HESHAN SANDARUWAN
200304512443
2301691035

01.

a. **class or template is a blueprint that defines the properties and behaviors of objects. It acts as a template to create objects with similar attributes and methods.**

b. **An object or instance is a concrete realization of a class. It is created using the blueprint defined by the class.**

c. **blocks of code that perform specific tasks.**

d. **Attributes or properties are the characteristics or data associated with a class or an object.**

e. **Reference variables store references (memory addresses) to objects rather than holding the actual object data.**

f. **Primitive variables hold simple, predefined data types (e.g., integers, floats, booleans) and directly store their values.**

g. **Method parameters are variables that are specified in the method signature to receive values when the method is called.**

h. **Local variables are declared within a method, constructor, or block and are only accessible within that scope.**

I. **Default values are the initial values automatically assigned to variables if no explicit value is provided.**

02. **D**

03. **89**

04. **Attributes:** Attributes are the characteristics or properties that describe the state of an object. They represent the data an object holds.

Ex: color/model/year

Methods: Methods, on the other hand, are actions or behaviors that an object can perform.

Ex: startEngine()/brake()

HESHAN SANDARUWAN
200304512443
2301691035

05. 2/3/4

variable directly without specifying the object it belongs to.

06. Volume : 0
length of box : 0
width of box : 0
height of box : 0

07. Volume : 180
length of box : 12
width of box : 5
height of box : 3

08. default constructor
length of box : 2
width of box : 2
height of box : 2

09. Parameterized constructor
Volume : 60
Parameterized constructor
Volume : 180

10. compile error: width has private access in Box.

11. 4,5,6
This line attempts to assign a value to the `length` variable directly within the class body.

12. *initializing the object of that class.
***constructors are used to set initial values to the instance variables or perform any necessary setup operations when an object is instantiated.**

13. constructors-class eke namin sade
no return type
nitializing objects when they are created

method-has return type
kamathi name ekak damiya haka

HESHAN SANDARUWAN
200304512443
2301691035

14. **a,b,c**

15. **line 2 and 3 not variable.**

16. **1 2 3 4 - c**

17. **a,b,d**

18. **100 101**
0 0

19. **Code : 3001**

20. **Code : 2001**

21. **b,c**

22. **a,d,e**

23. **Encapsulation is a fundamental concept in object-oriented programming (OOP) that involves bundling the data (attributes) and the methods (functions or behaviors) that operate on the data into a single unit, called a class. It allows for the protection and control of access to the internal state of an object, ensuring that the data is kept safe from external interference and misuse.**

24. **"Tightly encapsulated" and "loosely encapsulated" refer to the degree to which the internal workings of a class or an object are hidden and how much control is exerted over access to its attributes and methods.**

25. **D. Compiler error at line 2**

26.
```
class Date{
int year=1970;
int month=1;
int day=1;

void printDate(){
        System.out.println(year+"-"+month+"-"+day);
}
void setYear(int x){}

void setMonth(int y){}
```

HESHAN SANDARUWAN
200304512443
2301691035

```
        void setDay(int z){}

        int getYear(){
                return year;

        }
        int getMonth(){
                return month;
        }
        int getDay(){
                return day;
                }
        }

//--------------------Demo.java-----------------------
        class Main{
        public static void main(String args[]){
                Date d1=new Date();
                d1.printDate(); //1970-1-1
                d1.year=2016; //Illegal
                d1.month=5; //Illegal
                d1.day=30; //Illegal
/*year, month and day attributes
*cannot be accessed to another class
*/
                d1.setYear(2016);
                d1.setMonth(5);
                d1.setDay(31);
        System.out.println("Year : "+d1.getYear());
        System.out.println("Month :"+d1.getMonth());
        System.out.println("Day : "+d1.getDay());
        }
}
```

27.

```
line 1-  Declaration of c1
line 2-  Instantiating c1 with id="C001" and name="Danapala"
line 3-  Printing details of c1
line 4-  Declaration of c2
line 5-  Instantiating c2 using the default constructor
line 6-  Setting id="C002" and name="Gunapala" for c2
line 7-  Printing details of c2
line 8-  Declaration of c3
line 9-  Instantiating c3 using the default constructor
line 10- Setting id="C003" for c3
line 11- Setting name="Somapala" for c3
```

