C. simaple A default Capital A
D. simaple A default E. Compilation fails.

answer

The content to be selected by the switch is the "A" string to which the s variable refers, so the case on line 11 is executed, and "Capital A" is output and the switch is left.

Question: 3

```
public class Test {
  static String[][] arr = new String[3][];
  private static void doPrint() {
       //insert code here
    }
  public static void main(String[] args) {
       String[] class1 = {"A", "B", "C"};
       String[] class2 = {"L", "M", "N", "O"};
       String[] class3 = {"I", "J"};
       arr[0] = class1;
       arr[1] = class2;
       arr[2] = class3;
       Test.doPrint();
    }
}
```

Which code fragment, when inserted at line //insert code here, enables the code to print COJ?

A.

```
6   int i = 0;
7   for (String[] sub : arr) {
8       int j = sub.length - 1;
9       for (String str : sub) {
10          System.out.println(str[j]);
11          i++;
12      }
13 }
```

B.

```
6 for (int i = 0; i < arr.length; i++) {
7     int j = arr[i].length - 1;
8     System.out.print(arr[i][j]);
9 }
```

C.

```
6   int i = 0;
7   for (String[] sub : arr[][]) {
8       int j = sub.length;
9       System.out.print(arr[i][j]);
```

```
10    i++;
11 }
```

D.

```
6  for (int i = 0; i < arr.length - 1; i++) {
7      int j = arr[i].length - 1;
8      System.out.print(arr[i][j]);
9      i++;
10 }
```

answer

Option A, line 10 will compile errors because the type of the str variable is not an array.
Option B, which outputs the last element of each one-dimensional array element of the two-dimensional array, is exactly "COJ".
Option C, line 7 will compile error, wrong foreach usage.
Option D will only output "C".

Question: 4

Given:

TestApp.java

```
1  public class TestApp {
2      public static void main(String[] args) {
3          TestApp t = new TestApp();
4          try {
5              t.doPrint();
6              t.doList();
7          } catch (Exception e2) {
8              System.out.println("Caught " + e2);
9          }
10     }
11     public void doList() throws Exception {
12         throw new Error("Error");
13     }
14     public void doPrint() throws Exception {
15         throw new RuntimeException("Exception");
16     }
17 }
```

What is the result?

A.

```
Caught java.lang.RuntimeException: Exception
Exception in thread "main" java.lang.Error: Error
at TestApp.doList(TestApp.java: 14)
at TestApp.main(TestApp.java: 6)
```

B.

```
Exception in thread "main" java.lang.Error: Error
at TestApp.doList(TestApp.java: 14)
at TestApp.main(TestApp.java: 6)
```

C.

```
Caught java.lang.RuntimeException: Exception
Caught java.lang.Error: Error
```

D.

```
Caught java.lang.RuntimeException: Exception
```

answer

Line 5 of the program calls the doPrint method on line 14, and then throws a RuntimeException object. This RuntimeException will be caught in the catch on line 7 of the program and then output "Caught java.lang.RuntimeException: Exception".

Question: 5

Given:

```
1  class Cake {
2
3      int model;
4      String flavor;
5
6      Cake() {
7          model = 0;
8          flavor = "Unknown";
9      }
10 }
11
12 public class Test {
13
14     public static void main(String[] args) {
15         Cake c = new Cake();
16         bake1(c);
17         System.out.println(c.model + " " + c.flavor);
18         bake2(c);
19         System.out.println(c.model + " " + c.flavor);
20     }
21
22     public static Cake bake1(Cake c) {
23         c.flavor = "Strawberry";
24         c.model = 1200;
25         return c;
26     }
27
28     public static void bake2(Cake c) {
29         c.flavor = "Chocolate";
30         c.model = 1230;
31         return;
32     }
33 }
```

What is the result?

A.

```
0 unknown
0 unknown
```

B.

```
1200 Strawberry
1200 Strawberry
```

C.

1200 Strawberry
1230 Chocolate
D. Compilation fails

answer
The Cake category and its object fields are not decorated with modifiers, so the default access range is under the same suite, and of course the Test category on line 12.
The bake1 method on line 22 of the program sets the object field to c.flavor = "Strawberry", c.model = 1200.
The bake2 method on line 28 of the program sets the object field to c.flavor = "Chocolate", c.model = 1230.

Question: 6

Given the code fragment:
Java

```
6 float x = 22.00f % 3.00f;
7 int y = 22 % 3;
8 System.out.print(x + ", " + y);
```

What is the result?
A. 1.0, 1
B. 1.0f, 1
C. 7.33, 7
D. Compilation fails
E. An exception is thrown at runtime

answer
"%" is a remainder operator that supports remainder calculations for integers and single precision floating point (float) types. If one of the operands is a single precision floating point number type, the calculated result will also be a single precision floating point number type.

The result of dividing 22 by 7 is 7 and 1, so the sixth line will calculate the "1" of the single precision floating point type, and the seventh line will calculate the "1" of the integer type.

Question: 7

Given:

```
1 public class MyClass {
2
3    public static void main(String[] args) {
4        String s = " Java Duke ";
5        int len = s.trim().length();
6        System.out.print(len);
7    }
8 }
```

What is the result?
A. 8
B. 9
C. 11
D. 10
E. Compilation fails

answer
The trim method of the String object can eliminate all spaces at the beginning and end, and return the new string without changing the contents of the original string object. So in this question, the fifth line will remove the "Java Duke" string and turn it into a "Java Duke", so the string length will become "9".

Question: 8

The protected modifier on a Field declaration within a public class means that the field _____.
A. Cannot be modified
B. Can be read but not written from outside the class
C. Can be read and written from this class and its subclasses only within the same package
D. Can be read and written from this class and its subclasses defined in any package

answer
The protected modifier allows members to "freely access the same suite, and different suites need inheritance to access."

Question: 9

Given
```
public double getArea() {
      return /* ??? */;
   }
}
class App {
   public static void main(String[] args) {
      Circle c1 = new Circle(17.4);
      c1.area = Math.PI * c1.getRadius() * c1.getRadius();
   }
}
```

The class is poorly encapsulated. You need to change the circle class to compute and return the area instead.
Which two modifications are necessary to ensure that the class is being properly encapsulated?
A. Remove the area field.
B. Change the getArea( ) method as follows:
Java
```
1 public double getArea ( ) { return Match.PI * radius * radius; }
```
C. Add the following method:
Java
```
1 public double getArea ( ) {area = Match.PI * radius * radius; }
```
D. Change the cacess modifier of the SerRadius ( ) method to be protected.

answer

In order to make the result of calculating the area of the Circle class undisturbed, we can simply use the getArea method to pass back the result value. So first remove the area field, then implement the area algorithm (πr2) in the getArea method. So the answer is option A and option B.
Option C, will cause a compilation error, because the method has a defined return type, but no return value.
Option D, change setRadius to use protected, only make it impossible to access under different suites.

Question: 10

Given:
Java

```
1  public class Series {
2
3      public static void main() {
4          int arr[] = {1, 2, 3};
5          for (int var : arr) {
6              int i = 1;
7              while (i <= var);
8              System.out.println(i++);
9          }
10     }
11 }
```

What is the result?

A.	B.	C.
1	1	2
1	2	3
1	3	4

D. Compilation fails
E. The loop executes infinite times
answer
It should be noted here that the while loop of the 7th line of the program is finally marked with a semicolon, so the range of the while loop is only on the 7th line. "i <= var" will always be established and become an infinite loop.

Question: 11

Given the code fragment:

```
5   // insert code here
6   arr[0] = new int[3];
7   arr[0][0] = 1;
8   arr[0][1] = 2;
9   arr[0][2] = 3;
10
11  arr[1] = new int[4];
12  arr[1][0] = 10;
13  arr[1][1] = 20;
14  arr[1][2] = 30;
```

```
15  arr[1][3] = 40;
```

Which two statements, when inserted independently at line // insert code here, enable the code to compile?

A. int[][] arr = null;
B. int[][] arr = new int[2];
C . int[][] arr = new int[2][];
D. int[][] arr = new int[][4];
E. int[][] arr = new int[2][4];
F. int[][] arr = new int[0][4];

answer

From lines 6 through 15, you can conclude that the array size referenced by this arr variable is at least 2 x 4.

Option A, without the materialized two-dimensional array object, will throw NullPointerExc eption when accessing the array elements later.

Option B, which produces a one-dimensional array of objects, will be compiled with a two-dimensional array of styles.

Option C , the correct two-dimensional array materialization method, the size also meets the demand

Option D, the wrong 2D array materialization method.

Option E, the correct two-dimensional array materialization method, the size also meets the requirements.

Option F, the correct 2D array is materialized, but the size does not meet the requirements.

Question: 12

Given:

```
1   public class ComputeSum {
2
3       public int x;
4       public int y;
5       public int sum;
6
7       public ComputeSum(int nx, int ny) {
8           x = nx;
9           y = ny;
10          updateSum();
11      }
12
13      public void setX(int nx) {
14          x = nx;
15          updateSum();
16      }
17
18      public void setY(int ny) {
19          x = ny;
20          updateSum();
21      }
22
23      void updateSum() {
```

```
24          sum = x + y;
25      }
26  }
```

This c lass needs to prote c t an invariant on the sum field.
Whi c h three members must have the pr iv ate a cc ess modifier to ensure that this invariant is maintained?
A. The x field
B. The y field
C . The sum field
D. The C omputerSum ( ) c onstru c tor
E. The setX ( ) method
F. The setY ( ) method
answer
Only in the options C , D, E, F will change the value of the sum field, but the option D is a constructor , will be executed when the object is materialized, the materialization can no longer be called, and then There is also no chance to change the value of the sum field.
So after removing option D, the answer is option C , E, F

Question: 13

Given:

```
1   public class Whizlabs {
2
3       public static void main(String[] args) {
4           try {
5               Double number = Double.valueOf("120D");
6           } catch (NumberFormatException ex) {
7           }
8           System.out.println(number);
9       }
10  }
```

What is the result?
A. 120
B. 120D
C. A NumberFor Mate X C eption Will BE thrown.
D. C ompilation fails Total Line Due to 5. The AT error
E. C ompilation Tails AT Line Due to error. 8.
answer
On line 5 of the program, use the valueOf method of the Double class to convert the string into a Double object.
"120D" or "120d" is the correct numerical representation in Java, which is expressed as 120 in the double type. If it is "120F" or "120f", it means 120 of the float type. If it is "120" or "120", it means 120 of the int type. The code examples are as follows:

```
1  int a = 120;
2  float b = 120f;
3  double c = 120d;
```

So the fifth line of the program will execute successfully and generate a Double object without throwing an exception. However, in the eighth line of the program, the number variable is used, but the number variable is defined only in the try block starting at line 4, and the valid range is only in the try block. Therefore, the 8th line of the program will compile errors because no number is defined.

Question: 14

Whi c h statement is true about the default c onstru c tor of a top-level c lass?
A. It c an take arguments.
B. It has pr iv ate a cc ess modifier in its de c laration.
C. It c an be overloaded.
D. The default c onstru c tor of a sub c lass always invokes the no - argument c onstru c tor of its su pe r c lass.
answer
Option A, the default constructor cannot pass arguments.
Option B, the default constructor is decorated with public.
Option C, the constructor cannot be overwritten.
Option D, the default construction sub-preset will call the upper-level constructor without parameters. For example, the default constructor of the A category, the automatically generated code is as follows:

```
1  public A(){
2      super();
3  }
```

Question: 15

Given the code fragment:
Java

```
1   public class Test {
2
3       public static List data = new ArrayList();
4
5       // insert code here
6       {
7           for (String x : strs) {
8               data.add(x);
9           }
10          return data;
11      }
12
13      public static void main(String[] args) {
14          String[] d = {"a", "b", "c"};
15          update(d);
16          for (String s : d) {
17              System.out.print(s + " ");
18          }
19      }
20  }
```

Which code fragment, when inserted at // insert code here, enables the code to compile and and print a b c?
A. List update (String[] strs)
B. static ArrayList update(String[] strs)
C. static List update (String[] strs)
D. static void update (String[] strs)
E. ArrayList static update(String[] strs)
answer
The program originally provided by the title, the strs on line 7 will compile errors because there is no announcement. The return on line 10 will compile errors because the current block is not a way to return the List object. Line 15 will compile errors because the update method is not found.
In order to solve the problem of compilation errors, the declaration of the method must be completed on line 5. The name of the method is called "update", because it can be used directly in the main method, so it must be a static (stati c) method, which should be modified with stati c . The update method can only accept the arguments of the string array arguments, so you need to set the parameters of a string array. As for the parameter name, you can know from the 7th line, which is the name of the variable "strs" . The 10th line needs to return the List object, so the return value type of the update method should be declared as the parent or category of the List or List.
Options A, B, D, and E do not meet the requirements, only option C is correct.

Question: 16

What is the name of the Java c on c ept that uses a cc ess modifiers to prote c t variables and h ide them within a c lass?
A. Encapsulation
B. Inheritance
C. Abstra c tion
D. Instantiation
E. Polymorphism
answer
The concept of encapsulation is to protect the program from being modified and hidden by the program (code).

