After invoking following code, value of a, c are ?

```
int x = 182;
int a, c;
c = x / 100;
a = x % 10;
```

c=1
a=2

Which expression can get max value from x & y

a) x > y ? y : x; b) x < y ? y : x; c) x > y ? x + y : x - y; d) x = = y ? y : x;

boolean ? true : false.

What is the output after running the code?

```
StringBuffer buffer = new StringBuffer("hello2024");
buffer.insert(7, "@");
System.out.println(buffer.toString());
```

hello20@24

```
public class Main {
    public static void main(String[] args) {
                //                          012345678
        StringBuffer buffer = new StringBuffer("hello2024");
        buffer.insert( offset: 7,  str: "@");
        System.out.println(buffer);
        //                  012345
        String string = "sdasda";
        String sub = string.substring(1, 5);
        System.out.println(sub);
    }
}
```

⇒ dasd

Read code, when method run, the correct output is ?

```
public class Point {
    public void method() {
        int x = 6, y = 8;
        System.out.print("(" + x + "," + y + ")");
        switchCoords(x, y);
        System.out.print("(" + x + "," + y + ")");
    }

    public void switchCoords(int x, int y) {
        int temp;
        temp = x;
        x = y;
        y = temp;
        System.out.print("(" + x + "," + y + ")");
    }

}
```

⇒ (6,8) (6,8)

call by value: primitive type

⇒ object: call by reference → this will change.

```
Main.java   MyObject.java
public class Main {
    public static void main(String[] args) {

        MyObject myObject = new MyObject( age: 20,  name: "ABC");
        System.out.println("Before method running: " + myObject);
        method(myObject); // this will change value.
        System.out.println("After method running: " + myObject);
    }

    public static void method(MyObject myObject) {
        myObject.setName("Mr/Miss " + myObject.getName());
    }
}
```

```
Before method running: MyObject{age=20, name='ABC'}
After method running: MyObject{age=20, name='Mr/Miss ABC'}
```

create a new class

```
public class MyObject {
    private int age;
    private String name;

    public MyObject(int age, String name) {
        this.age = age;
        this.name = name;
    }

    public int getAge() {
        return age;
    }

    public void setAge(int age) {
        this.age = age;
    }

    public String getName() {
        return name;
    }
}
```

Read code, when method run, the correct output is ?

```
int[] arr = {122, 33, 55, 678, -987};
int max = arr[0];
for (int i = 1; i <= arr.length; i++) {
    if (arr[i] > max) {
        max = arr[i];
    }
}
System.out.println(max);
```

arr.length = 5

1, 2, 3, 4, 5

run time exception index out of bound.

use for each loop. this case won't happen.

What is the value of k after code run ?

```
int x = 2, y = 5, k = 0;
switch(x % y) {
    case 0 -> k = x+y;
    case 1 -> k = x -y;
    case 2 -> k = x * y;
    default -> k = x / y;
}
```

2 % 5 ⇒ 2

2 % -5 ⇒ 2

k = 0

What is the value of k after code run ?

```
int x = 2, y = 5, k = 0;
switch(x % y) {
    case 0 : k = x+y;
    case 1 : k = x -y;
    case 2 : k = x * y;
    default : k = x / y;
}
```

⇒ This one doesn't have break.

k = 0

3 % 5 = 3

3 % -5 = 3

-3 % -5 = -3

-3 % 5 = -3

Read code, when method run, the correct output is ?

```
public class MyClass {
    static int x = 10;
    public static void main(String[] args) {
        for (int n = 3; n > 0; n- -) {
            x *= x;
        }
        System.out.println(x);
    }
}
```

not static variable cannot directly use in static method

If I want to access the non-static variable. we need to create an object first.

```
public class Main {
    private int x = 10;
    public static void main(String[] args) {
        Main main = new Main();
        for (int n=3; n > 0; n--) {
            main.x *= main.x;
        }
        System.out.println(main.x);
    }
}
```

Read code, how many times would statement a = a + 1; execute ?

```
public class Test {
    public static void main(String[] args) {
        int x = 8, a = 1;
        do {
            a = a + 1;
        } while (x > 0);
    }
}
```

unlimit - infinite

How many objects are created when following code run ?

```
int[] a = new int[10];
String b = "abc";
String c = new String("abc");
MyTest test = new MyTest();
String[] words = new String[10];
```

one object

5

Find errors from code below;

```
Int count;   //1
int x[] = new int[10]; //2
for (int i = 0; i < 5; i++) {  //3
    count++;   //4
}
System.out.print(x[5]);  //5
```

Integer or int

this one is also correct but prefer to write in this format.

int[] x = new int[10]

=) 0

```
public class Main {
    private int x = 10;
    public static void main(String[] args) {
        int number = 123;
        System.out.println((number + "").substring(1, 2));

        char character = (number + "").charAt(1);
        System.out.println(character);
    }
```

to change number from int to string

String.valueof(number)

