HESHAN SANDARUWAN
200304512443

01.

A – super class
B – sub class
C – super class and sub class
D – sub class
E – sub class

02.

The `Car` class inherits common features from `Vehicle` (like `numberOfWheels` and `startEngine()`), and it adds its unique characteristics like `numberOfDoors`.
The `Motorcycle` class also inherits common features from `Vehicle` and introduces its specific attributes like `handleType` and methods like `wheelie()`

03.

nheritance allows classes to inherit properties and behaviors from other classes. By creating a hierarchy of classes, common functionalities can be defined in a superclass and shared by multiple subclasses. This reuse of code prevents the need to rewrite or duplicate the same code across different parts of the program.

04.

```
    //01
class Student{ //super class


    }
class GradateStdent extends Student{} //sub class
class UnderraduateStudent extends Student{}  //sub class


//02
class Shape{   //super class
```

HESHAN SANDARUWAN
200304512443

```
    }
class Circle extends Shape{} //sub class
class Triangle extends Shape{}  //sub class
class Rectangle extends Shape{} // sub class
class Sphere extends Shape{} // sub class
class Cube extends Shape{} // sub class

//03
class Loan{  //super class

    }
class Carloan extends Loan{} //sub class
class HomeImprovementLoan extends Loan{}  //sub class
class MortgageLoan extends Loan{}  //sub class

//04
class Employee{ //super class

    }

class EmployeeFaculty extends Employee{} //sub class
class Staff extends Employee{}  //sub class

//05
class BankAccount{ //super class
```

HESHAN SANDARUWAN
200304512443

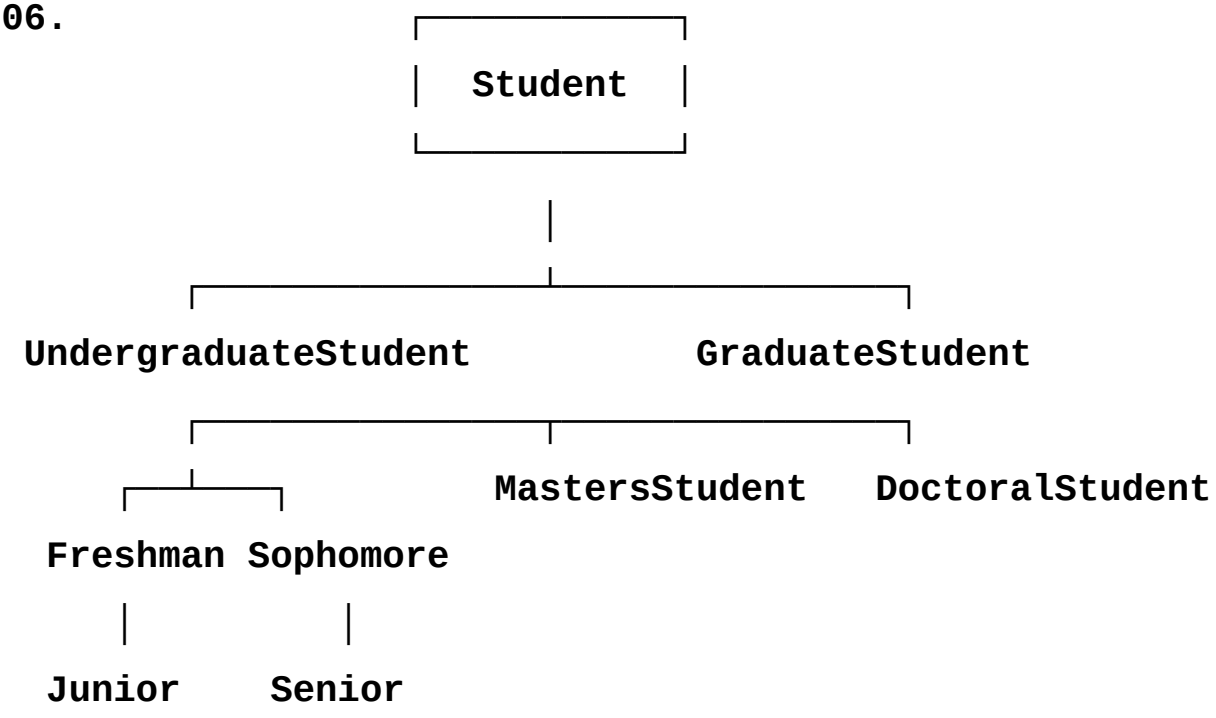
```
    }

class CheckingAccount extends BankAccount{} //sub class
class SavingsAccount extends BankAccount{}  //sub class



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

        }
    }
05.
06.
```



HESHAN SANDARUWAN
200304512443

07.
08.

```
        import javax.swing.JFrame;
class CalculatorView extends JFrame{
CalculatorView(){
        setSize(300,300);
        setTitle("Calculator");
        setDefaultCloseOperation(CalculatorView.EXIT_ON_CLOSE);
        }
}
class Main{
public static void main(String []args){
        CalculatorView v1=new CalculatorView();
        v1.setVisible(true);
        }
}
```

09.

A,B,C,D,E,F

10.

line 2 and 4.

11.

b, c

12.

A, B, C, and D

13.

A,B,C,D

14.

compile error(no argument)

15.

9,10,12,13,14,15,16,17

16.

b

HESHAN SANDARUWAN
200304512443

17.

D

18.

all awnswers incorrect.

19.

compile error no argument.