HESHAN SANDARUWAN
200304512443
2301691035

line 12- Printing details of c3

28. A. Compile Error at line 1
B. Compile Error at line 2

29. f

30.
```
import java.util.*;
class Rectangle{
     int width;
     int length;

public void Perimeter(int x,int y){
     width=x;
     length=y;

     if(width>0.0 && width<20.00){
          int perimeter=(2*width)+(2*length);
          System.out.println( "perimeter = "+perimeter);
          }else{
               System.out.println("wrong rectangle");

               }

     }

public void Area(int x,int y){
     width=x;
     length=y;

     if(width>0.0 && width<20.00){
          int area=width*length;
          System.out.println( "Area = "+area);
          }else{
               System.out.println("wrong rectangle");
               }
          }

     }

public class Main{
     public static void  main(String args[]){
     Rectangle r1 = new Rectangle();
```

HESHAN SANDARUWAN
200304512443
2301691035

```
Scanner input = new Scanner(System.in);

System.out.print("Rectangle width :");
int w = input.nextInt();

System.out.print("Rectangle length :");
int l = input.nextInt();

r1.Perimeter(w,l);
r1.Area(w,l);

}}
```

31. **A. Compile Error at line 1**
C. Compile Error at line 3

32. **E. 1 200 10 200 200 200**

33. **line 5**
line 8

34. **Instance Variables:**

- These are variables that belong to a specific instance (or object) of a class.
- Each instance of the class has its own copy of instance variables.
- They define the properties or attributes of an object and can have different values for each instance of the class.
- Instance variables are declared within a class but outside of any method or constructor in the class.

- **Static Variables:**

- These are variables that belong to the class itself, rather than to any specific instance of the class.
- There is only one copy of a static variable regardless of how many instances (objects) of the class are created.
- They are shared among all instances of the class.
- Static variables are declared with the `static` keyword within a class, usually outside of any method.

35. **A. Compile Error at line 1**

D. Compile Error at line 4

HESHAN SANDARUWAN
200304512443
2301691035

F. Compile Error at line 6

I. Compile Error at line 9

```
36.     class Date {
   private int year;
   private int month;
   private int day;

   public static final int YEAR = 1;
   public static final int MONTH = 2;
   public static final int DAY = 3;

   public void set(int field, int value) {
      switch (field) {
         case YEAR:
            this.year = value;
            break;
         case MONTH:
            this.month = value;
            break;
         case DAY:
            this.day = value;
            break;
         default:
            System.out.println("Invalid field");
      }
   }
```

HESHAN SANDARUWAN
200304512443
2301691035

```
    public void printDate() {
        System.out.println(year + "-" + month + "-" + day);
    }
}

class Main {
    public static void main(String args[]) {
        Date d1 = new Date();
        d1.set(Date.YEAR, 2016);
        d1.set(Date.MONTH, 5);
        d1.set(Date.DAY, 30);
        d1.printDate(); // Output: 2016-5-30
    }
}
```

37. A. Prints: [a,0],[b,1],1,0,0,2

38. C. Prints: 1,2,3,4,5,3

39. Instance Methods:

- **Context:** Instance methods are associated with objects (instances) of a class. They operate on instance variables and can access and modify the state of the object.
- **Access:** They are accessed via object instances and implicitly take the instance (`self` in Python, `this` in Java) as the first parameter. This parameter represents the instance itself and allows access to its attributes and methods.

- **Static Methods:**
- **Context:** Static methods are not bound to any specific instance of the class. They don't have access to instance variables or instance methods.

HESHAN SANDARUWAN
200304512443
2301691035

- **Access:** They are accessed via the class itself, not through instances. They do not require a reference to an instance (`self` or `this`) to be passed as the first parameter.

40. class Cylinder {

private double length;

private double radius;

public Cylinder() {

this.length = 0.0;

this.radius = 0.0;

}

public Cylinder(double length, double radius) {

this.length = length;

this.radius = radius;

}

public double getLength() {

return length;

}

public void setLength(double length) {

this.length = length;

}

public double getRadius() {

return radius;