↳ can also change int to string

Write the expression to get the middle digit of a 3-digits integer;

those are int

Suppose x = 1, y = 2, z = 3; What is the value of y —> y += z - - / ++x ;

y = y + z-- / ++x

3 = 2 + 3/2

After running following code, values of a, b c are ...?

```
int a = 6, b = 8;
boolean c;
c = a > b && ++a == - -b;
```

c = false

a = 6

b = 8

```
int x = 1, y = 2, z = 3;
  y += z-- / ++x;
  System.out.println(y);

x *= x-- + ++x / y++ * --z - ++z / x-- + z++;
```

x = x * (x-- + ++x / y++ * --z - ++z / x-- + z++);

1 * (1 + 1/2 * 2 - 3/1 + 3)

x = 1

What are the values of a b c d, after running following code;

```
int a = 100;
int b = 50;
int c = a -- - b;
int d = a - --b;
```

a=99 b=49 c=50 d=50

Use the ternary operator to find the largest of any three numbers.

```
int a, b, c
temp = a>b ? a : b
max = temp>c ? temp : c
```

⇒ `int maxNum = a>b ? (a>c ? a : c) : (b>c ? b : c);`

⇒ `int maxNum = a>b ? Math.max(a,c) : Math.max(b,c);`

Write code to get the union & intersection of 2 strings “Hello” & “Java”

```
String a = "Hello";
String b = "Java";

String ab = a + b;
ArrayList<String> union = new ArrayList<>();

for (int index = 0; index < ab.length(); index++) {
    if (!union.contains(String.valueOf(ab.charAt(index)))) {
        union.add(String.valueOf(ab.charAt(index)));
    }
}
```

```
ArrayList<String> intersection = new ArrayList<>();

for (int i = 0; i < a.length(); i++) {
    if (b.contains(String.valueOf(a.charAt(i)))
            && !intersection.contains(String.valueOf(a.charAt(i)))) {
        intersection.add(String.valueOf(a.charAt(i)));
    }
}
```

What is the output after running following code?

```
int[] arr = { 1, 0, 3, 0, 0, 5};
int sum = 0;
int i = 0;
for (int i = 0; i < arr.length; i++) {
    if (arr[i] == 0) {
        sum -= 1;
    } else {
        sum += arr[i];
    }
}
System.out.println(sum);
```

syntax error

⇒ 6

What is the output after running following code?

```
int i = 0;
try{
    i++;
    throw new IOException();
} catch (IOException e) {
    i++;
} catch (Exception e) {
    if (i != 0) {
        i++;
    }
} finally {
    i++;
}
i++;
System.out.print(i);
```

Be careful the order of Exception.

⇒ 4

Read code and answer questions?

```
ArrayList<Integer> answers = new ArrayList<>();
ArrayList<Integer> temp = new ArrayList<>();
answers.add(3);
answers.add(2);
answers.add(-5);
answers.add(2, 4);
int a = answers.get(2);
int b = answers.get(3);
answers.remove(2);
answers.remove(1);
int c = answers.lastIndexOf(2);
boolean d = answers.contains(3);
temp.add(3);
temp.add(-5);
boolean e = answers==temp;
```

Which one is true?

(1) a == 4 (2) b == -5 (3) c == -1 (4) d == true (5) e == true

index

index

value

3, 2, -5

3, 2, 4, -5

a=4

b=-5

3, -5

c=-1 ← value not found, will show -1

d=true

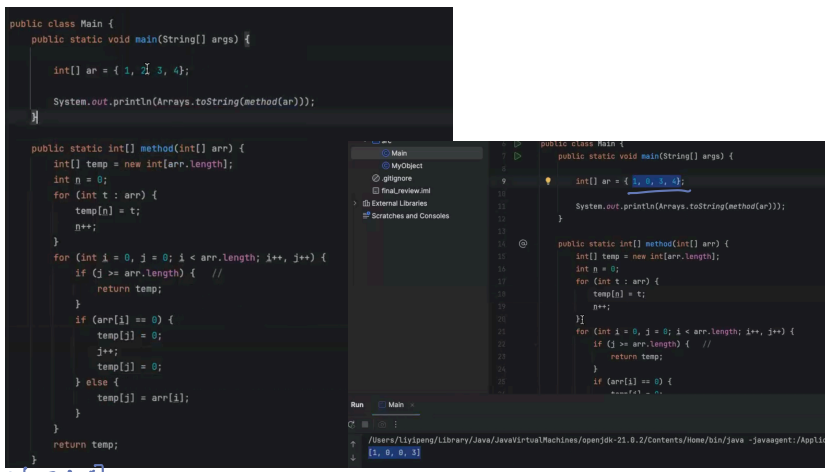
e=false ⇒ answers and temps are different object. (different memory location)

```
ArrayList<Integer> answers = new ArrayList<>();
ArrayList<Integer> temp = answers;//new ArrayList<>();
```

↳ In this case, e will be true. Because temp and answers has same memory location.