Question: 17

Given:

```
1  public class Palindrome {
2
3      public static int main(String[] args) {
4          System.out.print(args[1]);
5          return 0;
6      }
7  }
```

And the c ommands:

```
Javac Palindrome. java
java Palindrome Wow Mom
```

What is the result?

A. C ompilation fails
B. The c ode c ompiles, but does not exe c ute.
C. Paildrome
D. Wow
E. Mom

answer

The main method declaration method of the Java program entry point is "publi c stati c void main(String[] args)" or "publi c stati c void main(String... args)". The correct program entry point is not declared here, and the return type is not void. Therefore , when executing with the " java " instruction, only the return type must be prompted to use void.

Question: 18

Given:

```
1   public class Access {
2
3       private int x = 0;
4       private int y = 0;
5
6       public static void main(String[] args) {
7           Access accApp = new Access();
8           accApp.printThis(1, 2);
9           accApp.printThat(3, 4);
10      }
11
12      public void printThis(int x, int y) {
13          x = x;
14          y = y;
15          System.out.println("x:" + this.x + " y:" + this.y);
16      }
17
18      public void printThat(int x, int y) {
19          this.x = x;
20          this.y = y;
21          System.out.println("x:" + this.x + " y:" + this.y);
22      }
23  }
```

What is the result?

A. x:1 y:2
x:3 y:4
B. x:0 y:0
x:3 y:4

C. x:1 y:2
x:0 y:0
D. x:0 y:0
x:0 y:0

answer

On lines 13 and 14, the x and y parameters of the printThis method will obscure the x and y object fields of the A cc ess class, so the value of the A cc ess object field will not be modified to maintain the original x=0. y=0.
The first line 19, 20, x printThat method, y parameter masking A CC ess category of x, y object field, but here the use of "this" to represent the current A CC reference ess object, so you can access to A cc ess object field, the value of the field will be successfully modified to become x=3, y=4.

Question: 19

Given:

```
1   public class Access {
2
3       private int x = 0;
4       private int y = 0;
5
6       public static void main(String[] args) {
7           Access accApp = new Access();
8           accApp.printThis(1, 2);
9           accApp.printThat(3, 4);
10      }
11
12      public void printThis(int x, int y) {
13          x = x;
14          y = y;
15          System.out.println("x:" + this.x + " y:" + this.y);
16      }
17
18      public void printThat(int x, int y) {
19          this.x = x;
20          this.y = y;
21          System.out.println("x:" + this.x + " y:" + this.y);
22      }
23  }
```

What is the result?

A. x:1 y:2
x:3 y:4
B. x:0 y:0
x:3 y:4
C. x:1 y:2
x:0 y:0
D. x:0 y:0
x:0 y:0

answer

On lines 13 and 14, the x and y parameters of the printThis method will obscure the x and y object fields of the A cc ess class, so the value of the A cc ess object field will not be modified to maintain the original x=0. y=0.

The first line 19, 20, x printThat method, y parameter masking A CC ess category of x, y object field, but here the use of "this" to represent the current A CC reference ess object, so you can access to A cc ess object field, the value of the field will be successfully modified to become x=3, y=4.

Question: 20

Given the code fragment:

```
3   public static void main(String[] args) {
4       int iVar = 100;
5       float fVar = 100.100f;
6       double dVar = 123;
7       iVar = fVar;
8       fVar = iVar;
9       dVar = fVar;
10      fVar = dVar;
11      dVar = iVar;
12      iVar = dVar;
13  }
```

Whi c h three lines fail to c ompile?

A. Line 7
B. Line 8
C. Line 9
D. Line 10
E. Line 11
F. Line 12

answer

Int is a 32-bit integer type; float is a 32-bit floating point number type; double is a 64-bit floating point number type.
Int can be implied (im pl i c it) converted to float or double, float can be implied (im pl i c it) converted to double.
So this question floats to int Line 7, double to float Line 10 and double to int Line 12 will compile errors.

Question: 21

Given the code fragment:

```
for (int ii = 0; ii < 3; ii++) {
   int count = 0;
   for (int jj = 3; jj > 0; jj--) {
      if (ii == jj) {
         ++count;
         break;
      }
   }
   System.out.print(count);
   continue;
}
```

What is the result?

A. 011
B. 012

C. 123
D. 000
answer
The outer for loop will be executed three times.
The first time the outer for loop is executed, ii=0. The inner layer is for the loop , and the count range of jj is 3~1, which is executed 3 times. In the execution of this outer loop , jj will not be equal to ii, so the first outer loop is executed until the end, the value of the count variable will not be changed, and "0" will be output.
The second execution of the outer for loop , ii=1. Inner layer for loop , jj count range is 3~1, jj will be equal to ii when executed to the third time, so add count to 1, then immediately jump out of the inner for loop . The second outer loop is executed until the end, and "1" is output.
The third execution of the outer for loop , ii = 2. Inner layer for loop , jj count range is 3~1, jj will be equal to ii when executed to the second time, so add count to 1, then immediately jump out of the inner for loop . The third outer loop is executed until the end, and "1" is output.

Question: 22

Given the code fragment:

```
String[] cartoons = {"tom", "jerry", "micky", "tom"};
int counter = 0;
if ("tom".equals(cartoons[0])) {
   counter++;
} else if ("tom".equals(cartoons[1])) {
   counter++;
} else if ("tom".equals(cartoons[2])) {
   counter++;
} else if ("tom".equals(cartoons[3])) {
   counter++;
}
System.out.print(counter);
```

What is the result?
A. 1
B. 2
C. 4
D. 0
answer
The string object referenced by cartoons[0] is "tom", so the conditional expression of the first if will be true, and the value of the counter variable is incremented by one. Since the other conditions are judged to use else if, after the current condition has been established, the judgment will not be continued, so the final output is "1".

Question: 23

Given the code fragment:

```
public static void main(String[] args) {
   String str = " ";
   str.trim();
   System.out.println(str.equals("") + " " + str.isEmpty());
```

```
}
```

What is the result?

A. True true
B. True false
C. False false
D. False true

answer

The trim method of the String object can eliminate all spaces at the beginning and end, and return the new string without changing the contents of the original string object. So in this question, after the execution of the 7th line, the string object referenced by str is still " ".

Question: 24

Given:

```
public class Painting {
   private String type;
   public String getType() {
      return type;
   }
   public void setType(String type) {
      this.type = type;
   }
   public static void main(String[] args) {
      Painting obj1 = new Painting();
      Painting obj2 = new Painting();
      obj1.setType(null);
      obj2.setType("Fresco");
      System.out.print(obj1.getType() + " : " + obj2.getType());
   }
}
```

What is the result?

A. : Fres c o
B. Null : Fres c o
C. Fres c o : Fres c o
D. A NullPointerEx c eption is thrown at runtime

answer

When the "+" operator is used, if at least one of the operands is a string (object) type, the string concatenation operation is performed. If the connected string (object) reference is null, it will be treated as a "null" string of length 4.

Question: 25

Given:

```
class Dog {
   Dog() {
      try {
         throw new Exception();
      } catch (Exception e) { }
   }
}
```

```
class Test {
   public static void main(String[] args) {
      Dog d1 = new Dog();
      Dog d2 = new Dog();
      Dog d3 = d2;
      // do complex stuff
   }
}
```

How many objects have been created when the line // do complex stuff is reached?

A. Two

B. Three

C. Four

D. Six

answer

This question is asking how many objects have been created when I executed the "// do complex stuff" line. Note that the object referred to here is not only a Dog object, but all items are counted.

On the 13th line and the 14th line of the program, a Dog object is created. When creating objects Dog, will perform the third line constructor and a TRY- c AT c establish and throw in the new Ex H c eption objects. Therefore, each time a Dog object is created, two objects are actually created. When two Dog objects are created, four objects are created.

Question: 26

DoInterface.java

```
package p1;
public interface DoInterface {
   void method1(int n1); // line n1
}
```

Test.java

```
package p3;
import p1.DoInterface;
class DoClass implements DoInterface {
   public DoClass(int p1) {
   }
   public void method1(int p1) {} // line n2

   private void method2(int p1) {} // line n3
}
public class Test {
   public static void main(String[] args) {
      DoClass doi = new DoClass(100); // line n4
      doi.method1(100);
      doi.method2(100);
   }
}
```

Whi c h c hange will enable the c ode to c ompile?

A. Adding the publi c modifier to the de c laration of method1 at line n1

B. Remo vi ng the publi c modifier from the definition of method1 at line n2
C . C hanging the pr iv ate modifier on the de c laration of method 2 publi c at line n3
D. C hanging the line n4 Do C lass doi = new Do C lass();

answer
Program title previously provided will compile error because the Do C method2 method lass category of used PR iv ATE modified to modify the word, is a private members only in the Do C can only be accessed within lass category. Therefore, the main method of the Test class will have a compile error if it wants to access the method2 method of the Do C lass object.
Option A, the method defined by interfa c e will be automatically modified with the publi c modifier, so the result of adding publi c in line n1 is the same as that without publi c .
Option B, removing the publi c modifier of line n2 does not change the case where method2 can only be used in the Do C lass category.
Option C , use publi c to decorate the method2 method so that it can be used outside of the Do C lass category as the correct answer.
Option D, the Do C lass category does not have a parameterless constructor , so the modification will cause more compilation problems.

Question: 27

Which two statements correctly describe checked exception ?
THESE are EX A. C eptional C onditions that A Well-written App Li C ation Should Anti C IP ATE and Re C over from.
B. THESE EX are C eptional C onditions that are to External
The App Li C ation, and that The app li c ation usually c annot anti c ip ate or re c over from.
C . These are ex c eptional c onditions that are internal to the app lic ation, and that the app li c ation usually c annot anti c ip ate or re c over from.
D. Every c lass that is a sub c lass of RuntimeEx c eption and Error is c ategorized as checked exception .
E. Every c lass that is a sub c lass of Ex c eption, ex c luding RuntimeEx c eption and its sub c lasses, is c ategorized as checked exception .
answer
Option A, checked exception must be processed when writing the program, so this statement is correct.
Option B, this is the description of Error.
Option C , which is a description of
RuntimeEx c eption. RuntimeEx c eption is unchecked exception .
Options D, RuntimeEx c eption and Error, and exceptions and errors that inherit them are unchecked exceptions .
Option E, in addition to RuntimeEx c eption
and inheritance RuntimeEx c outside eption exceptions, Ex c eption and other inherited Ex c eption exceptions are the checked Exception .

Question: 28

Which of the following can fill in the blank in this code to make it compile? (Select 2 options.)

```
public void method() ___ Exception{
   ____ Exception();
}
```

A. On line 1, fill in throws
B. On line 1, fill in throws new
C. On line 2, fill in throw new
D. On line 2, fill in throws
E. answer
To have the method throw exceptions out, use the "throws" keyword. To throw a new exception, use the "throw" keyword. To instantiate an exception, you need to use the "new" operator.
On line 2, fill in throws new

Question: 29

View the exhibit.

```
class MissingInfoException extends Exception {
}
class AgeOutofRangeException extends Exception {
}
class Candidate {
   String name;
   int age;
   Candidate(String name, int age) throws Exception {
      if (name == null) {
         throw new MissingInfoException();
      } else if (age <= 10 || age >= 150) {
         throw new AgeOutofRangeException();
      } else {
         this.name = name;
         this.age = age;
      }
   }
   public String toString() {
      return name + " age: " + age;
   }
}
```

Given the code fragment:

```
public class Test {
   public static void main(String[] args) {
      Candidate c = new Candidate("James", 20);
      Candidate c1 = new Candidate("Williams", 32);
      System.out.println(c);
      System.out.println(c1);
   }
}
```

James age: 20
Williams age: 32
A.
Replacing line 5 with
public static void main (String [] args) throws MissingInfoException, AgeOutofRangeExcepti
on {
B.
Replacing line 5 with

public static void main (String [] args) throws.Exception {
C.
Enclosing line 6 and line 7 within a try block and adding:
catch (MissingInfoException e1) { //code goes here
}
catch (AgeOutofRangeException e2) {//code goes here
}
catch(Exception e3) { //code goes here
}
D.
Enclosing line 6 and line 7 within a try block and adding:
catch (MissingInfoException e2) { //code goes here
}
catch (AgeOutofRangeException e3) {//code goes here
}
answer
Compilation errors program provides original title will appear on line 6 and line 7, because C andidate objects have thrown Ex c eption. Since Ex c eption is a need to examine exceptions (the checked Exception), it is necessary to write the program to deal with it.
Option A, just let the main method throws MissingInfoEx c eption and AgeOutofRangeEx c eption is not enough, we must throw Ex c eption job.
Option B, the usage of throws is wrong.
Option C , with c at c h to Ex c eption, the correct option.
Option D, just let a TRY- c AT c H caught MissingInfoEx c eption and AgeOutofRangeEx c eption is not enough, we must catch Ex c eption job.

