1. Which method is called when an object is garbage collected in Java?

A. finalize()
B. destroy()
C. dispose()
D. clean()

2. In Java, which keyword is used to implement inheritance between classes?

A. inheritsFrom
B. inherits
C. implements
D. extends

3. What is inheritance in Java?

A. The process of acquiring properties and behaviors of one class by another
B. The process of creating objects
C. The process of encapsulation
D. The process of overloading methods

4. What is the result of the following code snippet?

```
class Parent {

    void display() {

        System.out.println("Parent");
    }

}

class Child extends Parent {

    public static void main(String[] args) {

        Child obj = new Child();

        obj.display();

    }
}
```

A. Runtime exception
B. Compilation error
C. Child
D. Parent

```
 class Parent {

    int x = 10;
}
class Child extends Parent {

    int x = 20;

    void display() {
        System.out.println(super.x + " " + this.x + " " + x);
    }
}
public class Main {

    public static void main(String[] args) {

        Child obj = new Child();

        obj.display();

    }

}
```

A.20 10
B.10 20 20
C.Runtime ex
D.20 10

```
class Main {

    public static void main(String[] args) {

        int p = 3;
        switch (p) {
            case 1:
                System.out.print("One");
                System.out.print("Two");
                System.out.print("Default");

            case 2:

            default:

        }

    }

}
```

A. Default
B. OneTwo Default
C. One
D. DefaultOneTwo

```
class Main {

    public static void main(String[] args) {
        int j = 0;
        while (j < 5) {
            System.out.print(j + " ");
            j += 2;
        }
    }

}
```

A. 5 4 3 2 1

B. 2 4 6 8 10

C. 0 2 4

D.0 1 2 3 4

```
class Main {

    public static void main(String[] args) {

        for (int m = 0; m < 5; m++) {

            if (m == 2) {

                continue;

            }

            System.out.print(m + " ");

        }

    }

}
```

m=0 0<5 0==2

m=1 1<5 1==2

m=2 2<5 2==2

m=3 3<5 3==2

m=4 4<5 4==2

m=5 5<5

A. 0 2 4
B. 2 4
C. 0 1 3 4
D. 0 1 2 3 4

```
class Main {

    public static void main(String[] args) {

        if (m == 2) {

            break;

            for (int m = 0; m < 5; m++) {
            }
        }

        System.out.print(m + " ");

    }

}
```

m=0 0<5 0==2

m=1 1<5 1==2

m=2 2<5 2==2

A. 0 2 4
B. 2 4
C. 0 1 3 4
D.0 1

```
class Main {

    public static void main(String args[]) {

        Encapsulation Demo d = new EncapsulationDemo();
        d.getname();
        d.setname("Java Institute");
    }

}

class Encapsulation

    Demo {

private String name;

    public String getname() {
        return name;
    }

    public void setname(String text) {
        name = text;
        System.out.println(name);
    }

}
```

A. null
B. Java Institute
C. text
D. Compile Error

What is encapsulation in Java?

A. A mechanism to hide the implementation details of a class
B. A way to achieve inheritance between classes
C. A technique to perform type casting in Java
D. A feature to handle exceptions in Java

What is the purpose of getter methods in encapsulation?

A. To modify the value of a variable
B. To retrieve the value of a variable
C. To create a new instance of a class
D. To delete an instance of a class

Which of the following is an example of encapsulation in action?

A. A class with only public instance variables.
B. A class with private instance variables and public getter and setter methods.
C. A class with no instance variables.
D. A class with only static methods.

Below class ABC doesn't have even a single abstract method, but it has been declared as abstract. Is it correct?

```
abstract class ABC {
    void firstMethod() {
        System.out.println("First Method");
    }
    void secondMethod() {
        System.out.println("Second Method");


    }
}
```

yes

```
abstract class AbstractClass {

    abstract void abstractMethod() {
        System.out.println("First Method");
    }

}
```

Compilation Error

Which class is instantiable? Class A or Class B?

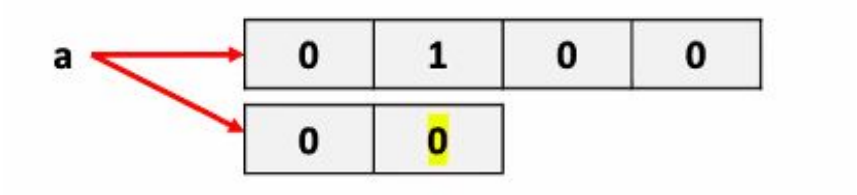
```
abstract class A {

}

class B extends A {

}
```

Length එකෙන් බලන්නෙ අපි තියන elements ගන.