Read code answer question:

```
public static int[] method(int[] arr) {
    int[] temp = new int[arr.length];
    int n = 0;
    for (int t : arr) {
        temp[n] = t;

        n++;

    }

    for (int i = 0, j = 0; i < arr.length; i++, j++) {
        if (j >= arr.length) {    //
            return temp;
        }
        if (arr[i] == 0) {
            temp[j] = 0;
            j++;
            temp[j] = 0;
        } else {
            temp[j] = arr[i];
        }

        }
}
```



⇒ [1, 2, 3, 4]

What is the output for following program:

```
class Base {
    public void show() {
        System.out.println("Base::show() called");
    }
}

class Derived extends Base {
    public void show() {
        System.out.println("Derived::show() called");
    }
}

public class Main {
    public static void main(String[] args) {
        Base b = new Derived();
        b.show();
    }
}
```

由子類創建出來的

⇒ Derived::show() called

What is the output for following program:

```
class Base {
    final public void show() {
        System.out.println("Base::show() called");
    }
}

class Derived extends Base {
    public void show() {
        System.out.println("Derived::show() called");
    }
}

class Main {
    public static void main(String[] args) {
        Base b = new Derived();;
        b.show();
    }
}
```

What is the output for following program:

```
public class Test
{
  public int getData()
  {
    return 0;
  }
  public long getData()
  {
    return 1;
  }
  public static void main(String[] args)
  {
    Test obj = new Test();
    System.out.println(obj.getData());
  }
}
```

not allow to do this overload

method name is equal but parameters are not the same

What is the output for following program:

```
public class Test
{
  private String function(String temp, int data)
  {
    return ("1");
  }
  private String function(int data, String temp)
  {
    return ("2");
  }
  public static void main(String[] args)
  {
    Test obj = new Test();
    System.out.println(obj.function(4, "HY"));
  }
}
```

=> overload.

What is the output?

```
int j = -1;
if (j > 0 || 1 / (j + 1) > 10) {
  System.out.println("True branch");
} else {
  System.out.println("False branch");
}
```

Error ArithmeticException

What is the output?

```
int[] arr = {1, 8, 4, 11, 7, 14, 10};
int x = arr[6];
for (int i = 0; i < arr.length && arr[i] <= x; i++) {
  if (arr[i] % 2 == 0) {
    System.out.print(i + " ");
  }
}
```

=> 1 2

What is the output?

```
boolean[] arr = new boolean[4];
arr[0] = true;
arr[1] = arr.length > 4 ? true : false;
System.out.printf("%b %b %b", arr[0], arr[1], arr[3]);
```

truefalsefalse

What is the output?

```
int arr[] = { 1, 2, 3, 4, 5};
arr[4] = arr[4] * 2;
int total = 0;
for (int value: arr) {
  total += value;
}
System.out.println(total);
```

1+2+3+4+10 = 20

720

What is the output?

```
public class TestClass {
  int x;
  public static void main(String[] args) {
    TestClass object;
    System.out.println(object.x);
  }
}
```

Syntax Error

Select all statements that are true regarding abstract classes.

1. Subclasses can access their private methods by name; X
2. They can have constructors; √
3. Subclasses inherit their private variables and methods; X
4. They can be instantiated; X

無法創建對象

Select all statements that are true regarding interfaces.

1. Interfaces add flexibility and re-usability for software components. √
2. An interface is a reference type in Java. X interface is not an object
3. Interfaces are classes. X
4. Methods in an interface are public implicitly. X

public or none

In the code below, the Triangle class is a child class of the Shape class. The Shape class declares a non-static method draw() which will draw a square to screen. The Triangle class also declares a non-static method draw() which will draw a triangle onto screen. Pick the statement that best applies from the list.

```
public class TestClass {
  public static void main(String[] args) {
    ArrayList<Shape> shapes = new ArrayList<Shape>();
    shapes.add(new Shape());
    shapes.add(new Triangle());
    for (Shape shape: shapes) {
      shape.draw();
    }
  }
}
```

1. The code will draw 2 squares onto screen.
2. √ The code will draw a square and then a triangle onto screen.
3. The code will not compile because ArrayList only takes Shape objects, but we are trying to a Triangle object to it as well.
4. The code will draw 2 triangles onto screen.

The following piece of code is stored in a file named TestException.java. What is the output of this program if it is run by typing the following command: javac TestException.java; java TestException

```
public class TestException {
  public static void main(String[] args) {
    while (true) {
      try{
        throw new Exception();
      } catch (Exception e) {
        System.out.println("An exception is caught"); break;
      } finally {
        System.out.println("Inside the finally block");
      }
    }
    System.out.println("Before exiting");
  }
}
```

1. Before exiting
2. An exception is caught Before exiting
3. An exception is caught Inside the finally block
4. An exception is caught Inside the finally block Before exiting

The following piece of code is stored in a file named TestException.java. What is the output of this program if it is run by typing the following command: javac TestException.java; java TestException

```
public class TestException {
  public static void main(String[] args) {
    while (true) {
      try{
        // nothing wrong here
      } catch (Exception e) {
        System.out.println("An exception is caught"); break;
      } finally {
        System.out.println("Inside the finally block");
      }
    }
    System.out.println("Before exiting");
  }
}
```

1. The program run into a infinite loop with no message
2. The message "Inside the finally block" is printed in an infinite loop
3. Before exiting
4. An exception is caught Inside the finally block Before exiting

Consider the code that uses inheritance below, and select the most correct statement.

```
public class Shape {}

public class Rectangle extends Shape {}

public class Square extends Rectangle {}
```

A Rectangle is a Square and also a Shape.
A Square is a Rectangle but not a Shape.
Such inheritance is not allowed in Java.
A Rectangle is a Shape but not a Square.