Question: 30

```
int var1 = -5;
int var2 = var1--;
int var3 = 0;
if (var2 < 0) {
   var3 = var2++;
} else {
   var3 = --var2;
}
System.out.println(var3);
```

What is the result?
A. -6 B. -4 C . -5 D. 5 E.4 F. C ompilation fails
answer
In line 7 of the program, assign the value of var1 to the var2 variable, and then decrement the value of the var1 variable by 1. At this time var1=-6, var2=-5.

The if line condition of the program line 9 is established, so the 10th itinerary is executed, and the value of var2 is assigned to the var3 variable, and then the value of the var2 variable is incremented by 1. At this time var3=-5, var2=-4.

On line 14 of the program, the value of var3 is output as "-5".

Question: 31

Whi c h statement is/are true?
I. Default c onstru c tor only c ontains "su pe r();" c all.
II. We c an't use any a cc ess modifier with a c onstru c tor.
III. A c onstru c tor should not Have a return ty pe .
A. Only I. B. Only II.
C. Only I and II. D. Only I and III. E. ALL

answer
Narrative I is correct.
Narrative II is wrong, and constructors can also use the modifiers of publi c , prete c ted, pr iv ate, etc. to adjust the visibility.
Narrative III is correct, and the constructor cannot define the type of the return value.

Question: 32

Which of the following exception will be thrown due to the statement given here?
int array[] = new int[-2];
A.NullPointerEx C eption
B. Negat IV eArraySizeEx C eption
C . ArrayIndexOut OFB oundsEx C eption
D. Index Out OF Bounds Exc eption
E. This does Statement Not CAuse the any EXCeption.

answer
The Negat ive Array Size Exception exception is thrown because the array length defined when the array is materialized is less than zero .

Question: 33

```
public class TestLoop1 {
   public static void main(String[] args) {
      int a = 0, z = 10;
      while (a < z) {
         a++;
         --z;
      }
      System.out.print(a + " : " + z);
   }
}
```

What is the result?
A.5 : 5
B.6 : 4
C .6 : 5
D.5 : 4
answer
The while loop is executed for the first time, 0 is less than 10, the conditional expression is established, a=0+1=1, z=10-1=9.
For the second execution, 1 is less than 9, and the conditional expression is established, a=1+1=2, and z=9-1=8.

The third execution, 2 is less than 8, the conditional expression is established, a=2+1=3, and z=8-1=7.
For the fourth execution, 3 is less than 7, and the conditional expression is established, a=3+1=4, and z=7-1=6.For the fifth execution, 4 is less than 6, and the conditional expression is established, a=4+1=5, and z=6-1=5.
In the sixth execution, 5 is not less than 5, so jump out of the while loop and output "5: 5".

Question: 34

```
System.out.println(2 + 4 * 9 - 3); //Line 21
System.out.println((2 + 4) * 9 - 3); // Line 22
System.out.println(2 + (4 * 9) - 3); // Line 23
System.out.println(2 + 4 * (9 - 3)); // Line 24
System.out.println((2 + 4 * 9) - 3); // Line 25
```

Which line of codes prints the highest number?

A. Line 21
B. Line 22
C . Line 23
D. Line 24
E. Line 25

answer

The formula of the Java program follows the four arithmetic rules of "first multiply and then add and subtract" and "prefix in parentheses".
The algorithm on line 21 is:

```
2 + 4 * 9 - 3 = 2 + 36 - 3 = 38 - 3 = 35
```

The algorithm on line 22 is:

```
(2 + 4) * 9 - 3 = 6 * 9 - 3 = 54 - 3 = 51
```

The algorithm on line 23 is:

```
2 + (4 * 9) - 3 = 2 + 36 - 3 = 38 - 3 = 35
```

The algorithm on line 24 is:

```
2 + 4 * (9 - 3) = 2 + 4 * 6 = 2 + 24 = 26
```

The algorithm on line 25 is:

```
(2 + 4 * 9) - 3 = (2 + 36) - 3 = 38 - 3 = 35
```

Question: 35

```
interface Runnable{
   public void run();
}
```

Which of the following will create instance of Runnable type ?

A. Runnable run = () -> {System.out.println("Run");};
B. Runnable run = () -> System.out.println("Run");
C. Runnable run = () > System.out.println("Run");
D. Runnable run = > System.out.println("Run");
E. None of the above.

answer

Runnable run = () -> {System.out.println("Run");};
Java 8 imports the Lambda syntax, you can refer to this article to learn:

Question: 36

```
class Jump {
   static String args[] = {"lazy", "lion", "is", "always"};
   public static void main(String[] args) {
      System.out.println(args[1] + " " + args[2] + " " + args[3] + " jumping");
   }
}
```

And the c ommands:

```
Javac Jump. java
java Jump c razy elephant is always
```

What is the result?

A. Lazy lion is jumping
B. Lion is always jumping
C. c razy elephant is jumping
D. Elephant is always jumping
E. C ompilation fails

answer

Instruction execution topic will compile "Jump. The Java " category due Jump helpful publi c modified, so the default main method executes Jump category, and the " c Razy Elephant IS Always" as the main parameter. The index value of the parameter starts at 0.
The args parameter of the main method shadows the args category (static) variable of the Jump class.
Therefore, "elephant is always jumping" is output.

Question: 37

```
public class Test {
   static double dvalue;
   static Test ref;
   public static void main(String[] args) {
      System.out.println(ref);
      System.out.println(dvalue);
   }
}
```

What is the result?

A.	B	C .
p1.Test. c lass 0.0	0.000000	Null 0.0

D. C ompilation fails
E. A NullPointerEx c eption is thrown at runtime

answer

Dvalue and ref are the class variables of Test and are automatically initialized when they are declared. The type of dvalue is double and will be initialized to 0. The type of ref is Test, which is the object reference type and will be initialized to null.

The println method will output "null" directly if the reference passed in is null.
So this question the answer is option C .

Question: 38

```
package p1;
public interface DoInterface {
   void m1(int n); // line n1
   public void m2(int n);
}

package p3;
public class DoClass implements DoInterface{
   int x1, x2;
   DoClass(){
      this.x1 = 0;
      this.x2 = 10;
   }
   public void m1(int p1) { x1+=p1; System.out.println(x1); } // line n2
   public void m2(int p1) { x2+=p1; System.out.println(x2); }
}

package p2;
import p1.*;
import p3.*;

class Test {
   public static void main(String[] args) {
      DoInterface doi = new DoClass(); // line n3
      doi.m1(100);
      doi.m2(200);
   }
}
```

What is the result?
A.

```
100
210
```

B. C ompilation fails Total Due to AN error in Line N1
C . C ompilation fails Total Due to AT AN error Line n2
D. Compilation fails Total Due to AT AN error Line n3
answer
Line n3 materializes the object entity of the Do C lass category, but since the construct of Do C lass does not use the publi c modifier declaration, it can only be instantiated within the same suite (pa c kage). Compilation errors can occur if the Do C lass object is instantiated under a different suite .

Question: 39

Given:

```
class Mid {
   public int findMid(int n1, int n2) {
      return (n1 + n2) / 2;
   }
}
public class Calc extends Mid {
   public static void main(String[] args) {
      int n1 = 22, n2 = 2;
      // insert code here
      System.out.print(n3);
   }
}
class Mid {
   public int findMid(int n1, int n2) {
      return (n1 + n2) / 2;
   }
}
public class Calc extends Mid {
   public static void main(String[] args) {
      int n1 = 22, n2 = 2;
      // insert code here
      System.out.print(n3);
   }
}
```

Whi c h two c ode fragments, when inserted at // insert c ode here, enable the c ode to c ompile and print 12?

A. Java Calc c = new Calc(); int n3 = c.findMid(n1, n2); 12 13 Calc c = new Calc(); int n3 = c.findMid(n1, n2);	B. Java int n3 = super.findMid(n1, n3); 12 int n3 = super.findMid(n1, n3);
C. Java Calc c = new Mid(); int n3 = c.findMid(n1, n2); 12	D. Java Mid m1 = new Calc(); int n3 = m1.findMid(n1, n2); 12

13 Calc c = new Mid(); int n3 = c.findMid(n1, n2);	13 Mid m1 = new Calc(); int n3 = m1.findMid(n1, n2);
E. Java int n3 = Calc.findMid(n1, n2); 12 int n3 = Calc.findMid(n1, n2);	

Answer :
findMid is an object method, you must materialize the C al c or Mid object to use, so options B, E are wrong.

The C al c category inherits the Mid category, so the C al c object type can be implied (im pl i c it) into a Mid object type, and vice versa, so option C is wrong.

The last remaining options A, D are correct.

Question: 40

Given:
Java

```
public class String1 {
   public static void main(String[] args) {
      String s = "123";
      if (s.length() > 2) {
         s.concat("456");
      }
      for (int x = 0; x < 3; x++) {
         s += "x";
      }
      System.out.println(s);
   }
}
public class String1 {

   public static void main(String[] args) {
      String s = "123";
      if (s.length() > 2) {
         s.concat("456");
      }
      for (int x = 0; x < 3; x++) {
         s += "x";
      }
      System.out.println(s);
```

```
   }
}
```

What is the result?

A. 123 B. 123xxx C . 123456 D. 123456xxx E. C ompilation fails

Answer :

On line 6 of the program, using the " c on c at" method of the string object will return the result of the string concatenation, but will not change the string represented by the object itself. Therefore, this question will only string three "x"s after the string "123".

Question: 41

Given:

```
public class ColorTest {
   public static void main(String[] args) {
      String[] colors = {"red", "blue", "green", "yellow", "maroon", "cyan"};
      int count = 0;
      for (String c : colors) {
         if (count >= 4) {
            break;
         } else {
            continue;
         }
         if (c.length() >= 4) {
            colors[count] = c.substring(0, 3);
         }
         count++;
      }
      System.out.println(colors[count]);
   }
}
public class ColorTest {
   public static void main(String[] args) {
      String[] colors = {"red", "blue", "green", "yellow", "maroon", "cyan"};
      int count = 0;
      for (String c : colors) {
         if (count >= 4) {
            break;
         } else {
            continue;
         }
         if (c.length() >= 4) {
            colors[count] = c.substring(0, 3);
         }
         count++;
      }
      System.out.println(colors[count]);
   }
}
```

What is the result?

A. Yellow B. Maroon C . C ompilation fails Total

D. A StringIndexOut OFB oundsEx C eption thrown IS AT Runtime.

Answer :
Line 12 will compile the error, because the if line of the 7th line will not skip the condition of the conditional formula, it will directly skip the program after the for loop , jump out of the loop or enter the next loop , so the 12th line The program will never execute and will be checked during the compilation phase.

Question: 42

Given:
Java

```
public class SuperTest {
   public static void main(String[] args) {
      // statement1
      // statement2
      // statement3
   }
}
class Shape {
   public Shape() {
      System.out.println("Shape: constructor");
   }
   public void foo() {
      System.out.println("Shape: foo");
   }
}
class Square extends Shape {
   public Square() {
      super();
   }
   public Square(String label) {
      System.out.println("Square: constructor");
   }
   public void foo() {
      super.foo();
   }
   public void foo(String label) {
      System.out.println("Square: foo");
   }
}
public class SuperTest {
   public static void main(String[] args) {
      // statement1
      // statement2
      // statement3
   }
}

class Shape {
   public Shape() {
```

```
        System.out.println("Shape: constructor");
    }
    public void foo() {
        System.out.println("Shape: foo");
    }
}

class Square extends Shape {
    public Square() {
        super();
    }

    public Square(String label) {
        System.out.println("Square: constructor");
    }

    public void foo() {
        super.foo();
    }

    public void foo(String label) {
        System.out.println("Square: foo");
    }
}
```

What should statement1, statement2, and statement3, be respectively, in order to produce the result?

Shape: constructor
Square: foo
Shape: foo

A.
 	
Square square = new Square("bar");
 	
square.foo("bar");
 	
square.foo();
 	
Square square = new Square("bar");
 	
square.foo("bar");
 	
square.foo();
	B.
	
	Square square = new Square("bar");
	
	square.foo("bar");
	
	square.foo("bar");
	
	Square square = new Square("bar");
	
	square.foo("bar");
	
	square.foo("bar");

C. Square square = new Square(); square.foo();	D. Square square = new Square(); square.foo();

square.foo(bar);
 	
Square square = new Square();
 	
square.foo();
 	
square.foo(bar);
	square.foo("bar");
	
	Square square = new Square();
	
	square.foo();
	
	square.foo("bar");

E.
 	
Square square = new Square();
 	
square.foo();
 	
square.foo();
 	
Square square = new Square();
 	
square.foo();
 	
square.foo();
	F.
	
	Square square = new Square();
	
	square.foo("bar");
	
	square.foo();
	
	Square square = new Square();
	
	square.foo("bar");
	
	square.foo();

Answer :
The first action of the output is "Sh ape : c onstru c tor", so the program of line 13 is executed only when the Sh ape object is instantiated . Square Category inherited Sh APE category, do not pass a parameter entity of the Square object will perform to Sh APE 's constructor . So for now, statement1 can instantiate a Square object or a Sh ape object, and options A and B are wrong.