20.

s,d,e,f

21.

```
class Superclass {
   private int superClassValue;

   // Superclass constructor
   public Superclass(int value) {
      this.superClassValue = value;
      System.out.println("Superclass constructor invoked with value: " +
superClassValue);
   }
}

class Subclass extends Superclass {
   private int subClassValue;

   // Subclass constructor
   public Subclass(int value1, int value2) {
      super(value1); // Invoking superclass constructor explicitly with super()
      this.subClassValue = value2;
      System.out.println("Subclass constructor invoked with value: " +
subClassValue);
   }
}

public class Main {
   public static void main(String[] args) {
      Subclass obj = new Subclass(10, 20);
   }}
```

HESHAN SANDARUWAN
200304512443

22.

```
        class Superclass {
    public String field = "Superclass Field";
}

class Subclass extends Superclass {
    public String field = "Subclass Field";

    public void printFields() {
        System.out.println("Field in superclass: " + super.field);
        System.out.println("Field in subclass: " + this.field);
    }
}

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

23.

G. method defined in C from a method in A.

24.

F. hi hi, followed by an exception

25.

26.

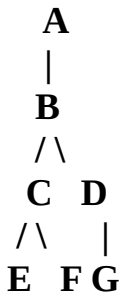
Inside Area for Rectangle.
Area is 45.0
Inside Area for Triangle.
Area is 40.0
undefined
Area is 0.0

HESHAN SANDARUWAN
200304512443

27.

new F(); thibboth compile error
nattam output – D

28.



29.

D. none of the above;

30.

a,b,c,d

31.

e and f.

32.

C and D.

33.

ABC

34.

B and C.

35.

```
class Customer {
  private int code;
  private String name;

  public Customer(int code, String name) {
     this.code = code;
     this.name = name;
  }
```

HESHAN SANDARUWAN
200304512443

```
    public String toString() {
        return code + "-" + name;
    }
}

class CustomerStack {
    private Customer[] stack;
    private int top;
    private int capacity;

    public CustomerStack() {
        capacity = 10; // Initial capacity set to 10 (can be adjusted)
        stack = new Customer[capacity];
        top = -1; // Stack is initially empty
    }

    public void push(Customer customer) {
        if (top == capacity - 1) {
            System.out.println("Stack overflow. Cannot add more customers.");
            return;
        }
        stack[++top] = customer;
    }

    public Customer pop() {
        if (top == -1) {
            System.out.println("Stack is empty. Cannot pop.");
            return null;
        }
        return stack[top--];
    }

    public void printCustomerStack() {
        System.out.print("[");
        for (int i = top; i >= 0; i--) {
            System.out.print(stack[i]);
            if (i != 0) {
                System.out.print(", ");
            }
        }
        System.out.println("]");
```

```
    }
}

class Demo {
    public static void main(String args[]) {
        CustomerStack stack = new CustomerStack();
        stack.push(new Customer(1001, "Danapala"));
        stack.push(new Customer(1002, "Gunapala"));
        stack.push(new Customer(1003, "Somapala"));
        stack.push(new Customer(1004, "Siripala"));
        stack.printCustomerStack();

        stack.pop();
        stack.printCustomerStack();
    }
}
```

36.

```
        class Demo {
    public static void main(String args[]) {
        VehicleQueue queue = new VehicleQueue();
        queue.enQueue(new Car("C001"));
        queue.enQueue(new Bus("B001"));
        queue.enQueue(new Bus("B002"));
        queue.enQueue(new Car("C002"));
        queue.enQueue(new Car("C003"));
        queue.enQueue(new Van("V001"));
        queue.enQueue(new Car("V002"));
        queue.enQueue(new Bus("B003"));
        queue.printVehicleQueue();

        queue.callPark();
        queue.deQueue();
        queue.printVehicleQueue();
    }
}

class Vehicle {
    protected String id;

    public Vehicle(String id) {
```

```
        this.id = id;
    }

    public String toString() {
        return id;
    }
}

class Car extends Vehicle {
    public Car(String id) {
        super(id);
    }

    public void park() {
        System.out.println("Car Parking " + id);
    }
}

class Bus extends Vehicle {
    public Bus(String id) {
        super(id);
    }

    public void park() {
        System.out.println("Bus Parking " + id);
    }
}

class Van extends Vehicle {
    public Van(String id) {
        super(id);
    }

    public void park() {
        System.out.println("Van Parking " + id);
    }
}

class VehicleQueue {
    private Vehicle[] queue;
    private int front;
```

HESHAN SANDARUWAN
200304512443

```
    private int rear;
    private int capacity;

    public VehicleQueue() {
        capacity = 10; // Initial capacity set to 10 (can be adjusted)
        queue = new Vehicle[capacity];
        front = rear = -1;
    }

    public void enQueue(Vehicle vehicle) {
        if (rear == capacity - 1) {
            System.out.println("Queue is full. Cannot enqueue.");
            return;
        }
        if (front == -1) {
            front = 0;
        }
        queue[++rear] = vehicle;
    }

    public Vehicle deQueue() {
        if (front == -1 || front > rear) {
            System.out.println("Queue is empty. Cannot dequeue.");
            return null;
        }
        Vehicle dequeued = queue[front++];
        return dequeued;
    }

    public void printVehicleQueue() {
        System.out.print("[");
        for (int i = front; i <= rear; i++) {
            System.out.print(queue[i]);
            if (i != rear) {
                System.out.print(", ");
            }
        }
        System.out.println("]");
    }

    public void callPark() {
```

HESHAN SANDARUWAN
200304512443

```
        for (int i = front; i <= rear; i++) {
            queue[i].park();
        }
    }
}
```

37.

Prints Sub : null -> Panadura