Given class Engine {

```
    private boolean isInitialized = true;
    public static boolean isInitialized() {
        return this.isInitialized;
    }
}
```

could not run the non-static variable in static method

this is to object

1. The code will compile and run without a problem, but an object of the class must be created before the isInitialized method is called.
2. The code will not compile because isInitialized is a static method, and it cannot access a non-static instance variable.
3. The code will compile and run without a problem, when the isInitialized method is called true will be returned.
4. The code will compile and run without a problem, when the isInitialized method is called false will be returned.

What is the most probable cause of a NullPointerException?

1. using an object that has not been instantiated.
2. trying an illegal type cast.
3. referencing a file that cannot be located. FileNotFoundException
4. exhausting all available memory. FileNotFoundException

What is the output ? primitive default. 0.0

```
double[] arr = new double[4];
arr[0] = 1.0;
arr[1] = 4.0;
System.out.println(arr[0] + arr[1] + arr[2]);
```

=> 5.0

What is the output ?

```
public class Car {
   private String make;
   private int year;
   public Car() {
      make = "Fiat";
      year = 2021;
   }
   public Car(String make, int year) {
      this.make = make;
      this.year = year;
   }
   public String toString() {
      return make + " " + year;
   }
   public static void main(String[] args) {
      Car aCar = new Car("Kia", 2024);
      System.out.println(aCar);
   }
}
```

=> Kia 2024

What is the output ?

```
public class Car {
   private String make;
   private int year;
   public Car() {
      make = "Fiat";
      year = 2021;
   }
   public Car(String make, int year) {
      make = make;
      year = year;
   }
   public String toString() {
      return make + " " + year;
   }
   public static void main(String[] args) {
      Car aCar = new Car("Kia", 2024);
      System.out.println(aCar);
   }
}
```

can not define parameter name is initial name.

=> Fiat 2021

Given the following definition of class Base & Derived. What is the output of following statements?

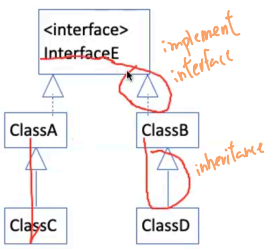
```
public class Base {
   public void method1() {
      System.out.println("A");
      method2();
   }
   public void method2() {
      System.out.print("B");
   }
}
public class Derived extends Base {
   public void method1() {
      super.method1();
      System.out.print("C");
   }
   public void method2() {
      super.method2();
      System.out.print("D");
   }
}
```

```
Base object = new Derived();
object.method1();
```

b.method2();

A B D C

Given the UML diagram below. choose all correct implementations.



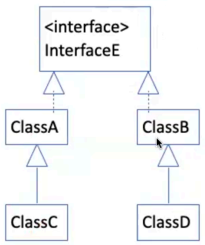
1) .public interface InterfaceE {...}
public class ClassA implements InterfaceE {...}
public class ClassB implements InterfaceE {...}
public class ClassC extends ClassA {...}
public class ClassD extends ClassB {...}

2) .public interface InterfaceE {...}
public class ClassA extends InterfaceE {...}
public class ClassB extends InterfaceE {...}
public class ClassC implements ClassA {...}
public class ClassD implements ClassB {...}

3) .public interface InterfaceE {...}
public class ClassA extends InterfaceE {...}
public class ClassB extends InterfaceE {...}
public class ClassC extends ClassA {...}
public class ClassD extends ClassB {...}

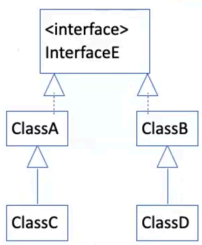
4) .public interface InterfaceE {...}
public class ClassA implements InterfaceE {...}
public class ClassB implements InterfaceE {...}
public class ClassC implements ClassA {...}
public class ClassD implements ClassB {...}

Given the UML diagram below. choose all correct statements.