The second behavior of the output is "Square: foo", which is the foo(String label) object method in the Square category of the program. So statement1 should instantiate the Square object, and must pass the string argument when statement2 calls the foo method of the Square object. Therefore options C , D, E are wrong.

The third behavior of the output, "Square: foo", is to execute the foo() method in the Sh ape category on line 17 of the program . This method is overwritten in line 31 of the program, which is the Square category, but the overridden program uses "su pe r" to call the original foo() method of the Sh ape category, so call the foo of the Square object (The method will also output "Square: foo". Option F is the correct answer

Question: 43

Given the code format:
Java
```
class DBConfiguration {
   String user;
   String password;
}
class DBConfiguration {
   String user;
   String password;
}
```

And,

Java
```
public class DBHandler {
   DBConfiguration configureDB(String uname, String password) {
      // insert code here
   }
   public static void main(String[] args) {
      DBHandler r = new DBHandler();
      DBConfiguration dbConf = r.configureDB("manager", "manager");
   }
}
public class DBHandler {
   DBConfiguration configureDB(String uname, String password) {
      // insert code here
   }
   public static void main(String[] args) {
      DBHandler r = new DBHandler();
      DBConfiguration dbConf = r.configureDB("manager", "manager");
   }
}
```
Which code fragment must be inserted at line 6 to enable the code to compile?

A.	B.	C.	D.
DBConfiguration f; return f; 6 7 DBConfiguration f; return f;	return DBConfiguration; 6 return DBConfiguration;	return new DBConfiguration(); 6 return new DBConfiguration();	retutn 0; 6 retutn 0;

Answer :
The c onfigureDB method on line 5 of the program returns the DB C onfiguration object, so line 6 needs to instantiate the DB C onfiguration object and return the result to complete the compilation.
Option A, f is the object reference variable and needs to be initialized before it can be used.
Option B, you cannot directly return a category.
Option C , the correct answer.
For option D, the return type cannot be a value of 0, but the DB C onfiguration object is correct.

Question: 44

```
public class Test {
 public static void main(String[] args) {
```

```
        Integer num = Integer.parseInt(args[1]);
        System.out.println("Number is : " + num);
    }
}
```

And the commands:

```
javac Test.java
java Test 12345
```

What is the result?

A. Number us : 12345

B. A NullPointerException is thrown at runtime

C. A NumberFormatException is thrown at runtime

D. An ArrayIndexOutOfBoundException is thrown at runtime.

answer

The command executed by the title will compile "Test. java ". Since the Test category is modified with publi c , the main method of the Test category is executed by default, and "12345" is used as the main parameter. The index value of the parameter starts at 0.

Line 4 of the program takes the value of parameter index 1, but there is only one parameter here, so it will exceed the array index range, and ArrayIndexOut OfB oundEx c eption is thrown .

Question: 45

```
int row = 10;
for (; row > 0;) {
    int col = row;
    while (col >= 0) {
        System.out.print(col + " ");
        col -= 2;
    }
    row = row / col;
}
```

What is the result?

A. 10 8 6 4 2 0

B. 10 8 6 4 2

C. AnArithmeticException is thrown at runtime

D. The program goes into an infinite loop outputting: 10 8 6 4 2 0. . .

E. Compilation fails

answer

The for loop of line 13 is executed when the value of the row variable is greater than zero.

The first execution of the for loop , row = 10, c ol = 10, so the while loop will repeat, print "10 8 6 4" ... in sequence, until the value of the c ol variable is less than 0. The last time the while loop is executed , the value of c ol is -2, which is less than 0, so it jumps out of the while loop .

Then change the value of row to 10/-2=-5, and then prepare to enter the next for loop . However, at this time, the row is less than or equal to 0, so it will jump out of the for loop .

Question: 46

```
public static void main(String[] args) {
   String theString = "Hello World";
   System.out.println(theString.charAt(11));
}
```

What is the result?
A. The program prints nothing
B.
d
C. A StringIndexOutOfBoundsException is thrown at runtime.
D. An ArrayIndexOutOfBoundsException is thrown at runtime.
E. A NullPointerException is thrown at runtime.
answer
theString variable reference to the string length is 11 " Hello World " program line 7 to get " Hello World " index character position 11, would exceed the string length, and therefore will throw StringIndexOut OFB oundsEx c The eption exception.

Question: 47

```
public class Whizlabs {
  public static void main(String[] args) {
       LocalDate date = LocalDate.of(2015, 3, 26);
       Period p = Period.ofDays(1);
       System.out.println(date.plus(p));
    }
}
```

What is the output?
A. 2015-03-27
B. 2015-04-27
C. 2015-02-27
D. Compilation fails due to error at line 6.
E. Compilation fails due to error at line 8.
answer
This question is in the date and time (Date-Time) API added to Java 8 , you can refer to the following article:
Https://magiclen.org/java-8-date-time-api/
On line 4 of the program, a LocalDate object date was created on March 26, 2015 .
Program line 5, the establishment of a day Pe RIOD object p.
In the sixth line of the program, the "Day of March 26, 2015" represented by the date object plus the "one day" represented by the periodic object p will return the LocalDate object representing "March 27, 2015" .

Question: 48

Which three statements are true about the structure of a Java class?
A. A class can have only one private constructor.
B. A method can have the same name as a field.

C. A class can have overloaded static methods.
D. A public class must have a main method.
E. The methods are mandatory components of a class.
F. The fields need not be initialized before use.
answer
Option A, a category can have a private decorated private constructor. This is true, Java does not limit the number of private constructors, for example:

Java
```
public class Test {
   private Test(){
   }
   private Test(int a){
   }
   private Test(String s){
   }
}
public class Test {
   private Test(){
   }
   private Test(int a){
   }
   private Test(String s){
   }
}
```
Option B, the name of a method can be the same as the field name. This is true, for example:

Java
```
public class Test {
   int A;
   int A(){
      return A;
   }
}
public class Test {
   int A;

   int A(){
      return A;
   }
}
```
Option C, a category can have a static category of overload. This is true, for example:

Java
```
public class Test {
   public static void test(String a, String b){
   }
   public static void test(int a, int b){
   }
}
```

```
public class Test {
   public static void test(String a, String b){
   }
   public static void test(int a, int b){
   }
}
```

Option D, a public category does not have to have a main method.
Option E, not every category needs to write a method, or even write nothing in the category.
Option F, the field is not initialized, but is initialized when it is announced.

Question: 49

Which three statements are true about the structure of a Java class?
A. A class can have only one private constructor.
B. A method can have the same name as a field.
C. A class can have overloaded static methods.
D. A public class must have a main method.
E. The methods are mandatory components of a class.
F. The fields need not be initialized before use.
answer
Option A, a category can have a private decorated private constructor. This is true, Java does not limit the number of private constructors, for example:

```
public class Test {
   private Test(){
   }
   private Test(int a){
   }
   private Test(String s){
   }
}
public class Test {
   private Test(){
   }
   private Test(int a){
   }
   private Test(String s){
   }
}
```

Option B, the name of a method can be the same as the field name. This is true, for example:

```
Java
public class Test {
   int A;
   int A(){
      return A;
   }
}
public class Test {
   int A;
   int A(){
```

```
        return A;
    }
}
```

Option C, a category can have a static category of overload. This is true, for example:

```
public class Test {
    public static void test(String a, String b){
    }
    public static void test(int a, int b){

    }
}
public class Test {
    public static void test(String a, String b){
    }
    public static void test(int a, int b){
    }
}
```

Option D, a public category does not have to have a main method.
Option E, not every category needs to write a method, or even write nothing in the category.
Option F, the field is not initialized, but is initialized when it is announced.

Question: 50

```
24 float var1 = (12_345.01 >= 123_45.00) ? 12_456 : 14_56.02f;
25 float var2 = var1 + 1024;
26 System.out.print(var2);
```

What is the result?
A. 13480.0
B. 13480.02
C. Compilation fails
D. An exception is thrown at runtime
answer
First, see the judgment line "12_345.01 >= 123_45.00" on line 24, which is obviously true, so the value of var1 is "12456". The value of var2 is 12456+1024=13480.

Question: 51

```
1 public class App {
2
3    public static void main(String[] args) {
4        int i = 10;
5        int j = 20;
6        int k = j += i / 5;
7        System.out.print(i + " : " + j + " : " + k);
8    }
9 }
```

What is the result?
A. 10 : 22 : 20
B. 10 : 22 : 22
C. 10 : 22 : 6

D. 10 : 30 : 6

answer

"j += i / 5" divides the 10 stored in the i variable by 5, adds the value back to the j variable, and then returns the value of the last j variable.

Question: 52

```
1  class Caller {
2
3      private void init() {
4          System.out.println("Initialized");
5      }
6
7      public void start() {
8          init();
9          System.out.println("Started");
10     }
11 }
12
13 public class TestCall {
14
15     public static void main(String[] args) {
16         Caller c = new Caller();
17         c.start();
18         c.init();
19     }
20 }
```

What is the result?

A.

```
Initialized
Started
```

B.

```
Initialized
Started
Initialized
```

C. Compilation fails

D. An exception is thrown at runtime

answer

Line 18 uses the program C the init method aller objects. Since the init method is PR iv ATE modification being modified, it is only visibility in C within this category of aller, so the Test C All categories can not go to the call to C init method aller object, an error occurs at compile time.

Question: 53

Which usage represents a valid way of compiling java source file with the name "Main"?

A. javac Main.java

B. java Main.class

C. java Main.java

D. javac Main

E. java Main

answer
To compile Java source code, use the javac compiler. The usage is as follows:
The full path to the javac " .java " file
So option A is correct.

Question: 54

```
public static void main(String[] args) {
   int x = 5;
   while (isAvailable(x)) {
      System.out.print(x);
  }
}

public static boolean isAvailable(int x) {
   return x-- > 0 ? true : false;
}
```

Which modification enables the code to print 54321?
A. Replace line 6 with System.out.print(--x);
B. At line 7, insert x--;
C. Replace line 6 with --x; and, at line 7, insert System.out.print(x);
D. Replace line 12 With return (x > 0) ? false : true;
answer
On line 12 of the program, using "x--" in the isAvailable method does not affect the x variable of the main method, because Java is always "pass by value".
Option A will make the output "43210".
Option B, you can successfully output "54321".
Option C , such logic is the same as option A.
Option D, while loop will not be executed.

Question: 55

Given the code fragment?

```
1  public class Test {
2
3      public static void main(String[] args) {
4          Test t = new Test();
5          int[] arr = new int[10];
6          arr = t.subArray(arr, 0, 2);
7      }
8
9      // insert code here
10 }
```

Which method can be inserted at line // insert code here to enable the code to compile?
A.

```
public int[] subArray(int[] src, int start, int end) {
```

```
    return src;
}
```

B.

```
public int subArray(int src, int start, int end) {
    return src;
}
```

C.

```
  public int[] subArray(int src, int start, int end) {
      return src;
  }
```

D.

```
      public int subArray(int[] src, int start, int end) {
          return src;
      }
```

answer

Line 6 of the program uses the subArray method of the Test object referenced by the t variable, but the existing program does not define and implement the subArray method. There is no need to consider whether the execution result of the subArray method is correct. The problem only requires compilation to succeed.

The three arguments passed in the subArray method on line 6 are an array of integers, integers, and integers, respectively, and the return type is an array of integers, so option A is correct.

Question: 56

```
public class Msg {
 public static String doMsg(char x) {
      return "Good Day!";
   }
  public static String doMsg(int y) {
      return "Good Luck!";
   }
  public static void main(String[] args) {
      char x = 8;
      int z = '8';
      System.out.println(doMsg(x));
      System.out.print(doMsg(z));
   }
}
```

What is the result?

A.

```
GoodDay!
Good Luck!
```

B.

```
Good Day!
Good Day!
```

C .

Good Lu c k!
Good Day!
D.
Good Lu c k!
Good Lu c k!
E. C ompilation fails
answer
On line 12, the value 8 is assigned to the character variable x for storage. In line 13, the character value of the character "8" is assigned to the integer z variable for storage.
Line 14 because the x variable is a character type, it will call the doMsg multi-load method on line 3 and output "Good Day!".
Line 15 because the x variable is an integer type, it calls the doMsg multi-load method on line 7 and outputs "Good Luck!".

