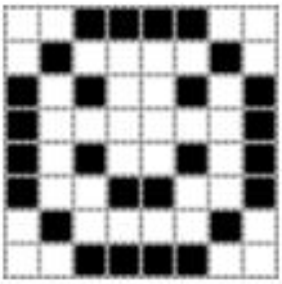
Problem 1: Simple ASCII Art



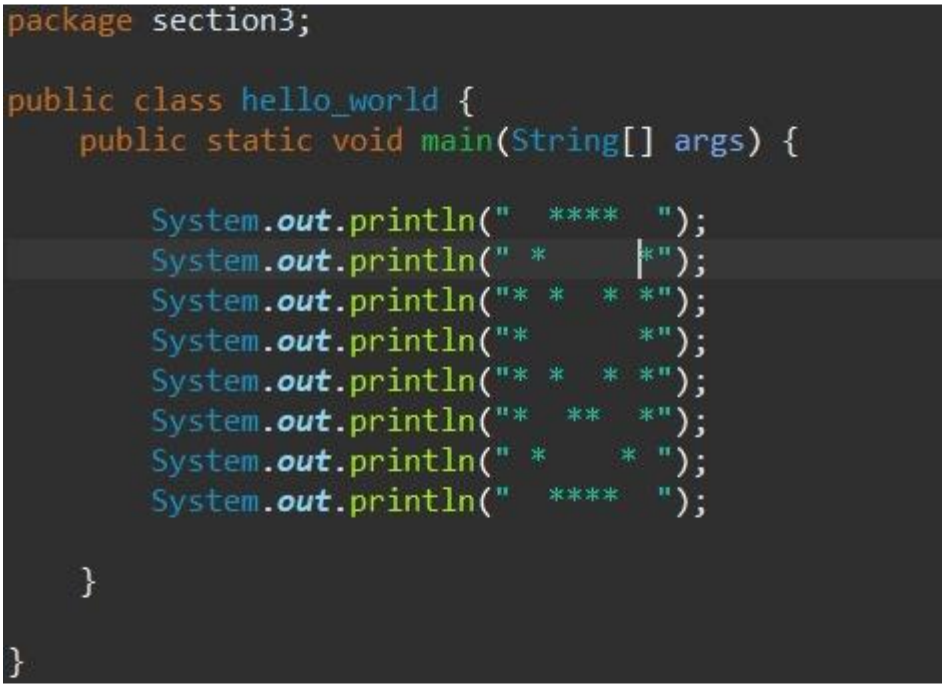
Overview

Using text to create a picture is known as ASCII art. In section 2, we made an ASCII art cat. In this practice, you'll use print statements to recreate the image above.

Task

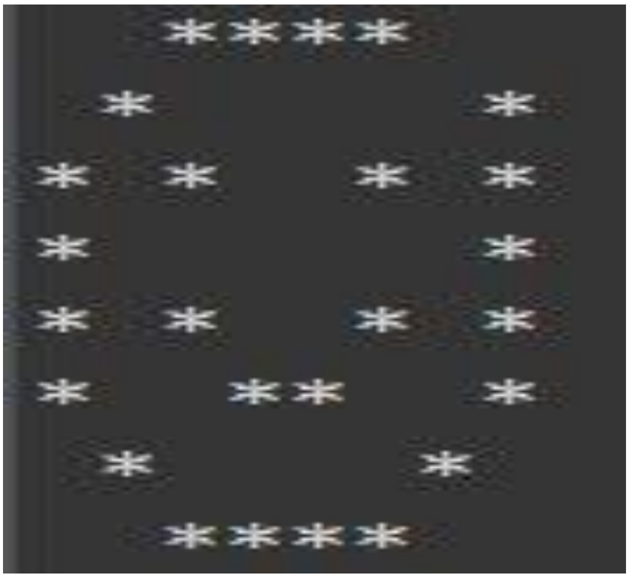
Use 8 print statements to recreate the smiley face above. Your art will rely on only a single character, besides space, such as X or #.