1. ClassC extends ClassA and ClassD extends ClassB
2. Both ClassA and ClassB implements InterfaceE
3. ClassC implements ClassA & ClassD implements ClassB
4. Both ClassA and ClassB extends InterfaceE

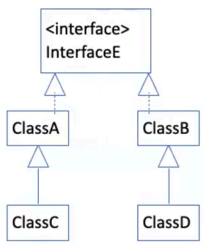
Given the UML diagram below. choose all expressions that evaluate 'true'.




```
ClassA objA = new ClassA();
ClassA objA2 = objA;
ClassA objA3 = new ClassA();
ClassC objC = new ClassC();
ClassD objD = new ClassD();
```

1. objC instanceof InterfaceE
2. objD instanceof ClassB
3. objA2 == objA
4. objC instanceof ClassA
5. objA == objA3

Given the UML diagram below. choose all code blocks that will introduce compilation errors.




```
1) .ArrayList<InterfaceE> aList = new ArrayList<InterfaceE>();
aList.add(new ClassA());
aList.add(new ClassB());
aList.add(new ClassC());
aList.add(new ClassD());

2) .ArrayList<ClassA> aList = new ArrayList<>();
aList.add(new ClassA());
aList.add(new ClassB());
aList.add(new ClassC());
aList.add(new ClassD());

3) .ArrayList<InterfaceE> aList = new ArrayList<InterfaceE>();
aList.add(new ClassA());
aList.add(new ClassA());
aList.add(new ClassB());
aList.add(new ClassB());

4) .ArrayList<ClassC> aList = new ArrayList<>();
aList.add(new ClassA());
aList.add(new ClassC());
```

Choose all unhandled exceptions that could cause the following program to terminate.

```
public class TestFileReadingExceptions {
  public static void main(String[] args) {
    Scanner inputStream = null;
    try{
      inputStream = new Scanner(new FileInputStream(args[1]));
    } catch (FileNotFoundException e) {
      System.out.println("File not found");
    }
    while (inputStream.hasNextLine()){
      String line = inputStream.nextLine();
      int number = Integer.parseInt(line);
      System.out.println(number);
    }
    inputStream.close();
  }
}
```

1. ArrayIndexOutOfBoundsException 2. NumberFormatException 3. FileNotFoundException
4. NullPointerException 5. ClassCastException 轉換類型

```java
public class TestClass {
  private String str = "a";
  public void methodA() {
    try{
      str += "b";
      methodB();
    } catch (Exception e) {
      str += "c";
    }
  }
  public void methodB() throws Exception {
    try{
      str += "d";
      methodC();
    } catch (Exception e) {
      throw new Exception();
    } finally {
      str += "e";
    }
    str += "f";
  }
```

ab

abdefc

abd

abd

abdef

```java
  public void methodC() throws Exception {
      throw new Exception();
  }
  public void display() {
    System.out.println(str);
  }

  public static void main(String[] args) {
    TestClass aTest = new TestClass();
    aTest.methodA();
    aTest.display();
  }
}
```

=> abdefc

Which snippet you should insert into the header of the inner loop to print following pattern?

```
*
***
*****
*******
*********
```

```java
for (int i = 0; i < 5; i++) {
      for (<insert code here>) {
            System.out.print("*");
      }
      System.out.println();
}
```

X `int j = 0; j >= i; j++`

✓ `int j = 0; j <= i * 2; j++`

X `int j = 0; j <i * 2; j--`

X `int j = 0; j >= i; j--`

which expression evaluates to 'false'

8/4 == 2

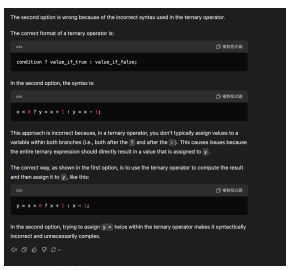
8 > 4 || 8 < 5

8 % 10 == 0

5 % 2 == 1

Statement if (x < 0) y = x + 1; else y = x - 1; can be replaced by ...?

1. y = x < 0 ? x + 1 : x - 1;
2. x < 0 ? y = x + 1 : y = x - 1;
3. y = if (x < 0) x : 0
4. can not rewrite this statement.



What does 'overriding a method ' mean?

1. A never-ending loop within a method.
2. Different methods have the same name, but different signatures overloading
3. The definition of an inherited method can be changed in the definition of a derived class so that it has a meaning in the derived class that is different from what it is in the base class.
4. The number of a parameters exceeds what is needed in the function body

```java
public static void main(String[] args) {
  int x = 20;
  int count = 0;
  do {
    x = x - 2;
    count = count + 1;
  } while (x > 0);
}
```

What is the value of count?

1. 10
2. 80
3. 20
4. 40

```java
public static void main(String[] args) {
  int x = 20;
  int count = 0;
  do {
    x = x - 2;
    for (int i = 0; i < x % 2; i++) {
        count = count + 1;
    }
  } while (x > 0);
}
```

What is the value of count?