HESHAN SANDARUWAN
200304512443
2301691035

```
    }

    public void setRadius(double radius) {
        this.radius = radius;
    }

    public double calculateVolume() {
        return 3.14* radius * radius * length;
    }

    public double calculateSurfaceArea() {
        return (2 * 3.14 * radius * length) + (2 * 3.14 * radius * radius);
    }
}
public class Main {
    public static void main(String[] args) {
        Cylinder cylinder1 = new Cylinder();
        cylinder1.setLength(5.0);
        cylinder1.setRadius(2.0);

        double volume1 = cylinder1.calculateVolume();
        double area1 = cylinder1.calculateSurfaceArea();

        System.out.println("Cylinder 1:");
        System.out.println("Volume: " + volume1);
        System.out.println("Surface Area: " + area1);

        Cylinder cylinder2 = new Cylinder(8.0, 3.5);
```

HESHAN SANDARUWAN
200304512443
2301691035

```
        double volume2 = cylinder2.calculateVolume();
        double area2 = cylinder2.calculateSurfaceArea();

        System.out.println("\nCylinder 2:");
        System.out.println("Volume: " + volume2);
        System.out.println("Surface Area: " + area2);
    }
}
```

41. **Box is loaded into memory**

42. **A box object is created..**
A box object is created..

43. **Box is loaded into memory**
A box object is created..
A box object is created..
A box object is created..

44.
- **a.** **1 2 3**
- **b.** **1 2 3 2 3**
- **c.** **1 4**
- **d.** **compile error.**
- **e.** **1 4**
- **f.** **blank**

HESHAN SANDARUWAN
200304512443
2301691035

45. Line 3: **Cannot use this in a static context**

Line 4: **Cannot use this in a static context**

46. **Constructor Overloading:**

```
public class Rectangle {
    private int length;
    private int width;

    // Default constructor
    public Rectangle() {
        length = 0;
        width = 0;
    }

    public Rectangle(int length, int width) {
        this.length = length;
        this.width = width;
    }

    public Rectangle(int side) {
        this.length = side;
        this.width = side;
    }

}
```

Method Overloading:

```
public class Calculator {
```

HESHAN SANDARUWAN
200304512443
2301691035

```
    public int add(int a, int b) {
        return a + b;
    }


    public int add(int a, int b, int c) {
        return a + b + c;
    }


    public double add(double a, double b) {
        return a + b;
    }


}
```

47. **a,b,c,d,e,f,h**

48. **a,b,c,e,f,g**

49. **C. (5, 3) (3, 5) (5, 3)**

50. Example of Method Call by Value in Java:

```
public class Example {
```

HESHAN SANDARUWAN
200304512443
2301691035

```
    public static void main(String[] args) {
        int number = 10;
        System.out.println("Before method call: " + number);
        modifyValue(number);
        System.out.println("After method call: " + number);
    }

    public static void modifyValue(int value) {
        value = 20; // Changes made here won't affect the original 'number'
        System.out.println("Inside method: " + value);
    }
}
```

OUTPUT

//Before method call: Alice

//Inside method: Bob

//After method call: Bob

Example of Method Call by Reference in Java (using Objects):

```
class Person {
    String name;

    public Person(String name) {
        this.name = name;
    }
}
```

HESHAN SANDARUWAN
200304512443
2301691035

```
public class Example {
    public static void main(String[] args) {
        Person person = new Person("Alice");
        System.out.println("Before method call: " + person.name);
        modifyReference(person);
        System.out.println("After method call: " + person.name);
    }

    public static void modifyReference(Person p) {
        p.name = "Bob"; // Changes made here affect the original 'person' object
        System.out.println("Inside method: " + p.name);
    }
}

//Before method call: Alice
//Inside method: Bob
//After method call: Bob
```

51. line 3 and 5.

52.

```
class Cat{
private String name;
Cat(String name){
        this.name=name;
        }
public String getName(){
```

HESHAN SANDARUWAN
200304512443
2301691035

```
        return name;
        }
public void setName(String name){
        this.name=name;
        }
}
class Main{
public static void main(String args[]){
Cat[] cats={new Cat("Aldo"),new Cat("Bear"),new Cat("Toby"),new Cat("Teddy"),new Cat("Henry")};
 System.out.print("[");
for (int i = 0; i < cats.length; i++)
{
        System.out.print(cats[i].getName());
         if (i < cats.length - 1) {
            System.out.print(", ");
        }
}
         System.out.println("]");
}
}
```

```
53.     public class Employee {
   private String firstName;
   private String lastName;
   private double monthlySalary;

   public Employee(String firstName, String lastName, double monthlySalary) {
```