Question: 57

```
class X {
  int x1, x2, x3;
}

class Y extends X {
 int y1;
  Y() {
       x1 = 1;
       x2 = 2;
       y1 = 10;
    }
}

class Z extends Y {
  int z1;

    Z() {
       x1 = 3;
       y1 = 20;
       z1 = 100;
    }
}
public class Test3 {
    public static void main(String[] args) {
       Z obj = new Z();
       System.out.println(obj.x3 + ", " + obj.y1 + ", " + obj.z1);
    }
}
```

Which constructor initializes the variable x3?
The default Only A. ConstruCTor of CLASS X-
B. Only argument The NO- ConstruCTor of CLASS the Y
C . The NO-Only argument ConstruCTor of CLASS the Z
D. Only The default Constructor of object class

answer

The 31st line of the program physically draws the Z object and stores the materialized object reference to the variable obj. When the Z object is materialized, the constructor of the Z category in line 21 is executed first. Since the constructor does not use "super" or "this" to specify which constructor to call first, the preset will be compiled. Add "super();" to the first line. The parent category of the Z category is the Y category, so the Z category constructor calls "super()" to execute the constructor of the Y-category in line 10. For the same reason, the constructor of the Y category will also call the constructor of the X category. The X category does not have a constructor, so the constructor of the default X category is automatically added during the compile time:

```
4 public X(){
5     super();
6 }
```

In the process of materializing the Z object, the value of the x3 variable is not initialized when the constructor is executed . When the x3 integer variable is declared, it is preset to 0, so the description is reasonable. Is option A.

Question: 58

Which of the following can fill in the blank in this code to make it compile?

```
1 public class Exam {
2
3     void method() {
4     }
5 }
```

```
1 public class OCAJP extends Exam{
2     _____ void method(){}
3 }
```

A.abstract
B.final
C.private
D.default
E.int

answer

In Java , if a subcategory overrides a method of a parent class, its method's visibility can only be equal to or greater than the parent class's method. Therefore, when the method to be overwritten does not use the visibility modifier (default), the overwrite can only be used without the visibility modifier or the higher visibility range of prote c ted or publi c . However, there is no suitable visibility option available in this topic, so you should leave the modifier without using the visibility modifier.

So option B, using the final modifier that makes the method no longer overwritten is the best answer

---

Question: 59

```
public class Test {
   public static void main(String[] args) {
      boolean isChecked = false;
      int arry[] = {1, 3, 5, 7, 8, 9};
      int index = arry.length;
      while (~code1~) {
         if (arry[index - 1] % 2 == 0) {
            isChecked = true;
         }
         ~code2~
      }
      System.out.print(arry[index] + ", " + isChecked);
   }
}
```

Which set of changes enable the code to print 1, true?

A. Replacing with index > 0 and replacing with index--;

B. Replacing with index > 0 and replacing with --index;

C. Replacing with index > 5 and replacing with --index ;

D. Replacing with index and replacing with --index ;

answer

The final arry[index] of the program needs to be 1; is Checked needs to be true. The value of index at the beginning is 6.

Option A, can successfully make the index variable value 0 when jumping out of the while loop , and modify the value of the is C he c ked variable to true.

Option B, the reason is the same as option A.

Option C , the while loop is only executed once, and the value stored in the index variable is changed to 5, the value of is C he c ked is because arry[5] % 2 is 1, so the output will become:

```
9, false
```

The option D, index is not a Boolean variable, so it cannot be directly used as a conditional expression of the while loop , and a compile error will occur.

Question: 60

```
public class App {
   // Insert code here
      System.out.print("Welcome to the world of Java");
   }
}
```

Which two code fragments, when inserted independently at line // Insert code here, enable the program to execute and print the welcome message on the screen?

A. static public void main (String[] args) {

B.
static void main (String[] args) {

C. public static void Main (String[] args) {

D.
public static void main (String[] args) {

E.
public void main (String[] args) {
answer
The program given by the title lacks the program entry point - the "main" static method . So you need to add it correctly to the program.
Option A, use stati c and publi c to decorate the "main" method, making it a static and public method, which is how the correct program entry point is used.
Option B, does not use publi c to modify the "main" method, can not be used as a program entry point . Although the program can compile, it will not find the "main" method that can be entered.
Option C , the name of the "main" method as the program entry point can only be "main", not "Main", otherwise it will not be used as a program entry point .
Option D, correct, reason is the same as option A.
Option E, without using stati c to modify the "main" method, can not be used as a program entry point . Although the program can compile, it will not find the "main" method that can be entered.

Question: 61

```
1  public class TestField {
2
3      int x;
4      int y;
5
6      public void doStuff(int x, int y) {
7          this.x = x;
8          y = this.y;
9      }
10
11     public void display() {
12         System.out.print(x + " " + y + " : ");
13     }
14
15     public static void main(String[] args) {
16         TestField m1 = new TestField();
17         m1.x = 100;
18         m1.y = 200;
19         TestField m2 = new TestField();
20         m2.doStuff(m1.x, m1.y);
21         m1.display();
```

```
22        m2.display();
23    }
24 }
```

What is the result?
A. 100 200 : 100 200 :
B. 100 0 : 100 0 :
C. 100 200 : 100 0 :
D. 100 0 : 100 200 :
answer
First, see line 8 of the program. The " doStuff" method does not assign the value of the parameter y to the y variable of the object, so the " doStuff" method will only change the x variable of the animal.
Lines 17~18, directly set the two object variables x and y of the m1 object . After the setting is completed, m1.x is 100, and m1.y is 200.
The " doStuff" method using the m2 object on line 20 only changes the object variable x, so m2.x is 100 and m2.y is 0 after the setting is completed.
So option C is correct.

Question: 62

```
public static void main(String[] args){
   ArrayList<String> list = new ArrayList<>();
   list.add("SE");
   list.add("EE");
   list.add("ME");
   list.add("SE");
   list.add("EE");

   list.remove("SE");

   System.out.print("Values are : " + list);
}
```

What is the result?
A. Values are : [EE, ME] B. Values are : [EE, EE, ME]
C. Values are : [EE, ME, EE] D. Values are : [SE, EE, ME, EE]
E. Values are : [EE, ME, SE, EE]
answer
The List collection allows multiple elements with the same logic. The remove method will look for the first matching element from the beginning and remove it, so only the "SE" string of index 0 will be removed in this question.

Question: 63

```
import java.util.ArrayList;
import java.util.List;

public class Whizlabs{
   public static void main(String[] args){
```

```
        List<Integer> list = new ArrayList<>();
        list.add(21); list.add(15);
        list.add(30); list.add(11);
        list.add(2);
        //insert here
        System.out.println(list);
    }
}
```

Which inserted at line 11, will provide the following output?

```
[21, 15, 11]
```

A. 1 list.removeIf(e > e % 2 != 0);

B. list.removeIf(e -> e % 2 == 0);

C. list.remove(e -> e % 2 == 0);

D. None of the above.

answer

Option A will remove all items in the integer set that cannot be divisible by 2. So the elements in the collection will have only "30" and "2" left.
Option B removes all items in the integer set that can be divisible by 2. So the elements in the collection will have "21", "15" and "11".
Option C , the usage of the remove method is incorrect.
Option D, because option B is the answer, so the error.

Question: 64

Given the code in a file Traveler.java:

```
1  class Tours {
2
3      public static void main(String[] args) {
4          System.out.print("Happy Journey! " + args[1]);
5      }
6  }
7
8  public class Traveler {
9
10     public static void main(String[] args) {
11         Tours.main(args);
12     }
13 }
```

And the commands:

```
javac Traveler.java
java Traveler Java Duke
```

What is the result?

A. Happy Journey! Duke

B. Happy Journey! Java

C. An exception is thrown at runtime

D. The program fails to execute due to a runtime error

answer

On line 11 of the program, call the main static method of the Tours category and continue passing the main args argument as an

argument. In C pass to the next LI interface Java programming parameters index value is zero, the subject of arguments to Traveler category is " Java Duke" so args [0] = " Java ", args [1] = "Duke".

Question: 65

Given the code fragment:

```
10 String color = "teal";
11
12 switch (color) {
13     case "Red":
14         System.out.println("Found Red");
15     case "Blue":
16         System.out.println("Found Blue");
17         break;
18     case "Teal":
19         System.out.println("Found Teal");
20         break;
21     default:
22         System.out.println("Found Default");
23 }
```

What is the result?

A.

```
Found Red
Found Default
```

B.

```
Found Teal
```

C .

```
Found Red
Found Blue
Found Teal
```

D.

```
Found Red
Found Blue
Found Teal
Found Default
```

E.

```
Found Default
```

answer

The "break" keyword can be used in swit c h to make the program jump out of swit c h immediately . In this title, to judge the string is "teal", although "Teal" in c ASE, but they are "T" of the case do not meet, therefore SWIT c H is performed only to 21 to 22 lines, output "Found Default".

Question: 66

```
public class FieldInit {

    char c;
    boolean b;
```

```
    float f;

    void printAll() {
        System.out.println("c = " + c);
        System.out.println("c = " + b);
        System.out.println("c = " + f);
    }

    public static void main(String[] args) {
        FieldInit f = new FieldInit();
        f.printAll();
    }
}
```

What is the result?

A.	B.	C .	**D.**
c = null b = false f = 0.0F	c = 0 b = false f = 0.0f	c = null b = true f = 0.0	**c =** **b=false** **f = 0.0**

answer

The category or the field of the object, the initial value of the character type is "'\0'" (the character value is 0), the initial value of the numeric type is "0", and the initial value of the Boolean type is " False", the initial value of the object type is "null".

Question: 67

Given:

```
1  class Alpha {
2
3      int ns;
4      static int s;
5
6      Alpha(int ns) {
7          if (s < ns) {
8              s = ns;
9              this.ns = ns;
10         }
11     }
```

```
12
13    void doPrint() {
14        System.out.println("ns = " + ns + " s = " + s);
15    }
16 }
17
18 public class TestA {
19
20    public static void main(String[] args) {
21        Alpha ref1 = new Alpha(50);
22        Alpha ref2 = new Alpha(125);
23        Alpha ref3 = new Alpha(100);
24        ref1.doPrint();
25        ref2.doPrint();
26        ref3.doPrint();
27    }
28 }
```

What is the result?

A.	**B.**	C .	D.
Ns = 50 s = 125 ns = 125 s = 125 ns = 100 s = 125	**Ns = 50 s = 125** **ns = 125 s = 125** **ns = 0 s = 125**	Ns = 50 s = 50 ns = 125 s = 125 ns = 100 s = 100	Ns = 50 s = 50 ns = 125 s = 125 ns = 0 s = 125

answer

Ns is the object variable , s is the category variable , and s changes in the object will affect other objects. In the Alpha constructor on line 6 , ns and s are initialized with a larger value than the original s. The main method does not pass the value to Alpha's constructor and instantiates the different objects from small to large . The 125th line passed in the 22nd line is the maximum value, and the 23rd line passed to the constructor 's 100 will make the 7th line. The if conditional is not true, and the value of the ns variable that is not initialized in the constructor is preset to zero.

So the answer is option B.

Question: 68

Given:

```
1  class X {
2
3     public void mX() {
4        System.out.println("Xm1");
5     }
6  }
```

```
7
8  class Y extends X {
9
10    public void mX() {
11       System.out.println("Xm2");
12    }
13
14    public void mY() {
15       System.out.println("Ym");
16    }
17 }
18
19 public class Test {
20
21    public static void main(String[] args) {
22       X xRef = new Y();
23       Y yRef = (Y)xRef;
24       yRef.mY();
25       xRef.mX();
26    }
27 }
```

What is the result?

A.

```
Ym
Xm2
```

B.

```
Ym
Xm1
```

C. Compilation fails

D. A ClassCastException is thrown at runtime

answer

In line 22 of the program, the reference to the Y object entity is assigned to the variable storage of the X type. Since the Y category inherits the X category, an upward transformation of the implied (implicit) type is possible.

In the 23rd line of the program, the X-type variable is transformed down to the Y-type by the explicit (ex pl i c it) method, and the object referred to by the X variable is the Y object, so this transformation is also possible.

Then, in lines 24 and 25, the mY method and the 10th line mX method of the Y object in the 14th line are executed. Therefore, "Ym" and "Xm2" are printed separately.

Question: 69

Given the code fragment:

```
9  int a = -10;
10 int b = 17;
11 int c = expression1;
12 int d = expression2;
13 c++;
```

```
14 d--;
15 System.out.print(c + ", " + d);
```

What could expression1 and expression2 be, respectively, in order to produce output –8, 16?
A. ++a,--b **B.** ++a,b-- C. a++,--b D. a++, b--

Answer
This question can be calculated back to the 12th line, and finally c = -8 - 1 = -9, d = 16 + 1 = 17.
So on line 12, you want to assign the value of b directly to d, so option A and option C are not correct. On line 11, you want to increment the value of a and assign it to c , so option C and option D are not correct. So the answer is option B.

Question: 70

Given:

```
1 public class Whizlabs {
2    public static void main(String[] args) {
3       StringBuilder sb = new StringBuilder("1Z0");
4       sb.concat("-808");
5       System.out.println(sb);
6    }
7 }
```

What is the output?
A.

```
1Z0
```

B.

```
1Z0-808
```

C. An exception will be thrown.
D. Compilation fails due to error at line 3.
E. Compilation fails due to error at line 4.
answer
The StringBuilder object does not have a " c on c at" method, so the fourth line of the program will compile errors. If you want to concatenate strings, you should use the " app end" method of StringBuilder .