What does the length attribute of an array in Java represent

A. The number of elements in the array
B. The maximum value of elements in the array
C. The length of the variable name
D. The number of dimensions

```
class Test {

    public static void main(String[] args) {

        int[] a = new int[4];

        a[1] = 1;

        a = new int[2];

        System.out.println("a[1] is " + a[1]);

    }

}
```



A. The program has a compile error because new int[2]
B. The program has a runtime error because a[1] = null
C. a[1] is 0
D. a[1] is 1

What would be the result of attempting to compile and run the following code?

```
public class HelloWorld {

    public static void main(String[] args) {
        double[] x = new double[]{1, 2, 3};
        System.out.println("Value is " + x[1]);

    }

}
```

A. The program has a compile error because the syntax new double[\{1, 2, 3\} is wrong and it should be replaced by \{1, 2, 3\} .

B. The program has a compile error because the syntax new double [] \{1, 2, 3\} is wrong and it should be replaced by new double [3]\{1, 2, 3\} ;

C. The program has a compile error because the syntax new double[ ] \{1, 2, 3\} is wrong and it should be replaced by new double[] \{1, 2, 3\}

D. The program compiles and runs fine and the output : Value is 2.0

```
class Test {

    public static void main(String[] args) {

        int[] fun = new int[5];
        fun[0] = 1;
        fun[1] = 2;
        fun[2] = 3;
        fun[3] = 4;
        fun[4] = 5;
        int j = 3;
        System.out.println(fun[j - 1]);
    }
}
}
```

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

```
class Test {
    public static void main(String[] args) {
        int[] odd = {1, 3, 5, 7, 9, 11
        )
System.out.println(odd[0] + " " + odd[3]);

    }
}
```

A. 1 5
B. 6
C. 1 7
D. 8

```
class Test {

    public static void main(String[] args) {
        int[] evens = {2, 4, 6, 8, 10};
        evens[0] = 44;
        evens[4] = evens[2];
        System.out.println(evens[0] + " " + evens[4]);
    }
}
```

A. 44 10
B. 2 10
C. 54
D. 44 6

```
class Test {

    public static void main(String[] args) {
        double[] x = new double[4];
        x[0] = 8.5
        x[1] = 6.5
        x[2] = 9.5
        x[3] = 12.5
        System.out.println(x[1 + 2])
    }
}
```

A. 6.5 9.5
B. 12.5
C. 15.0
D. 9.5 12.5

```
class Test {

   public static void main(String[] args) {
      int[] car = new int[7];
      car[0] = 2004;
      car[1] = 2006;
      System.out.println(car[0] + " " + car[1] + " " + car[7]);

   }
}
```

A. 2004 2006 0
B. 2004 2006
C. 4010
D. An error will occur

```
    class A {

       A(int x) {
          System.out.println("A");
       }
    }

    class Test {

       public static void main(String[] args) {
          A a = new A();

       }
    }
```

Compile error

```
class A {

    A(int x) {
        System.out.println("A");
    }

}

class B extends A {

}

class Test {

    public static void main(String[] args) {
        B b = new B();
    }

}
```

A. Compilation error at class B extends A{ ... }

```
class A {

    A() {
        System.out.println("A");
    }
}

class B extends A {

    B() {
        this(10);
        System.out.println("B 1");
    }

    B(int x) {
        System.out.println("B 2");
    }
}

class Test {
    public static void main(String[] args) {
        B b = new B();
    }
}
```

```
A
B2
B1
```

```
class A {

    public void m() {

        System.out.println("A m");
    }
}

class B extends A {

    public void m() {

        System.out.println("B m");

    }
}

class Test {

    public static void main(String[] args) {

        B b = new A();

    }
}
```

Compile Error

```
    class A {
        public static void m() {
            System.out.println("A m");
        }
    }

 class B extends A
    public static void m() {
        System.out.println("B m");
    }
}

class Test {
    public static void main(String[] args) {
        B b = new B();
        b.m();

    }
}
```

B m

```
class A {

    public void m() {
        System.out.println("A m");
    }
}

class B extends A {

    protected void m() {
        System.out.println("B m");
    }
}

class Test {

    public static void main(String[] args) {
        B b = new B();
        b.m();
    }

}
```

Compile Error