The ProblemSet2_1 project is available to help you get started.



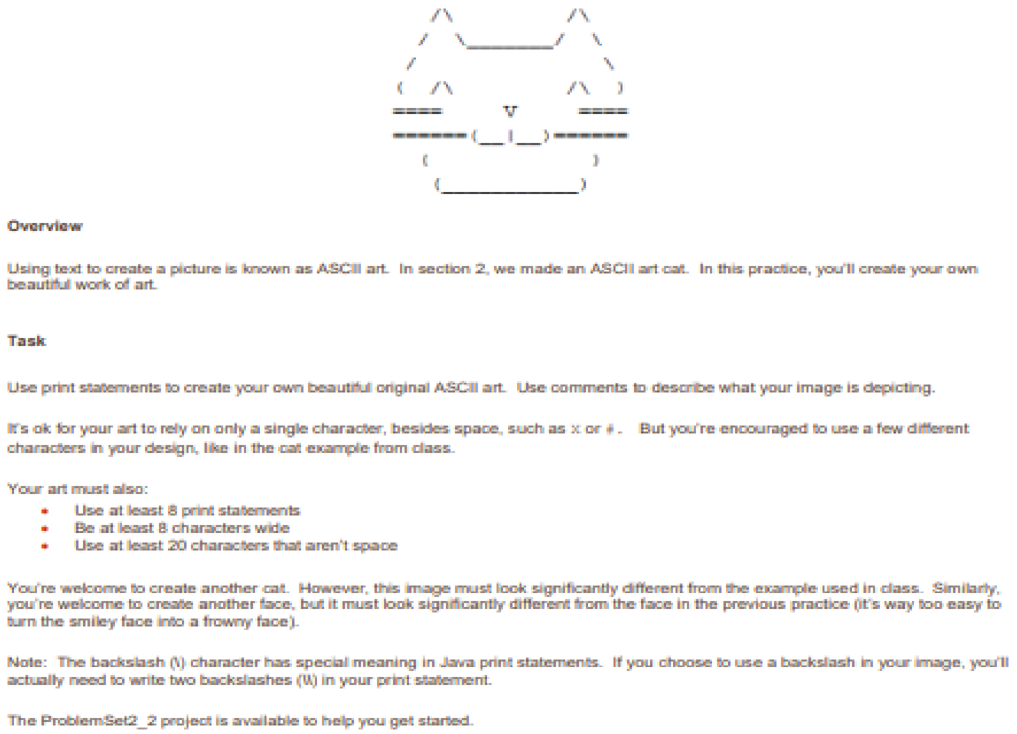
```
package section3;

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:



```
    * * * *
  *           *
*   *     *   *
*             *
*   *     *   *
*     * *     *
  *         *
    * * * *
```

Problem 2:



```
     /\           /\
    /  \_________/  \
   /                 \
  (   /\         /\   )
 ====      v      ====
 ======(___|___)======
     (             )
      (___________)
```

Overview

Using text to create a picture is known as ASCII art. In section 2, we made an ASCII art cat. In this practice, you'll create your own beautiful work of art.

Task

Use print statements to create your own beautiful original ASCII art. Use comments to describe what your image is depicting.

It's ok for your art to rely on only a single character, besides space, such as x or #. But you're encouraged to use a few different characters in your design, like in the cat example from class.

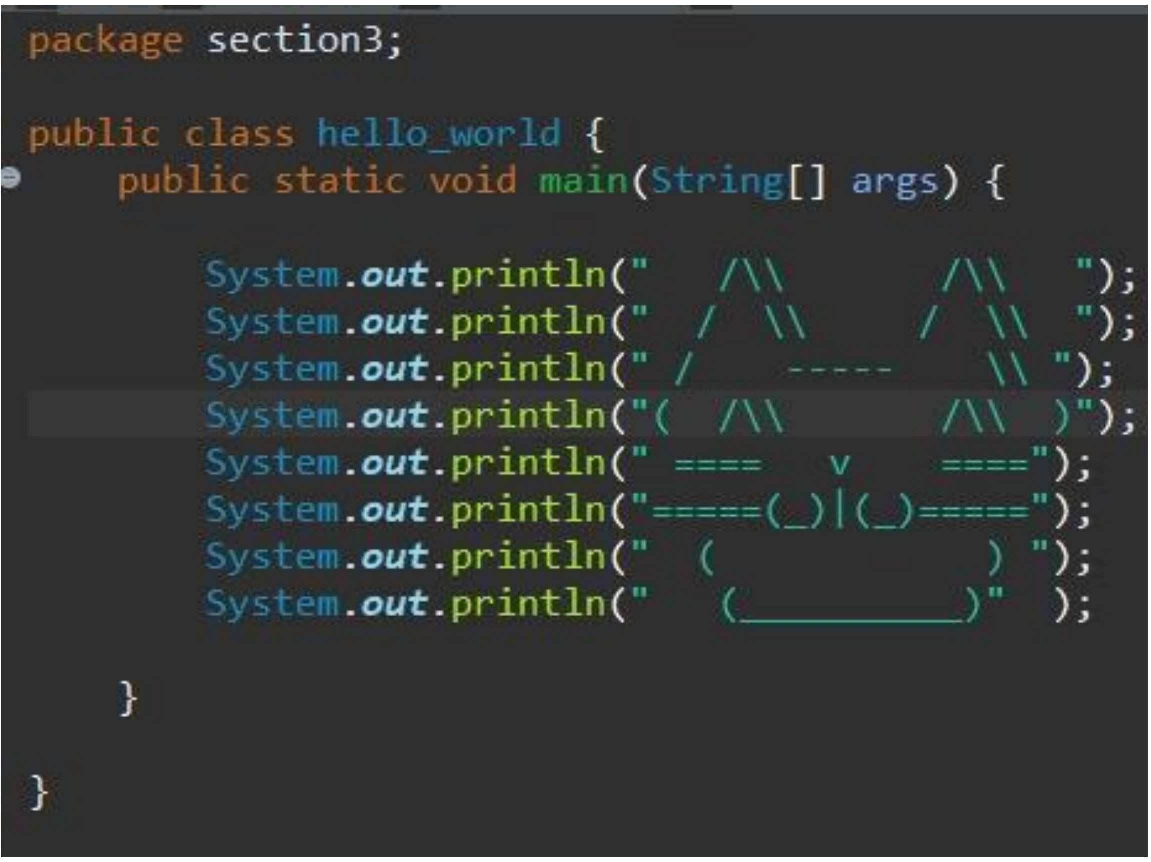
Your art must also:

- Use at least 8 print statements
- Be at least 8 characters wide
- Use at least 20 characters that aren't space

You're welcome to create another cat. However, this image must look significantly different from the example used in class. Similarly, you're welcome to create another face, but it must look significantly different from the face in the previous practice (it's way too easy to turn the smiley face into a frowny face).

Note: The backslash (\) character has special meaning in Java print statements. If you choose to use a backslash in your image, you'll actually need to write two backslashes (\\) in your print statement.

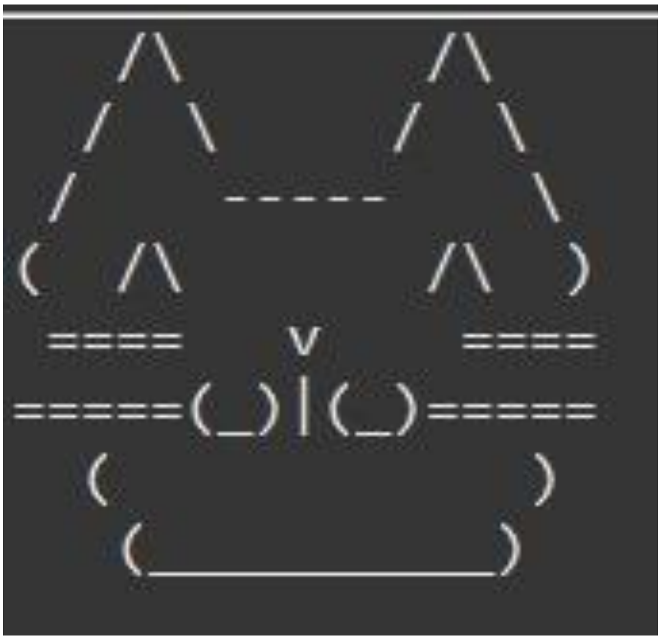
The ProblemSet2_2 project is available to help you get started.