Question: 71

Given:

```
1  public class Test {
2     public static void main(String[] args) {
3        int ax = 10, az = 30;
4        int aw = 1, ay = 1;
5        try {
6           aw = ax % 2;
7           ay = az / aw;
8        } catch (ArithmeticException e1) {
9           System.out.println("Invalid Divisor");
10       } catch (Exception e2) {
```

```
11          aw = 1;
12          System.out.println("Divisor Changed");
13       }
14       ay = az / aw; // Line 14
15       System.out.println("Succesful Division " + ay);
16    }
17 }
```

What is the result?

A.

```
Invalid Divisor
Divisor Changed
Successful Division 30
```

B.

```
Invalid Divisor
Successful Division 30
```

C.

```
Invalid Divisor
Exception in thread "main" java.lang.ArithmeticException: / by zero
at test.Teagle.main(Teagle.java:14)
```

D.

```
Invalid Divisor
Exception in thread "main" java.lang.ArithmeticException: / by zero
at test.Teagle.main(Teagle.java:14)
Successful Division 1
```

answer

The value of the "aw" variable on line 6 is the remainder of 10 divided by 2, which is 0. So line 7 takes the "aw" variable with a value of 0 as a divisor and throws Arithmetic Exception, which is then received by catch on line 8 .

Then the program executes to line 14, and still throws Arithmetic Exception, but this Arithmetic Exception will be thrown directly to the thread by the main method .

Question: 72

Given:

```
1 public class MyFor1 {
2    public static void main(String[] args) {
3       int[] x = {6, 7, 8};
4       for (int i : x) {
5          System.out.print(i + " ");
6          i++;
7       }
8    }
9 }
```

What is the result?

A.	B.	C.	D.
6 7 8	7 8 9	0 1 2	6 8 10

E. Compilation fails

answer

This question simply outputs the array elements one by one using the fore c h of line 4 of the array content. Line 6 of "i++" is executed after the output of "i" on line 5, so it does not affect the output of our array.

Question: 73

What is the result?

```
1  abstract class A1 {
2
3      public abstract void m1();
4
5      public void m2() {
6          System.out.println("Green");
7      }
8  }
9
10 abstract class A2 extends A1 {
11
12     public abstract void m3();
13
14     public void m1() {
15         System.out.println("Cyan");
16     }
17
18     public void m2() {
19         System.out.println("Blue");
20     }
21 }
22
23 public class A3 extends A2 {
24
25     public void m1() {
26         System.out.println("Yellow");
27     }
28
29     public void m2() {
30         System.out.println("Pink");
31     }
32
33     public void m3() {
34         System.out.println("Red");
35     }
36
37     public static void main(String[] args) {
38         A2 tp = new A3();
```

```
39        tp.m1();
40        tp.m2();
41        tp.m3();
42    }
43 }
```

What is the result?

A.	B.	C.	D. Compilation Fails
Yellow Pink Red	Cyan Blue Red	Cyan Green Red	

answer

The tp variable refers to the object entity of the A3 category, and the A3 category overrides the m1, m2, m3 methods.
Line 39 of the program executes the m1 method of the A3 category override and outputs "Yellow".
On line 40 of the program, the m2 method of the A3 category override is executed, and "Pink" is output.
Line 41 of the program executes the m2 method of the A3 category override and outputs "Red".

Question: 74

Given the code fragment

```
1  class Test2 {
2
3      int fvar;
4      static int cvar;
5
6      public static void main(String[] args) {
7          Test2 t = new Test2();
8          // insert code here to write field variables
9      }
10 }
```

Which code fragments, inserted independently, enable the code compile?

A. t.fvar = 200;
B. cvar = 400;
C. fvar = 200;
cvar = 400;
D. this.fvar = 200;
this.cvar = 400;
E. t.fvar = 200;
Test2.cvar = 400;
F. this.fvar = 200;
Test2.cvar = 400;

answer

Option A, using the reference variable of the object entity to access the field of the object entity, this is correct.
Option B, use static fields directly in the static method "main", which is also correct.

Option C , accessing the field of the object entity directly in the static method "main", because there is no object, it can not be accessed.
Option D, use "this" directly in the static method "main" to indicate the current object entity, because there is no object, so it can not be accessed.
Option E, using the reference variable of the object entity to access the field of the object entity, and using the category name to access the category field, this is also true.
Option F, you cannot use "this" for the same reason as option D.

Question: 75

```
1  public class TestA extends Root {
2
3      public static void main(String[] args) {
4          Root r = new TestA();
5          System.out.println(r.method1()); // line n1
6          System.out.println(r.method2()); // line n2
7      }
8  }
9
10 class Root {
11
12     private static final int MAX = 20000;
13
14     private int method1() {
15         int a = 100 + MAX;  // line n3
16         return a;
17     }
18
19     protected int method2() {
20         int a = 200 + MAX;  // line n4
21         return a;
22     }
23 }
```

Which line causes a compilation error?

A. Line n1
B. Line n2
C. Line n3
D. Line n4

answer

Line n1 will compile errors because the "method1" method is a private member of the Root class and cannot be accessed by outside classes.

Question: 76

Given the following four Java file definitions:

Foo.java

```
1 package facades;
2
3 public interface Foo { }
```

Boo.java

```
1 package facades;
2
3 public interface Boo extends Foo { }
```

Woofy.java

```
1 package org.domain;
2
3 // line n1
4
5 public class Woofy implements Boo, Foo { }
```

Test.java

```
1 package org;
2
3 // line n2
4
5 public class Test {
6     public static void main(String[] args) {
7         Foo obj = new Woofy();
8     }
9 }
```

Which set modifications enable the code to compile and run?

A.
At line n1, Insert:
import facades;
At line n2, insert:
import facades;
import org.domain;

B.
At line n1, Insert:
import facades.*;
At line n2, insert:
import facades.*;
import org.*;

C.
At line n1, Insert:
import facades.*;
At line n2, insert:
import facades.Boo;
import org.*;

D.
At line n1, Insert:
import facades.Foo, Boo;
At line n2, insert:

E.
At line n1, Insert:
import facades.*;
At line n2, insert:
import facades.*;
import org.domain.Woofy;

answer

The import keyword can introduce categories or interfaces, not the introduction package (package), so option A is wrong.
For option B and option C , note that "*" refers to all categories or interfaces under the suite, and does not include sub-kits. In fact, there is no numerator or mother kit, as long as the kit name is different , it is a different kit.
Option D, the syntax is wrong, you can't use "," to introduce different categories.
Option E is correct.

Question: 77

Given the code fragment:

```
9  public static void main(String[] args) {
10    List colors = new ArrayList();
11    colors.add("green");
12    colors.add("red");
13    colors.add("blue");
14    colors.add("yellow");
15    colors.remove(2);
16    colors.add(3, "cyan");
17    System.out.print(colors);
18 }
```

What is the result?
A. [green, red, yellow, cyan]
B. [green, blue, yellow, cyan]
C. [green, red, cyan, yellow]
D. Am IndexOutOfBoundsException is thrown at runtime

answer
Remove method on line 15 of the program, the colors objects in the 'blue' string removed. In this case Colors ArrayList object stored in the elements ["green", "red" , "yellow"].
The add method on line 16 of the program inserts the string " cyan" into the position of index 3, which is the next position of the "yellow" string. In this case Colors ArrayList object stored in the elements ["Green", "Red", "Yellow", " C Yan"].

Question: 78

Given:

```
1  public class Test {
2
3     static boolean bVar;
4
5     public static void main(String[] args) {
6
7        boolean bVar1 = true;
8        int count = 8;
9        do {
```

```
10          System.out.println("Hello Java! " + count);
11          if (count >= 7) {
12             bVar1 = false;
13          }
14       } while (bVar != bVar1 && count > 4);
15       count -= 2;
16    }
17 }
```

What is the result?

A.

```
Hello Java ! 8
Hello Java ! 6
Hello Java ! 4
```

B.

```
Hello Java ! 8
Hello Java ! 6
```

C .

```
Hello Java ! 8
```

D. C ompilation fails

answer

This program is a logical question quite confusing, really the time to write such a program will not write bad c the ODE. First, here you know, the category variable bVar without specifying the value of the initial situation, is false by default, so the first line 14 while according to the conditions, you can know when bVar1 is false when the conditional expression is not established, back The circle will not continue to execute.

Let's look at c ount this variable, c ount in a do-while loop did not do any of the body is altered, so while the condition on line 14 of c ount greater than 4 is established forever.

If line 11 in the loop will execute the first time set up, so will bVar1 set to false, the loop is executed only once over. Finally, " Hello Java ! 8" is only output once.

Question: 79

Given:

```
1  public class Test {
2
3     public static void main(String[] args) {
4        int arr[] = new int[4];
5        arr[0] = 1;
6        arr[1] = 2;
7        arr[2] = 4;
8        arr[3] = 5;
9        int sum = 0;
10       try {
11          for (int pos = 0; pos <= 4; pos++) {
12             sum = sum + arr[pos];
13          }
```

```
14        } catch (Exception e) {
15            System.out.println("Invalid index");
16        }
17        System.out.println(sum);
18    }
19 }
```

What is the result?
A.12
B.

```
Invalid Index
12
```

C .

```
Invalid Index
```

D. Compilation fails
answer
The value of the pos variable of the for loop in line 11 of the program is "0, 1, 2, 3, 4". When the pos variable value is 4, the 12th line will throw ArrayIndexOut because the array index exceeds the array length. OfB oundsEx c eption, this exception will be caught by c at c h on line 14 . Therefore, the program will output "Invalid index" first, and then output the result of 1+2+4+5.

Question: 80

Given the code fragment:

```
1 public class ForTest {
2
3    public static void main(String[] args) {
4        int[] array = {1, 2, 3};
5        for (foo) {
6        }
7    }
8 }
```

Whi c h three c ode fragments, when re pl a c ed indi vi dually for foo, enabling the prog ram to c ompile?
A. int i : array
B. int i = 0; i < 1;
C . ; ;
D. ; i < 1; i++ E. i = 0; i<1;
answer
Option A is the correct usage of Java forea c h.
Option B is also a standard for loop usage, except that because the step field is missing, the value of the i variable is never changed, so it becomes an infinite loop.
Option C is also the correct infinite loop .
Option D, variable i is not announced in advance.
Option E, the reason is the same as option D.

Question: 81

Given:

```
1 public class TestOerator {
2
3    public static void main(String[] args) {
4       int result = 30 - 12 / (2 * 5) + 1;
5       System.out.print("Result = " + result);
6    }
7 }
```

What is the result?

A.Result = 2
B.Result = 3
C .Result = 28
D.Result = 29
E.Result = 30

answer

Java 's formula follows the rules of "first multiply and divide, then add and subtract" and "calculate first in brackets". Therefore, the operation of this problem is as follows:

Result = 30 - 12 / (2 * 5) + 1 = 30 - **12** / **10** + 1 = 30 - **1** + 1 = 30

One more thing to note here is that Java 's integer division operation automatically omits the part of the decimal point.

Question: 82

```
1  class SpecialException extends Exception {
2
3     public SpecialException(String message) {
4        super(message);
5        System.out.println(message);
6     }
7  }
8
9  public class ExceptionTest {
10    public static void main(String[] args) {
11       try {
12          doSomething();
13       } catch (SpecialException e) {
14          System.out.println(e);
15       }
16    }
17    static void doSomething() throws SpecialException {
18       int[] ages = new int[4];
19       ages[4] = 17;
20       doSomethingElse();
21    }
22    static void doSomethingElse() throws SpecialException {
23       throw new SpecialException("Thrown at end of doSomething() method");
24    }
25 }
```

What will be the output?

A. SpecialException: Thrown at end of doSomething() method
B. Error in thread "main" java.lang.ArrayIndexOutOfBoundsError
C.

```
Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: 4
at ExceptionTest.doSomething(ExceptionTest.java:13)
at ExceptionTest.main(ExceptionTest.java:4)
```

D.

```
SpecialException: Thrown at end of doSomething() method
at ExceptionTest.doSomethingElse(ExceptionTest.java:16)
at ExceptionTest.doSomething(ExceptionTest.java:13)
at ExceptionTest.main(ExceptionTest.java:4)
```

answer
The program will throw ArrayIndexOut OfB oundsEx c eption on line 19 , and then continue to throw out from the main method, causing the thread to generate an exception interrupt. Because ArrayIndexOut OfB oundsEx c eption is not a subcategory of S pe cialEx c eption, it cannot be caught by c at c h on line 13 .