HESHAN SANDARUWAN
200304512443
2301691035

```
        this.firstName = firstName;
        this.lastName = lastName;
        if (monthlySalary > 0) {
            this.monthlySalary = monthlySalary;
        }
    }

    public String getFirstName() {
        return firstName;
    }

    public void setFirstName(String firstName) {
        this.firstName = firstName;
    }

    public String getLastName() {
        return lastName;
    }

    public void setLastName(String lastName) {
        this.lastName = lastName;
    }

    public double getMonthlySalary() {
        return monthlySalary;
    }

    public void setMonthlySalary(double monthlySalary) {
```

HESHAN SANDARUWAN
200304512443
2301691035

```
        if (monthlySalary > 0) {
            this.monthlySalary = monthlySalary;
        }
    }

    public double calculateYearlySalary() {
        return monthlySalary * 12;
    }

    public void applyRaise() {
        double raise = monthlySalary * 0.10;
        this.monthlySalary += raise;
    }
}

public class EmployeeTest {
    public static void main(String[] args) {
        Employee emp1 = new Employee("John", "Doe", 5000);
        Employee emp2 = new Employee("Jane", "Smith", 6000);

        System.out.println("Yearly salary for " + emp1.getFirstName() + " " +
emp1.getLastName() + ": $" + emp1.calculateYearlySalary());
        System.out.println("Yearly salary for " + emp2.getFirstName() + " " +
emp2.getLastName() + ": $" + emp2.calculateYearlySalary());

        emp1.applyRaise();
        emp2.applyRaise();
```

HESHAN SANDARUWAN
200304512443
2301691035

```
        System.out.println("\nAfter 10% raise:");
        System.out.println("Yearly salary for " + emp1.getFirstName() + " " +
emp1.getLastName() + ": $" + emp1.calculateYearlySalary());
        System.out.println("Yearly salary for " + emp2.getFirstName() + " " +
emp2.getLastName() + ": $" + emp2.calculateYearlySalary());
    }
}
```

54. code : 1001

    code : 1001

55.
```
class Date {
  private int year;
  private int month;
  private int day;

  public Date() {
  }

  public Date(int year, int month, int day) {
    this.year = year;
    this.month = month;
    this.day = day;
  }

  public Date(Date otherDate) {
    this.year = otherDate.year;
    this.month = otherDate.month;
```

HESHAN SANDARUWAN
200304512443
2301691035

```
    this.day = otherDate.day;
}

public void set(int field, int value) {
}

public void set(Date otherDate) {
    this.year = otherDate.year;
    this.month = otherDate.month;
    this.day = otherDate.day;
}

public void printDate() {
    System.out.println(year + "-" + month + "-" + day);
}

public static Date getDateInstance() {
    return new Date(1970, 1, 1);
}



public int getYear() {
    return year;
}

public void setYear(int year) {
    this.year = year;
}
```

HESHAN SANDARUWAN
200304512443
2301691035

}

```
public class DemoDate {
   public static void main(String args[]) {
      Date d1 = new Date();
      d1.set(Date.YEAR, 2016);
      d1.set(Date.MONTH, 5);
      d1.set(Date.DAY, 30);
      d1.printDate(); //2016-5-30

      Date d2 = new Date(2016, 1, 31);
      d2.printDate(); //2016-1-31

      Date d3 = new Date(d2);
      d3.printDate(); //2016-1-31

      Date d4 = new Date();
      d4.set(d1);
      d4.printDate();

      Date d5 = Date.getDateInstance();
      d5.printDate(); //1970-1-1
   }
}
```

56. C. 10,0,20

HESHAN SANDARUWAN
200304512443
2301691035

```
57.      class Invoice {
   private String partNumber;
   private String partDescription;
   private int quantity;
   private double pricePerItem;
   public Invoice(String partNumber, String partDescription, int quantity, double
pricePerItem) {
      this.partNumber = partNumber;
      this.partDescription = partDescription;
      if (quantity > 0) {
         this.quantity = quantity;
      } else {
         this.quantity = 0;
      }
      if (pricePerItem > 0.0) {
         this.pricePerItem = pricePerItem;
      } else {
         this.pricePerItem = 0.0;
      }
   }

   public String getPartNumber() {
      return partNumber;
   }

   public void setPartNumber(String partNumber) {
```