```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("   (__________)"  );

    }

}
```

t

Output:



```
   /\       /\
  /  \     /  \
 /    -----    \
(  /\       /\  )
 ====   v    ====
=====(_)|(_)=====
  (            )
   (__________)
```

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?

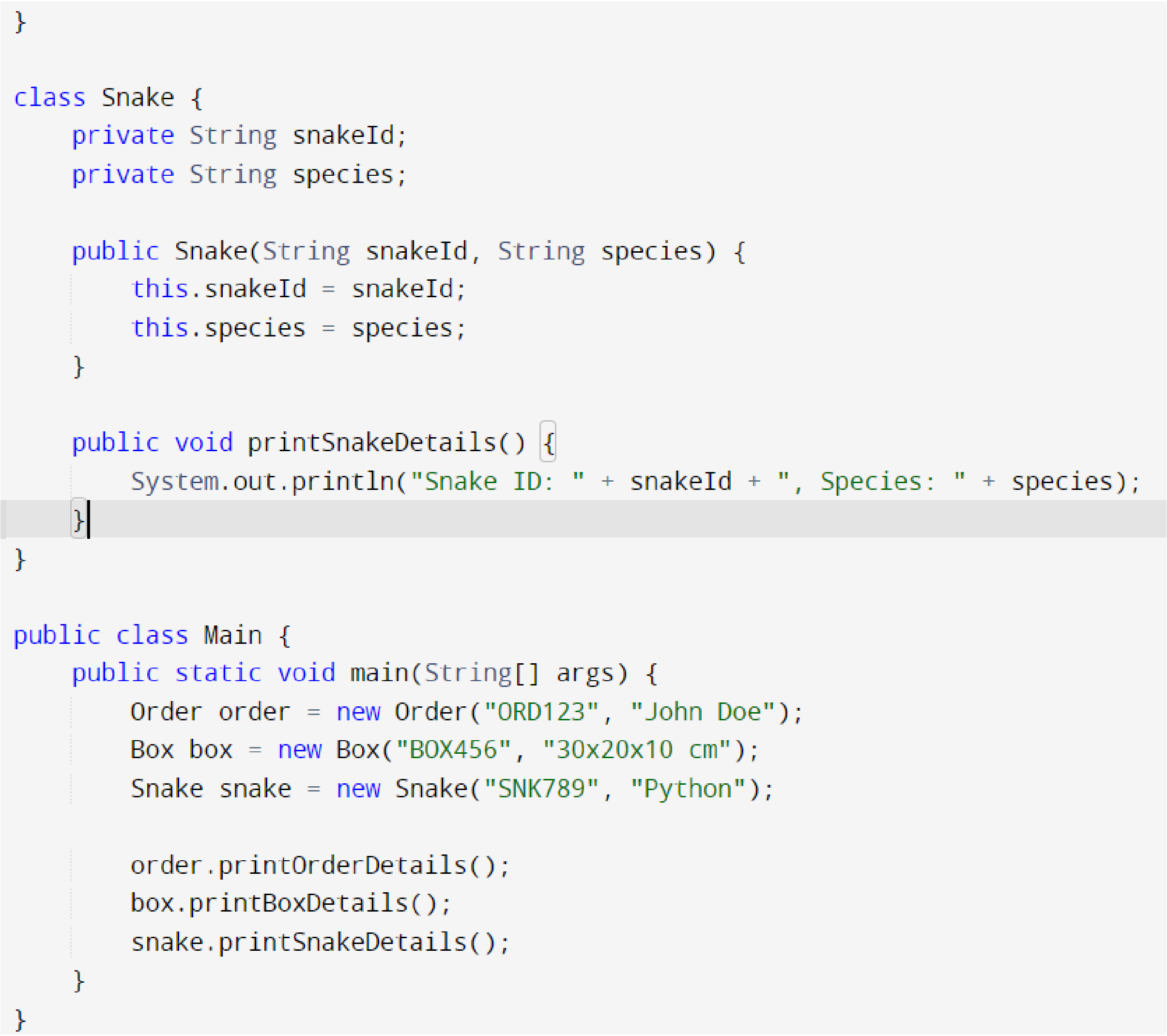
Sincerely,
President George Johnson,
The Snake Box Factory

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.

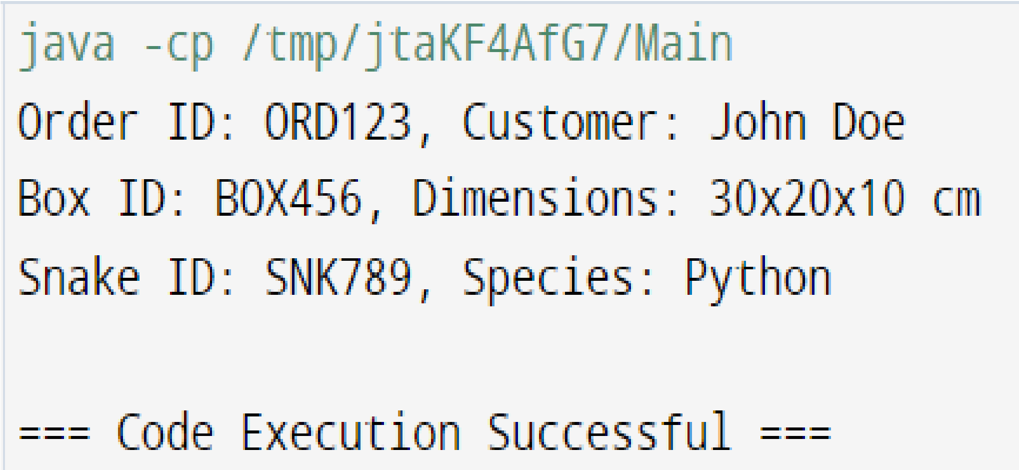
Write your solution as a document rather than a .java file.