Question: 83

Given:

```
1  public class Test1 {
2
3      static void doubling(Integer ref, int pv) {
4          ref = 20;
5          pv = 20;
6      }
7
8      public static void main(String[] args) {
9          Integer iQbj = new Integer(10);
10         int iVar = 10;
11         doubling(iQbj++, iVar++);
12         System.out.println(iQbj + "," + iVar);
13     }
14 }
```

What is the result?
A. 11, 11 B. 10, 10 C . 21, 11 D. 20, 20 E. 11, 12
answer
Since Java is always "pass by value", changes to the parameters ref and pv within the doubling method do not affect the outer variables. This question only needs to notice the 11th line of the program, and iQbj and iV ar are added to 1 action, so the output is:

```
11, 11
```

Question: 84

Given:

```
1  class Star {
2
3      public void doStuff() {
4          System.out.println("Twinkling Star");
```

```
5      }
6   }
7
8   interface Universe {
9
10     public void doStuff();
11  }
12
13  class Sun extends Star implements Universe {
14
15     public void doStuff() {
16        System.out.println("Shining Sun");
17     }
18  }
19
20  public class Bob {
21
22     public static void main(String[] args) {
23        Sun obj2 = new Sun();
24        Star obj3 = obj2;
25        ((Sun) obj3).doStuff();
26        ((Star) obj2).doStuff();
27        ((Universe) obj2).doStuff();
28     }
29  }
```

What is the result?

A.

```
Shining Sun
Shining Sun
Shining Sun
```

B.

```
Shining Sun
Twinkling Star
Shining Sun
```

C . Compilation fails Total

D. A CLASS CAstexCeption thrown IS AT Runtime

answer

Line 23 of the program materializes a Sun object, then transforms it up and calls its do Stuff method. Since the override category has Sun (OVERR IDE) do STuff method, regardless of which patterns are thus turned into a call to go to the second line 15 do STuff method.

Question: 85

```
1   import java.util.ArrayList;
2   import java.util.List;
3
4   public class Whizlabs {
5
6      public static void main(String[] args) {
```

```
7         List<int> list = new ArrayList<>();
8         list.add(21); list.add(13);
9         list.add(30); list.add(11);
10        list.removeIf(e -> e % 2 != 0);
11        System.out.println(list);
12    }
13 }
```

What is the output?
A. [21, 13, 11]
B. [30]
C. []
D. Compilation fails due to error at line 7
E. Compilation tails due to error at line 10

answer
Line 7 program to use Java 's generics (Gen ERI c) function, but the use is not correct, because generics can not be used in the basic data types . If you change "int" to "Integer" here, the program execution result is:

```
[30]
```

Program line 10 is the Java 8's Lambda and C Olle c new properties tion of, removeIf will C Olle c all eligible elements within tion delete objects, is to delete all odd here.
You can refer to this article to learn about Lambda :

Question: 86

Which of the following data types will allow the following code snippet to compile?

```
Java
float i = 4;
float j = 2;
____ z = i + j;
1
2
3
```

```
float i = 4;
float j = 2;
____ z = i + j;
```

Whi c h of the following data ty pe s will allow the following c ode sn ip pe t to c ompile?
A. long **B. double** C . int **D. float** E. byte
answer
A single precision floating point number (float) is a 32-bit floating point number, and a double precision floating point number (double) is a 64-bit floating point number. Since the i-variable and the j-variable are 32-bit single-precision floating-point numbers, their operation results are also 32-bit single-precision floating-point numbers, or 64-bit double-precision floating-point numbers. store. So the data types that z variables can use are float and double.

Question: 87

Given:
Java

```
public class Whizlabs {
   private String name;
   private boolean pass;

   public static void main(String[] args) {
      Whizlabs wb = new Whizlabs();
      System.out.print("name = " + wb.name);
      System.out.print(",pass = " + wb.pass);
   }
}

public class Whizlabs {

   private String name;
   private boolean pass;
   public static void main(String[] args) {
      Whizlabs wb = new Whizlabs();
      System.out.print("name = " + wb.name);
      System.out.print(",pass = " + wb.pass);
   }
}
```

What would be the output, if it is exe c uted as a prog ram ?
A. Name =, pass = B. Name = null, pass = null **C . Name = null, pass = false**
D. Name = null pass = true E. C ompile error.

answer
The Java category or object field will automatically give the initial value according to different data types . The initial values are as follows:
String, Obje c t: null
byte, short, int, long, float, double: 0
c har:\0
boolean: false

Question: 88

Giv en the c lasses:
* AssertionError
* Arithmeti c Ex c eption
* ArrayIndexOut ofB oundsEx c eption
* FileN otF oundEx c eption
* IllegalArgumentEx c eption
* IOError
* IOEx c eption
* NumberFor matE x c eption
* SQL Ex c eption

Whi c h option lists only those c lasses that belong to the unchecked exception c ategory?
A. AssertionError, ArrayIndexOut OfB oundsEx c eption, Arithmeti c Ex c eption
B. AssertionError, IOError, IO Exception
C. Arithmeti c Ex c eption, FileN otF oundEx c eption, NumberFor matE x c eption
D. FileN otF oundEx c eption , IOEx c eption, SQL Ex c eption

E. ArrayIndexOut OfB oundEx c eption, IllegalArgumentEx cEption, FileN otF oundEx c eption

answer
Java divides exceptions into two types, one for checked exceptions and the other for unchecked exceptions .
The exception to the need to check and not to check is that when writing a program, exceptions that need to be checked must use the "try- cache " structure or the "throws" keyword to handle exceptions, otherwise the program will fail to compile successfully without checking. Exceptions can be compiled successfully regardless of whether or not the program is written.
Exceptions that do not need to be checked include RuntimeEx c eption, Error, and all other exceptions (or errors) that inherit them. Common ones are ArrayIndexOut OfB oundsEx c eption and NullPointerEx c eption. The remaining exceptions are to be checked exception, a common IOEx c eption and SQL Ex c eption.
Option A is all an exception that does not need to be checked.
Option B's IOEx c eption is an exception that needs to be checked.
Option C 's FileN otF oundEx c eption is an exception that needs to be checked.
Option D is all an exception that needs to be checked.
The FileN otF oundEx c eption of option E is an exception that needs to be checked.

Question: 89

Given:

Java
```
public class MyClass{
   public static void main(String[] args){
      while(int ii = 0; ii < 2){
         ii++;
         System.out.println("ii = " + ii);
      }
   }
}

public class MyClass{
   public static void main(String[] args){
      while(int ii = 0; ii < 2){
         ii++;
         System.out.println("ii = " + ii);
      }
   }
}
```
What is the result?

A.
Ii = 1
ii = 2
B. C ompilation fails
C. The prog ram prints nothing
D. The prog ram goes into an infin ite loop with no output

E. The prog ram goes to an infin ite loop outputting:
Ii = 1
ii = 1
answer
The use of the while loop of this question is incorrect and will cause compilation errors. If you want to fix it, you can rewrite it as:

Java
```
public class MyClass{
   public static void main(String[] args){
      int ii = 0;
      while(ii < 2){
         ii++;
         System.out.println("ii = " + ii);
      }
   }
}
```

```
public class MyClass{
   public static void main(String[] args){
      int ii = 0;
      while(ii < 2){
         ii++;
         System.out.println("ii = " + ii);
      }
   }
}
```
Or use the for loop to rewrite it as:

Java

```
public class MyClass{
   public static void main(String[] args){
      for(int ii = 0; ii < 2;){
         ii++;
         System.out.println("ii = " + ii);
      }
   }
}
public class MyClass{
   public static void main(String[] args){
      for(int ii = 0; ii < 2;){
         ii++;
         System.out.println("ii = " + ii);
      }
   }
}
```

Question: 90

Given:

Java

```
class Test {
   int sum = 0;
   public void doCheck(int number) {
      if (number % 2 == 0) {
         break;
      } else {
         for (int i = 0; i < number; i++) {
            sum += i;
         }
      }
   }

   public static void main(String[] args) {
      Test obj = new Test();
      System.out.println("Red " + obj.sum);
      obj.doCheck(2);
      System.out.println("Orange " + obj.sum);
      obj.doCheck(3);
      System.out.println("Green " + obj.sum);
   }
}
class Test {
   int sum = 0;
   public void doCheck(int number) {
      if (number % 2 == 0) {
         break;
      } else {
         for (int i = 0; i < number; i++) {
            sum += i;
         }
      }
   }


   public static void main(String[] args) {
      Test obj = new Test();
      System.out.println("Red " + obj.sum);
      obj.doCheck(2);
      System.out.println("Orange " + obj.sum);
      obj.doCheck(3);
      System.out.println("Green " + obj.sum);
   }
}
```

What is the result?

A.	B.	**C .**	D.	E. C ompilation fails
Red 0 Orange 0 Green 3	Red 0 Orange 0 Green 6	**Red 0** **Orange 1**	Green 4	

answer
The "break;" statement can only be used in loops and "swit c h" structures, so this question will compile errors.

Question: 91

Given the fragment:

Java
```
String[][] arra = new String[3][];
arra[0] = new String[]{"rose", "lily"};
arra[1] = new String[]{"apple", "berry", "cherry", "grapes"};
arra[2] = new String[]{"beans", "carrot", "potato"};
// insert code fragment here
String[][] arra = new String[3][];
arra[0] = new String[]{"rose", "lily"};
arra[1] = new String[]{"apple", "berry", "cherry", "grapes"};
arra[2] = new String[]{"beans", "carrot", "potato"};
// insert code fragment here
```
Whi c h c ode fragment when inserted at line '// insert c ode fragment here', enables the c ode to su cc essfully c hange arra elements to up pe r c ase?

A.
```
for(int i = 0; i < arra.length; i++){
   for(int j = 0; j < arra[i].length; j++){
      arra[i][j] = arra[i][j].toUpperCase();
   }
}

for(int i = 0; i < arra.length; i++){
   for(int j = 0; j < arra[i].length; j++){
      arra[i][j] = arra[i][j].toUpperCase();
   }
}
```

B.
Java
```
for(int i = 0;i < 3; i++){
   for(int j=0; j < 4; j++){
      arra[i][j] = arra[i][j].toUpperCase();
   }
}
for(int i = 0;i < 3; i++){
   for(int j=0; j < 4; j++){
      arra[i][j] = arra[i][j].toUpperCase();
   }
}
```
C.
Java
```
for(String a[] : arra[][]){
   for(String x : a[]){
```

```
      x.toUpperCase();
   }
}
for(String a[] : arra[][]){
   for(String x : a[]){
      x.toUpperCase();
   }
}
```

D.

```
Java
for(int i : arra.length){
   for(String x : arra){
      arra[i].toUpperCase();
   }
}
for(int i : arra.length){
   for(String x : arra){
      arra[i].toUpperCase();
   }
}
```

answer
Option A, the range of index values, and the way the array is modified are correct.
Option B, the fixed index range of the inner loop error causes the program to throw ArrayIndexOut OfB oundsEx c eption.
Option C , "toUp pe r C ase" method does not directly modify the contents of the string, but will return the modified string, so this has no effect.
Option D has a bad syntax, and the for loop cannot be used like this.

Question: 92

```
class A {
}
class B {
}
interface X {
}
 interface Y {}
```

Which two definitions of c lass C are valid?
A. class C extends A implements X { }
B. class C implements Y extends B { }
C. class C extends A, B { }
D. class C implements X, Y extends B { }
E. class C extends B implements X, Y { }
answer
Category (c lass) can use "im pl ements" to implement multiple interfaces (interfa c e), but only "extends" can be used to inherit a single category, and "extends" must be used before "im pl ements".

1.

Question: 93

```
abstract class X {

   public abstract void methodX();
}

interface Y {

   public void methodY();
}
```

Which two code fragments are valid?

A.

```
1 class Z extends X implements Y{
2    public void methodZ(){}
3 }
```

B.

```
1 abstract class Z extends X implements Y{
2    public void methodZ(){}
3 }
```

C.

```
class Z extends X implements Y{
   public void methodX(){}
```

```
}
```

D.

```
abstract class Z extends X implements Y{
}
```

E. class Z extends X implements Y{
public void methodY(){}
}

answer

X is the abstract category , Y is the interface, to successfully establish the Z category to inherit X and implement Y. If Z is not an abstract class , you must implement the abstract method of X and the method interface defined by Y, so options A, C , and E will not work. If Z is an abstract class , you can allow methods that do not have a real block of code.

Question: 94

```
public abstract class Shape {

   private int x;
   private int y;

   public abstract void draw();

   public void setAnchor(int x, int y) {
      this.x = x;
      this.x = y;
   }
}
```

Which two classes use the shape class correctly?

A.

```
1 public class Circle implements Shape{
2    private int radius;
3 }
```

B.

```
1 public abstract class Circle extends Shape{
2    private int radius;
3 }
```

C.

```
1 public class Circle extends Shape{
2    private int radius;
3    public void draw();
4 }
```

D.

```
1 public abstract class Circle implements Shape{
2    private int radius;
3    public void draw();
```

```
4 }
```

E.

```
1 public class Circle extends Shape{
2    private int radius;
3    public void draw() { /* code here */ }
4 }
```

F.

```
1 public abstract class Circle implements Shape{
2    private int radius;
3    public void draw() { /* code here */ }
4 }
```

answer

This question is used in the abstract category (abstract class). Abstract methods are not allowed within the abstract category . Since the abstract category is also a category, it cannot be implemented using the "implements" keyword of the implementation interface (interface) , so the options A, D, and F are all wrong. In addition, the abstract class abstract method must be inheriting the abstract category of non- abstract classes to implement, so the option C is wrong.