1. 20
2. 0
3. 10
4. 5

What is the output?

```
public class TestDemo {
    public static void fuc1(int[] array){
        array = new int[]{1,2,3};
    }
    public static void fuc2(int[] array){
        array[0]=99;
    }
    public static void main(String[] args) {
        int[] array={5,6,7};
        fuc1(array);
        for (int i = 0; i < array.length; i++) {
            System.out.print(array[i]+" ");
        }
        System.out.println();
        fuc2(array);
        for (int i = 0; i < array.length; i++) {
            System.out.print(array[i]+" ");
        }
        System.out.println();
    }
}
```

⇒ 1. 2. 3
99. 6. 7

What would be the output of the code snippet given below when the value of integer **'i'** is "**-3**"?

```
return i % 2 == 1;
```

⇒ false

This program below increments a variable **j** repeatedly and then prints its value. What will be the output?

```
public static void main(Strin
g[] args){
    int j = 0;
    for(int i = 0; i < 100; i
++)
        j = j++; // post-operation
    System.out.println(j);
}
```

0

In the poor little program given below, the decision method returns both **true** and **false**. In not more than 2 lines write what is the **final output** that is printed and **why**?

```
public static void main(Strin
g[] args){
    System.out.println(decisi
on());
}

static boolean decision(){
    try {
        return true;
    }finally{
        return false;
    }
}
```

finally - 一定要執行

⇒ false

What will be the **output** of the following program:

```
public static void main(Strin
g[] args){
    try{
        System.out.println("H
ello world");
        System.exit(0);
    }finally{
        System.out.println("G
oodbye world");
    }
}
```

=> Hello world

```
public class Confusing{
    private Confusing(Object
o){
        System.out.println("O
bject");
    }

    private Confusing(double
[] dArray){
        System.out.println("d
ouble array");
    }

    public static void main(S
tring[] args){
        new Confusing(null);
    }
}
```

too general

need to find the more specific one

What is the **output** of the Confusing class?

=> double array

What does the following program print?

```
class Dog {
    public static void bark()
{
        System.out.print("woo
f ");
    }
}

class BullDog extends Dog{
    public static void bark()
{}
}

public class Bark {
    public static void main(S
tring args[]) {
        Dog labrador = new Do
g();
        Dog frenchBD = new Bu
llDog();
        labrador.bark();
        frenchBD.bark();
    }
}
```

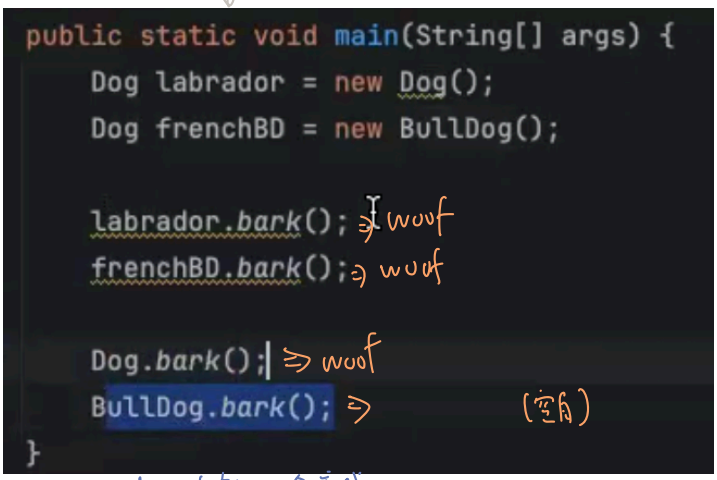
What is the most probable cause of NullPointerException?

- Exhausting all available memory
- Referencing a file that cannot be located
- Trying an illegal type cast
- Using an object that has not been instantiated

Consider the code that uses **inheritance** below and **select** the most correct statement.

```
public class Shape{
//...
}

public class Rectangle extend
s Shape{
//...
}

public class Square extends R
ectangle{
//...
}
```

- A Square is a Rectangle but not a Shape.
- A Rectangle is a Square and also a Shape.
- A Rectangle is a Shape but not a Square.
- Such inheritance is not allowed in Java.



如果去掉 static 會發生

=> 0 woof (空白)

```
public class Car{
    private String make;
    private int year;

    public Car(){
        make = "FIAT";
        year =  2021;
    }

    public Car(String make, i
nt year){
        this.make = make;
        this.year =  year;
    }

    @Override
    public String toString(){
        return (make+ " " + y
ear);
    }

    public static void main(S
tring args[]){
        Car aCar = new Car("K
ia", 2020);
        System.out.println(aC
ar);
    }
}
```

- Fiat 2021
- Compilation Error
- Fiat 2020
- Kia 2020

The following piece of code is stored in a file named `MyClass.java`. What is the output of this program if it is run by typing the following command?

```
javac Car.java; java Car
```

```
class MyClass{
    public static void main(S
tring[] args){
        boolean arr[] = new b
oolean[4];
        arr[0] =  true;
        arr[1] = arr.length >
4 ? true : false;
        System.out.printf("%b
%b %b", arr[0],arr[1],arr
[3]);
    }
}
```

- true false false
- true false true
- false true true
- true true false

Given the following piece of code:

```
public interface MyInterface{
    public MyInterface(){
        System.out.println("T
his is a constructor");
    }
    void myMethod();
}
```

syntax error

What will be the output of it?

No. the method in the interface can only have header.

Which **line** will cause an error in the code snippet given below?

```
1. interface MyInterface{
2.     final int k = 8;
3.     private int m;
4.     public void aMethod();
5.     String fun(char s);
6. }
```

✓ ✗ ✓ ✓

first return 的 method 也行

```
interface InterfaceA{
    void go();
}

class ClassB {
    public void go(){
        System.out.println("G
O!");
    }
}

class ClassC extends ClassB i
mplements InterfaceA{}

class Main{
    public static void main(S
tring[] args){
        InterfaceA a = new Cl
assC();
        a.go();
    }
}
```

What would be the **output**?

