OOPS TEST

1

Objects are the variables of the type ____?

A. String
B. Boolean
C. Class
D. All data types can be included

2

5. Which feature's behavior of OOP depends upon the types of data used in the operation.

A. Inheritance
B. Polymorphism
C. Abstraction
D. Encapsulation

3

4. ____ member variables are initialized to zero when the first object of its class is created?

A. Static
B. Local
C. Global
D. External

4

3. _____ is associated with polymorphism and inheritance.

A. Message parsing
B. Abstraction
C. Dynamic Binding
D. Encapsulation

5

```
class Automobile {
    private String drive() {
        return "Driving vehicle";
    }
}

class Car extends Automobile {
    protected String drive() {
        return "Driving car";
    }
}

public class ElectricCar extends Car {

    @Override
    public final String drive() {
        return "Driving an electric car";
    }

    public static void main(String[] wheels) {
        final Car car = new ElectricCar();
        System.out.print(car.drive());
    }
}
```

A. Driving vehicle

B. Driving an electric car

C. Driving car

D. The code does not compile

6

```java
class A {
  int i;
}

class B extends A {

  int i;

  B(int a, int b) {
    super.i = a;
    i = b;
  }

  void show() {
    System.out.println("i in superclass: " + super.i);
    System.out.println("i in subclass: " + i);
  }
}

class UseSuper {

  Run | Debug
  public static void main(String args[]) {
    B subOb = new B(a:1, b:2);
    subOb.show();
  }
}
```

A) Compile Time Error

B) I in superclass:0
   I in subclass:0

C) I in superclass:1
   I in subclass:2

D) Run Time Error

7

```java
class A {

  A() {
    System.out.println(x:"Inside A's constructor.");
  }
}

class B extends A {

  B() {
    System.out.println(x:"Inside B's constructor.");
  }
}
class C extends B {

  C() {
    System.out.println(x:"Inside C's constructor.");
  }
}

class question {
  Run | Debug
  public static void main(String args[]) {
    C c = new C();
  }
}
```

A)

```
Inside A's constructor.
Inside B's constructor.
Inside C's constructor.
```

B)

```
Inside C's constructor.
Inside B's constructor.
Inside A's constructor.
```

C)

```
Inside C's constructor.
```

8)

```
public static void main(String[] args) {
    method(str:null);
}

public static void method(String str) {
    System.out.println(x:"String method called");
}

public static void method(Object obj) {
    System.out.println(x:"Object method called");
}
```

A)String method called
B)Object method called
C)Compile Time Error
D) Null

9)

```
String a = "a";
String b = "a";
String c = new String("a");
String d = new String("a");


System.out.println(a == b);
System.out.println(b == c);
System.out.println(a == c);
System.out.println(c == d);
```

A) true false false true
B) true false false false
C) true true true false
D) false true true false

10)

```java
class SuperClass {
    SuperClass() {
        foo();
    }

    public void foo() {
        System.out.println(x:"In SuperClass.foo()");
    }
}

class SubClass extends SuperClass {
    private String member;

    public SubClass() {
        member = "HI";
    }

    public void foo() {
        System.out.println("In SubClass.foo(): " + member.toLowerCase());
    }
}

public class question {
    Run | Debug
    public static void main(String[] args) {
        SuperClass reference = new SubClass();
        reference.foo();
    }
}
```

a) In SuperClass.foo()

b) In Derived.foo(): hi

c) In SuperClass.foo() In Derived.foo(): hi

d) This program throws a NullPointerException.

11)

```
class Grandparent
{
    public void Print()
    {
        System.out.println("Grandparent's Print()");
    }
}

class Parent extends Grandparent
{
    public void Print()
    {
        System.out.println("Parent's Print()");
    }
}

class Child extends Parent
{
    public void Print()
    {
        super.super.Print();
        System.out.println("Child's Print()");
    }
}

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

A) Grandparent's Print()
   Parent's Print()
   Child's Print()
B) Parent's Print()
   Child's Print()
C) Compile Time Error
D) Run Time Error

12)

```
public class A extends B
{
    public static String sing()
    {
        return "fa";
    }
    public static void main(String[] args)
    {
        A a = new A();
        B b = new A();
    System.out.println(a.sing() + " " + b.sing());
    }
}
class B
{
    public static String sing()
    {
        return "la";
    }
}
```

A) Fa la
B) La Fa
C) Fa fa
D) La la
E) Run Time Exception

0

13)

```
class Point
{
    int x, y;

    public Point(int x, int y)
    {
        this.x = x;
        this.y = y;
    }

    public static void main(String args[])
    {
      Point obj = new Point();
    }
}
```

A) The code will be executed by calling default constructor of class Point

B) Compile Time Error

C) Run Time Error

14)

Q. What is the correct way to access a protected variable "count" from a subclass named "Child" in Java?

- ○A. child.count
- ◉B. super.count
- ○C. this.count
- ○D. count

15

The feature in object-oriented programming that allows the same operation to be carried out differently, depending on the object, is:

(A) Inheritance

(B) Polymorphism

(C) Overfunctioning

(D) Overriding