```java
import java.time.LocalDate;

class Order {
    private String orderId;
    private String customerName;

    public Order(String orderId, String customerName) {
        this.orderId = orderId;
        this.customerName = customerName;
    }

    public void printOrderDetails() {
        System.out.println("Order ID: " + orderId + ", Customer: " +
            customerName);
    }
}

class Box {
    private String boxId;
    private String dimensions;

    public Box(String boxId, String dimensions) {
        this.boxId = boxId;
        this.dimensions = dimensions;
    }

    public void printBoxDetails() {
        System.out.println("Box ID: " + boxId + ", Dimensions: " + dimensions);
    }
}
```



```java
}

class Snake {
    private String snakeId;
    private String species;

    public Snake(String snakeId, String species) {
        this.snakeId = snakeId;
        this.species = species;
    }

    public void printSnakeDetails() {
        System.out.println("Snake ID: " + snakeId + ", Species: " + species);
    }
}

public class Main {
    public static void main(String[] args) {
        Order order = new Order("ORD123", "John Doe");
        Box box = new Box("BOX456", "30x20x10 cm");
        Snake snake = new Snake("SNK789", "Python");

        order.printOrderDetails();
        box.printBoxDetails();
        snake.printSnakeDetails();
    }
}
```

Output :



```
java -cp /tmp/jtaKF4AfG7/Main
Order ID: ORD123, Customer: John Doe
Box ID: BOX456, Dimensions: 30x20x10 cm
Snake ID: SNK789, Species: Python


=== Code Execution Successful ===
```