//MARAM SRIPATHI

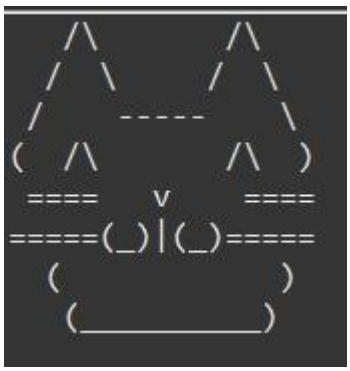
1.
```java
package section3;

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

        System.out.println("  ****  ");
        System.out.println(" *    *");
        System.out.println("* *  * *");
        System.out.println("*      *");
        System.out.println("* *  * *");
        System.out.println("*  **  *");
        System.out.println(" *    * ");
        System.out.println("  ****  ");

    }

}
```

OUTPUT:

2.

```java
package section3;

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

        System.out.println("   /\\       /\\   ");
        System.out.println("  /  \\     /  \\  ");
        System.out.println(" /    -----    \\ ");
        System.out.println("(  /\\       /\\  )");
        System.out.println(" ====   v    ====");
        System.out.println("=====(_)|(_)=====");
        System.out.println("  (           ) ");
        System.out.println("   (_________)"  );

    }

}
```

OUTPUT:



2. The Snake Box Factory Overview Dear Respectable Software Engineer, Here at the world renowned Snake Box Factory, we pride ourselves on our ability to deliver the highest quality, custom sized, cardboard boxes to our customers. Our boxes are filled with the highest quality, custom-ordered snakes. We service thousands of accounts worldwide and have a solid 98% satisfaction rating with customers. However, the entire ordering process is currently written on cardboard, which is transported between departments via carrier snake. We thought this would be a good way to show confidence in the quality and usefulness of our product. But as our business continues to grow, we're realizing this was a bad idea. We believe it's time for a more conventional and digitized approach to our operations. Would you be able to help us develop the software we need to make this happen? Sincerely, President George Johnson, The Snake Box Factory Tasks Read the scenario found in the overview and consider what objects could be modeled as part of creating a software

solution. Identify 3 objects from this scenario (remember, objects can be either tangible or abstract. List 3 properties and 3 behaviors belonging to each object. Write your solution as a document rather than a .java file

PROGRAM:

```java
class Customer {
    private String name;
    private int accountNumber;
    private double satisfactionRating;

    public Customer(String name, int accountNumber, double satisfactionRating) {
        this.name = name;
        this.accountNumber = accountNumber;
        this.satisfactionRating = satisfactionRating;
    }

    public void placeOrder(Order order) {
        System.out.println(name + " placed an order with ID: " + order.getOrderId());
    }

    public void trackOrder(Order order) {
        System.out.println("Order " + order.getOrderId() + " is currently " + order.getStatus());
    }

    public void provideFeedback(double rating) {
        this.satisfactionRating = rating;
        System.out.println(name + " provided feedback. New satisfaction rating: " + rating);
    }
}
```

```java
class Order {
    private int orderId;
    private String boxSize;
    private String snakeType;
    private String status;

    public Order(int orderId, String boxSize, String snakeType) {
        this.orderId = orderId;
        this.boxSize = boxSize;
        this.snakeType = snakeType;
        this.status = "Processing";
    }

    public int getOrderId() {
        return orderId;
    }

    public String getStatus() {
        return status;
    }

    public double calculateCost() {
        // Simplified cost calculation
        return boxSize.length() * 10 + snakeType.length() * 20;
    }

    public void updateStatus(String status) {
        this.status = status;
        System.out.println("Order " + orderId + " status updated to " + status);
    }

    public void generateInvoice() {
        System.out.println("Invoice for Order " + orderId + ":");
        System.out.println("Box Size: " + boxSize);
        System.out.println("Snake Type: " + snakeType);
        System.out.println("Total Cost: $" + calculateCost());
    }
```

```java
public int getOrderId() {
    return orderId;
}

public String getStatus() {
    return status;
}

public double calculateCost() {
    // Simplified cost calculation
    return boxSize.length() * 10 + snakeType.length() * 20;
}

public void updateStatus(String status) {
    this.status = status;
    System.out.println("Order " + orderId + " status updated to " + status);
}

public void generateInvoice() {
    System.out.println("Invoice for Order " + orderId + ":");
    System.out.println("Box Size: " + boxSize);
    System.out.println("Snake Type: " + snakeType);
    System.out.println("Total Cost: $" + calculateCost());
}
```

```java
// CarrierSnake class
class CarrierSnake {
    private int id;
    private double speed;
    private int capacity;

    public CarrierSnake(int id, double speed, int capacity) {
        this.id = id;
        this.speed = speed;
        this.capacity = capacity;
    }

    public void transportOrder(Order order) {
        System.out.println("Carrier Snake " + id + " is transporting order " + order.getOrderId());
    }

    public boolean checkAvailability() {
        // Simplified availability check
        return true;
    }

    public void reportLocation() {
        // Simplified location reporting
        System.out.println("Carrier Snake " + id + " is at the dispatch center.");
    }
}
```

```java
// Main class to demonstrate the interactions
public class SnakeBoxFactory {
    Run | Debug
    public static void main(String[] args) {
        Customer customer = new Customer(name:"Alice", accountNumber:1001, satisfactionRating:4.5);
        Order order = new Order(orderId:2001, boxSize:"Large", snakeType:"Python");
        CarrierSnake carrierSnake = new CarrierSnake(id:1, speed:10.5, capacity:5);

        customer.placeOrder(order);
        order.generateInvoice();
        carrierSnake.transportOrder(order);
        carrierSnake.reportLocation();
        order.updateStatus(status:"Shipped");
        customer.trackOrder(order);
        customer.provideFeedback(rating:4.8);
    }
}
```