⇒ GO

What will be the **output** of the following code snippet?

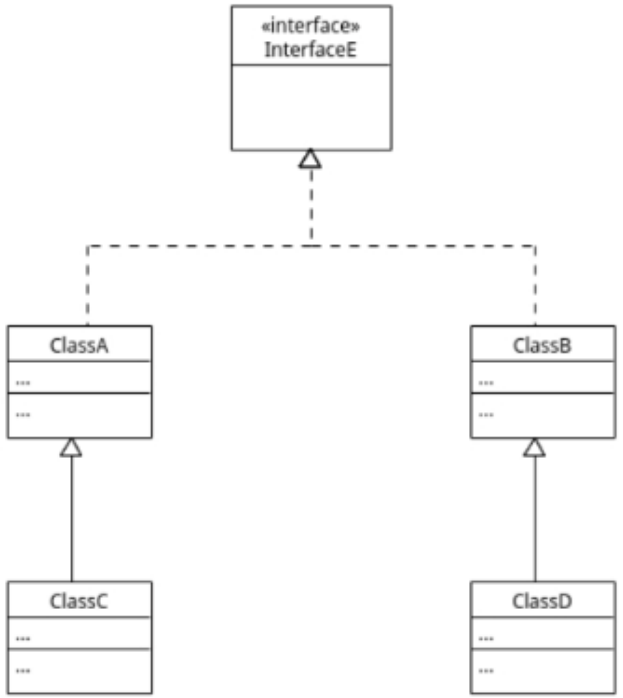
```
List<Object> object1 = new Ar
rayList<Long>();
object1.add("I don't fit i
n");
```

syntax error

Given the UML diagram below and the code lines provided, select **ALL** expressions that evaluate as **'true'**.

```
ClassA objectA = new ClassA
();
ClassA objectA2 = objectA;
ClassA objectA3 = new ClassA
();
ClassC objectC = new ClassC
();
ClassD objectD = new ClassD
();
```

- [x] objectC instanceof InterfaceE
- [x] objectC instanceof ClassA
- [x] objectA2 == objectA
- [ ] objectA == objectA3



What will be the output of the following piece of code when it is called from a main method?

```
public class MyException throws Exception{
    public MyException(){
        super("I am an exception");
    }
}
```

extends.

```
int j = -1;
if((j >0) || (1/(j+1) > 10)){
    System.out.println("True branch. j = "+j);
}else{
    System.out.println("False branch. j = "+j);
```

runtime error

What would be the output?

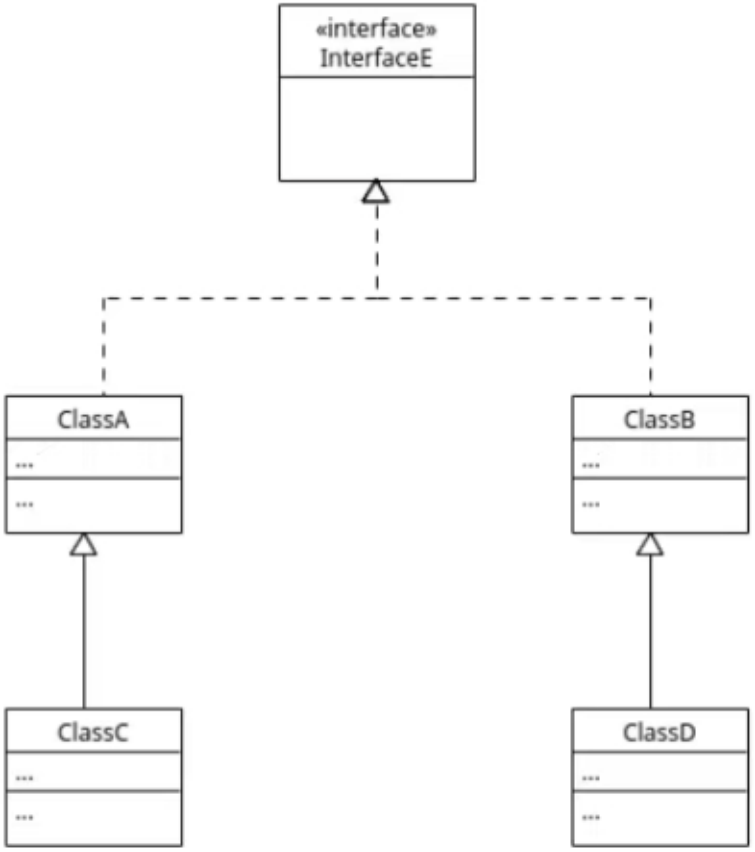
```
*
***
*****
*******
*********
```

```
1. for(int i=0;i<5; i++){
2.     for(<INSERT CODE HERE>){
3.         System.out.print("*");
4.     }
5.     System.out.println();
6. }
```

```
public static void main(String[] args){
    Scanner inputStream = null;

    try{
        inputStream = new Scanner(new FileInputStream(args[1]));
    }catch(FileNotFoundException ffe){
        System.out.println("File not found");
    }

    while(inputStream.hasNextLine()){
        String line = inputStream.nextLine();
        int number = Integer.parseInt(line);
        System.out.println(number);
    }

    inputStream.close();
}
```

- [x] ArrayIndexOutOfBoundsException
- [ ] FileNotFoundException
- [x] NullPointerException
- [ ] ClassCastException
- [x] NumberFormatException

Given the UML diagram below, choose **ALL** pieces of code that will introduce **compilation errors**.