Question: 95

```
public class Vowel {

   private char var;

   public static void main(String[] args) {
      char var1 = 'a';
      char var2 = var1;
      var2 = 'e';

      Vowel obj1 = new Vowel();
      Vowel obj2 = obj1;
      obj1.var = 'i';
      obj2.var = 'o';
      System.out.println(var1 + ", " + var2);
      System.out.println(obj1.var + ", " + obj2.var);
   }
}
```

A.

```
a,e
i, o
```

B.

```
a,e
o, o
```

C .

```
e,e
I, o
```

D.

```
e,e
o, o
```

answer
Here is the concept that Java is always "pass by value".
Var1 and var2 are basic type variables that store the value of the data directly; var is the object reference variable, the reference to the stored object (Referen ce), similar to the value of the memory address. In the process of passing variables, Java only passes the "value" of the variable.
On lines 6-8, var1 starts storing 'a', then var1 passes 'a' to var2 for storage, and finally var2 stores the value of 'e'. So at this time var1 stores 'a' and var2 stores 'e'.
In lines 10-11, obj1 refers to a new materialized Vowel object, and then the object reference is also stored in obj2. So at this time, the objects pointed to by the two variables obj1 and obj2 are the same one.
Lines 12-13, just modify the var variable of the same Vowel object, and finally the value of the var variable of the Vowel object is 'o'.

Question: 96

```
public static void main(String[] args) {
   int iArray[] = {65, 68, 69};
   iArray[2] = iArray[0];
   iArray[0] = iArray[1];
   iArray[1] = iArray[2];
   for (int element : iArray) {
      System.out.print(element + " ");
   }
}
```

A.
68, 65, 69
B.
68, 65, 65
C .
65, 68, 65
D.
65, 68, 69
E. Compilation fails
answer
After the third line of the program is executed, the values of the array are {65, 68, 65}.
After the fourth line of the program is executed, the values of the array are {68, 68, 65}.
After the fifth line of the program is executed, the values of the array are {68, 65, 65}.

Question: 97

```
public class Test3 {

   public static void main(String[] args) {
      String names[] = new String[3];
      names[0] = "Mary Brown";
      names[1] = "Nancy Red";
```

```
        names[2] = "Jessy Orange";
        try {
            for (String n : names) {
                try {
                    String pwd = n.substring(0, 3) + n.substring(6, 10);
                    System.out.println(pwd);
                } catch (StringIndexOutOfBoundsException sie) {
                    System.out.println("String out of limits");
                }
            }
        } catch (ArrayIndexOutOfBoundsException e) {
            System.out.println("Array out of limits");
        }
    }
}
```

What is the result?

A.

```
Marrown
String out of limits
JesOran
```

B.

```
Marrown
String out of limits
Array out of limits
```

C .

```
Marrown
String out of limits
```

D.

```
Marrown
NanRed
JesOran
```

answer

The length of na mes [0] is 10, the index range is 0~9; the length of na mes [1] is 9, the index range is 0~8; the length of na mes [2] is 12, and the index range is 0~11. . The "StringIndexOut OfB oundsEx c eption" exception is thrown when the string index range is incorrect .

The 11th line of the program uses the substring of the string object's "substring" method to obtain the substring of each string in the na mes string array. The substring has a character index range of 0~2 and 6~9, so na mes [1] with a length less than 10 will throw the exception of "StringIndexOut OfB oundsEx c eption", while other strings can successfully concatenate substrings.

Question: 98

```
int x = 10;
if (x > 10) {
    System.out.println(">");
```

```
} else if (x < 10) {
   System.out.println("<");
} else {
   System.out.println("=");
}
```

Which of the following is equivalent to the above code fragment?

A. System.out.printLn(x > 10 ? ">" : "<" : "=");

B. System.out.println(x > 10 ? ">" ? "<" : "=");

C . System.out.println(x > 10 ? ">" : x < 10 ? "<" : "=");

D. System.out.printLn(x > 10 ? ">" ? "<" ? "=");

E. None of the above

answer

This question is used to test the ternary operator (?:). The ternary operator can replace the if-else structure in some cases and can achieve better performance than if-else. The format of the ternary operator is as follows:

Judgment? The value to be returned when the judgment is established: The value to be returned when the judgment is not established

The "?" and ":" of the ternary operator must appear in pairs, so the options A, B, and D are all wrong, and the conditional logic of the option C matches the conditional program given by the title.

Question: 99

```
public class CharToStr {
   public static void main(String[] args) {
      String str1 = "Java";
      char str2[] = {'J', 'a', 'v', 'a'};
      String str3 = null;
         for (char c : str2) {
            str3 = str3 + c;
         }
         if (str1.equals(str3))
            System.out.print("Successful");
         else
            System.out.print("Unsuccessful");
   }
}
```

What is result?

A.

Successful

B.

Unsuccessful

C . Compilation fails Total

D. An Exception is thrown AT Runtime

answer

Line 3 of str3 is initially null, and then on line 7 directly use the "+" operator to string with the character, this way is allowed in Java. The

reference to null string is automatically changed to a "null" string of length 4 when the string is concatenated, so the str3 variable of this question is concatenated to the final result as "nullJava", "Java" and "nullJava". It is obviously not the same string, so the output is "Unsuccessful".

Question: 100

Given the class definitions:

```
class Alpha {
   public String doStuff(String msg) {
      return msg;
   }
}

class Beta extends Alpha {
   public String doStuff(String msg) {
      return msg.replace('a', 'e');
   }
}

class Gamma extends Beta {
   public String doStuff(String msg) {
      return msg.substring(2);
   }
}
```

And the code fragment of the main() method,

```
List<Alpha> strs = new ArrayList<Alpha>();
strs.add(new Alpha());
strs.add(new Beta());
strs.add(new Gamma());
for (Alpha t : strs) {
   System.out.println(t.doStuff("Java"));
}
```

What is the result?

A.

```
Java
Java
Java
```

B.

```
Java
Jeve
va
```

C .

```
Java
Jeve
ve
```

D. Compilation fails
answer
Since " doStuff" is an object method, overriding (Override)
occurs. Regardless of how the object is transformed upwards, it is necessary to look for the last method of overwriting from the category type of its physical object. In this topic, "Alpha", "Beta", and "Gamma" all have the method of overwriting the " doStuff" object, so their physical objects will use their own " doStuff" object method. Therefore the output is:
Java
Jeve
va

Question: 101

Which two items can legally be contained within a java class declaration?
A. An import statement
B. A field declaration
C . A package declaration
D. A method declaration

answer
Regardless of the annotation, the "package" of option C must be at the top of the ".java" file. The "import" position of option A should be immediately below the "package" in the ".java" file. If there is no "package" "" will be at the top of the ".java" file. The program example is as follows:
Text.java
```
package org.magiclen;

import java.util.Scanner;
import java.util.Base64;

public class Test {
    ...
}
```
Text.java
```
import java.util.Scanner;
import java.util.Base64;

public class Test {
    ...
}
```

The field pointed to by option B represents all the variables and constants. There is no need to add parentheses after the "." during access. The method of option D refers to the need to add parentheses after the ".". The program example is as follows:
```
public class Plane {

    private final int width = 10, height = 8;
```

```
   private static int computePlaneArea() {
      Plane plane = new Plane();
      return plane.width * plane.height;
   }

   public static void main(String[] args) {
      System.out.println(computePlaneArea());
   }
}
```

Question: 102

Given:

```
public class Calculator {
   public static void main(String[] args) {
      int num = 5;
      int sum;

      do {
         sum += num;
      } while ((num--) > 1);
      System.out.println("The sum is " + sum + ".");
   }
}
```

What is the result?
A. The sum is 2
B. The sum is 14
C. The sum is 15
D. The loop exe c utes **infin** ite ti mes
E. C ompilation fails
answer
Line 4 declares a sum integer variable, but it is not initialized. Line 7 "sum += num;" can be broken down into:

7 sum = sum + num;

Since the sum variable is not initialized, it is taken as an addition, so a compile error occurs.

Question: 103

Given:

```
class Sports {

   int num_players;
   String name, ground_condition;

   Sports(int np, String sname, String sground) {
```

```
        num_players = np;
        name = sname;
        ground_condition = sground;
    }
}

class Cricket extends Sports {

    int num_umpires;
    int num_substitutes;

    //insert code here
}
```

Which code fragment can be inserted at line "//insert code here" to enable the code to compile?

A.

```
Cricket() {
    super(11, "Cricket", "Condidtion OK");
    num_umpires = 3;
    num_substitutes= 2;
}
```

B.

```
Cricket() {
    super.ground_condition = "Condition OK";
    super.name= "Cricket";
    super.num_players = 11;
    num_umpires = 3;
    num_substitutes= 2;
}
```

C.

```
Cricket() {
    this(3,2);
    super(11, "Cricket", "Condidtion OK");
}
Cricket(int nu, int ns) {
    this.num_umpires = nu;
    this.num_substitutes= ns;
}
```

D.

answer

This question is the concept of constructing a constructor. Each category has at least one constructor. If the category does not have a constructor, it will be populated with a constructor subroutine at compile time. Take the category "Cricket" as an example. There is no implementation constructor at first, so the compiler will automatically add the following program:

1 public Cricket(){}

In addition, all constructors will first call the constructor of the parent class or other constructors of their own class. If the constructor does not specify which constructor to call first, then the compiler will always add "super();" in the first itinerary of the constructor. Take the category "Cricket" as an example. After adjusting the constructor, it will look like this:

```
class Cricket extends Sports {

   int num_umpires;
   int num_substitutes;

   public Cricket(){
      super();
   }
}
```

Since the parent class "Sports" of Cricket does not have the constructor "Sports()" , the original program will fail to compile. So this topic requires the correct constructor for the C ri c ket category , so that the program can compile successfully.

Option A adds "super(11, " Cricket", " Condidtion OK");" in the first line of the new constructor . The call originally exists in the Sports category of "Sports(int np , String sname, String sground)" constructor , so adding this program can make the program compile successfully. The new constructor of option B is not described in the first line using "super" or "this". The compiler will automatically add "super();", but the Sports category does not have "Sports()". This constructor will therefore fail to compile. In the new constructor of option C , "su pe r(11, " Cricket", " Condidtion OK");" is added, but it is placed on the second line, so a compilation error occurs. To call the constructor of the parent category with "super" , you can only construct the constructor.

The first line is used.

The reason for option D compilation error is the same as option C.

Question: 104

Given:

```
public class TestTry {

   public static void main(String[] args) {
      StringBuilder message = new StringBuilder("hello java!");
      int pos = 0;
      try {
         for (pos = 0; pos < 12; pos++) {
            switch (message.charAt(pos)) {
               case 'a':
               case 'e':
               case 'o':
```

```
                String uc =
Character.toString(message.charAt(pos)).toUpperCase();
                message.replace(pos, pos + 1, uc);
            }
          }
      } catch (Exception e) {
          System.out.println("Out of limits");
      }
      System.out.println(message);
   }
}
```

What is the result?
A. hEllO jAvA !
B. Hello java !
C .
Out of limits
hEllO jAvA !
D. Out of limits

answer
The string on line 4 is " hello java !" with a total of 11 characters. The length of the string is 11.
For the for loop starting at line 7 , the value of the pos variable is 0, 1, 2, 3, ..., 11, and the loop will be executed 12 times.
The switch starting at the 8th line will convert the characters "a", "e", and "o" in the corresponding position in the string to uppercase each time the loop is executed. However, when the pos variable is 11, the 8th line will throw an exception because the position of the character index to be obtained exceeds the length of the string itself. The exception will start at the 16th line and output "Out of limits". "."
Finally, line 19 will output the result of converting all the "a", "e", and "o" characters in " hello java !" to uppercase.

Question: 105

Which of the following will print current time?
A. System.out.print(new LocalTime().now());
B. System.out.print(new LocalTime());

C. System.out.print(LocalTime.now());
D. System.out.print(LocalTime.today());
E. None of the above.
answer
This question is only used to test the usage of the LocalTime category. Refer to this explanation:

Question: 106

```
public class Test {

   public static void main(String[] args) {
```

```
        int numbers[];
        numbers = new int[2];
        numbers[0] = 10;
        numbers[1] = 20;

        numbers = new int[4];
        numbers[2] = 30;
        numbers[3] = 40;
        for(int x : numbers){
           System.out.print(" " + x);
        }
    }
}
```

What is the result?

A. 10 20 30 40

B. 0 0 30 40

C . Compilation fails Total

D. An EXCeption thrown IS AT Runtime

answer

In line 9 of the program, a new array of integer lengths of 4 is assigned to the numbers variable, so the integer array of length 2 before this is not pointed by any variables (this array space can be called an island), which is occupied by memory will be automatically wait Java garbage collection (garbage C Olle c tion) mechanism to recover for reuse.

The value in the numerical array starts with an initial value of zero. The program only changed the values of number array object indexes 2 and 3 on lines 10-11. Therefore, the final output is "0 0 30 40".