HESHAN SANDARUWAN
200304512443
2301691035

```
    this.partNumber = partNumber;
}

public String getPartDescription() {
    return partDescription;
}

public void setPartDescription(String partDescription) {
    this.partDescription = partDescription;
}

public int getQuantity() {
    return quantity;
}

public void setQuantity(int quantity) {
    if (quantity > 0) {
        this.quantity = quantity;
    } else {
        this.quantity = 0;
    }
}

public double getPricePerItem() {
    return pricePerItem;
}

public void setPricePerItem(double pricePerItem) {
```

HESHAN SANDARUWAN
200304512443
2301691035

```
        if (pricePerItem > 0.0) {
            this.pricePerItem = pricePerItem;
        } else {
            this.pricePerItem = 0.0;
        }
    }

    public double getInvoiceAmount() {
        return quantity * pricePerItem;
    }
}

public class Main {
    public static void main(String[] args) {
        Invoice invoice = new Invoice("A001", "Hammer", 5, 12.50);

        System.out.println("Part Number: " + invoice.getPartNumber());
        System.out.println("Part Description: " + invoice.getPartDescription());
        System.out.println("Quantity: " + invoice.getQuantity());
        System.out.println("Price per Item: $" + invoice.getPricePerItem());
        System.out.println("Invoice Amount: $" + invoice.getInvoiceAmount());

        invoice.setQuantity(-3);
        invoice.setPricePerItem(-7.0);

        System.out.println("\nUpdated Quantity: " + invoice.getQuantity());
        System.out.println("Updated Price per Item: $" + invoice.getPricePerItem());
        System.out.println("Updated Invoice Amount: $" + invoice.getInvoiceAmount());
```

HESHAN SANDARUWAN
200304512443
2301691035

```
    }
}
```

58.

```
class Box {
    private int length;
    private int width;
    private int height;

    // Constructors
    public Box() {
        // Default constructor
    }

    public Box(int length) {
        this.length = length;
        this.width = length;
        this.height = length;
    }

    public Box(int length, int width, int height) {
        this.length = length;
        this.width = width;
        this.height = height;
    }

    public Box(Box otherBox) {
        this.length = otherBox.length;
        this.width = otherBox.width;
```

HESHAN SANDARUWAN
200304512443
2301691035

```
        this.height = otherBox.height;
    }

    // Setters
    public void setLength(int length) {
        this.length = length;
    }

    public void setWidth(int width) {
        this.width = width;
    }

    public void setHeight(int height) {
        this.height = height;
    }

    public void setDimension(int length, int width, int height) {
        this.length = length;
        this.width = width;
        this.height = height;
    }

    public void setDimension(Box otherBox) {
        this.length = otherBox.length;
        this.width = otherBox.width;
        this.height = otherBox.height;
    }
```

HESHAN SANDARUWAN
200304512443
2301691035

```
    // Getters
    public int getLength() {
        return length;
    }

    public int getWidth() {
        return width;
    }

    public int getHeight() {
        return height;
    }

    // Method to calculate volume
    public int getVolume() {
        return length * width * height;
    }

    // Method to print volume
    public void printVolume() {
        System.out.println("Volume: " + getVolume());
    }

    // Method to create a copy of the Box
    public Box getCopy() {
        return new Box(this);
    }
}
```

HESHAN SANDARUWAN
200304512443
2301691035

```
class Demo {
    public static void main(String args[]) {
        Box b1 = new Box();
        b1.setLength(12);
        b1.setWidth(5);
        b1.setHeight(3);
        b1.printVolume();
        b1.setDimension(120, 50, 30);
        System.out.println("Volume " + b1.getVolume());

        Box b2 = new Box(4, 2, 3);
        b2.printVolume();

        Box b3 = new Box(b2);
        b3.printVolume();

        Box b4 = new Box(10); // length for a square cube
        b4.printVolume(); //

        Box b5 = new Box();
        b5.setDimension(12); // length for a square cube
        b5.printVolume();

        Box b6 = new Box();
        b6.printVolume();
        b6.setDimension(b1); // copy dimensions of b1 into b6
        b6.printVolume();
```

HESHAN SANDARUWAN
200304512443
2301691035

```
        Box b7 = b3.getCopy();
        b7.printVolume();
    }
}
```

59. **e,k**

60. **E. Prints: 1, 2, 3, 4, 9,**

61. **MyClass()**

MyClass(int,int)

MyClass(int)

62. **s,d,f**