ArrayList<ClassC> aList = new ArrayList<ClassC>();

aList.add(new ClassA());

aList.add(new ClassC());

ArrayList<ClassA> aList = new ArrayList<ClassA>();

aList.add(new ClassA());

aList.add(new ClassB());

aList.add(new ClassC());

aList.add(new ClassD());

ArrayList<InterfaceE> aList = new ArrayList<InterfaceE>();

aList.add(new ClassA());

aList.add(new ClassA());

aList.add(new ClassB());

aList.add(new ClassB());

ArrayList<InterfaceE> aList = new ArrayList<InterfaceE>();

aList.add(new ClassA());

aList.add(new ClassB());

aList.add(new ClassC());

aList.add(new ClassD())

Given the following Java program, which one of the **following** choices given below will you put at **line number 17** to make the program compile?.

```
1. class Main {
2.     public static void mai
n(String[] args){
3.         Son sonObj = new So
n(3, "Noob");
4.     }
5. }
6.
7. class Parent {
8.     protected int x;
9.     Parent(int x) {
10.         this.x = x;
11.     }
12. }
13.
14. class Son extends Parent
{
15.     private String s;
16.     Son(int x, String s)
{
17.         <INSERT CODE HERE
>
18.         this.s = s;
19.     }
20. }
```

super(x)

An iterator method is a classical way to iterate through the ArrayList elements and it returns the elements in an **unordered** way.

True False

A java interface is of type Class?

True False

What is the output of this statement if `a = 25` and `b = 8`?

```
System.out.println(b/2 - a%7.0 +2)
```

The `writeObject()` method of the class `ObjectOutputStream` is used to write content on the text file.

True False

What will be the output for this code snippet?

```
int[] arr = {1,1,0,0,0};
for(int i= 2; i< arr.length -1; i++)
   arr[i] = arr[i-2]+arr[i-1];
for(int i =0 ; i< arr.length; i++)
   System.out.print(arr[i] + “,”);
```

Given the code snippet given below that prints a triangle like this:

```
   *
  ***
 *****
*******
```

```
1. int rows = 7;
2. for (int i = 1; i <= rows / 2 + 1; i++) {
3.     for (int j = 1; j <= rows / 2 + 1 - i; j++)
4.         System.out.print(" ");
5.     for (<INSERT CODE HERE>)
           System.out.print("*");
       System.out.println();
   }
```

- int k = 1; k <= 2 * i - 1; k--
- int k = 1; k <= 2 / i - 1; k--
- int k = 1; k <= 2 * i + 1; k++
- int k = 1; k <= 2 * i - 1; k++

What will be the output of the following piece of code.

```
class Parent {
    void display() {
        System.out.println("Parent class");
    }
}

class Child extends Parent {
    void display() {
        System.out.println("Child class");
    }
}

public class Main {
    public static void main(String[] args) {
        Parent obj = new Parent();
        ((Child) obj).display();
    }
}
```

Runtime Error

Child Class

Parent Class

Compilation Error

What will be the output of the following piece of code.

```
class Parent {
    private int x = 10;
}
class Child extends Parent {
    public void display() {
        System.out.println(x);
    }
}
public class Main {
    public static void main(String[] args) {
        Child obj = new Child();
        obj.display();
```

need get method

syntax error

The following declaration for enumerations in Java is correct? `enum Colours = {RED, GREEN, BLUE}`

class A {
RED,
GREEN,
BLUE
}

Given the following method:

```
 public static <T1, T2> boolean isSameClass(T1 a, T2
b) {
        return (a.getClass() == b.getClass());
}
```

The main method in the class where the above code snippet is given, calls the isSameClass method this way

string int

```
System.out.println(isSameClass("Hello", 3));
```

⇒ false

In not more than 2 lines explain what will be output when the **System.out.println** is called?

1.

Given a Class Pair with the Class definition as:

```
public class Pair<T>
```

The following code snippet is correct and will result in no error.

```
Pair[] a = new Pair[10];
```

Syntax error

```
public class Main {
    public static void main(String[] args) {
        Pair<String>[] a = new Pair[10];
    }
```

Given the code snippet below:

```
public static boolean calculate(){
   try {
       int x = 5 / 0; // exception is thrown here
       return true; // This line is never reached
   }catch (Exception e) {
       return false;
   } finally {
       return true;
   }
}
public static void main(String[] args) {
    System.out.println(calculate());
```

=> true