CSE 201
Java Programming I

Smart Coding School
website: http://www.smartcodingschool.com

Static Method

Definition: A named group of statements

```
public static void methodName() {
    … statements …
}
```

Static Method

Usage: 1. denotes the structure of a program
2. eliminates redundancy by code reuse

```
public static void methodName() {
     … statements …
}
```

Static Method

Pound Cake

1. **With a mixer, cream butter and shortening together.**
2. **Add sugar, a little at a time.**
3. **Add eggs, 1 at a time, beating after each addition.**
4. **Stir dry ingredients together in a bowl and add to mixer alternately with milk, starting with the flour and ending with the flour.**
5. **Mix in vanilla.**
6. **Pour into a greased and floured tube pan and bake for 1 to 1**
7. **1/2 hours, until a toothpick inserted in the center of the cake comes out clean.**

Static Method

Make it more structural

Pound Cake

Preparation:

1. With a mixer, cream butter and shortening together.
2. Add sugar, a little at a time.
3. Add eggs, 1 at a time, beating after each addition.

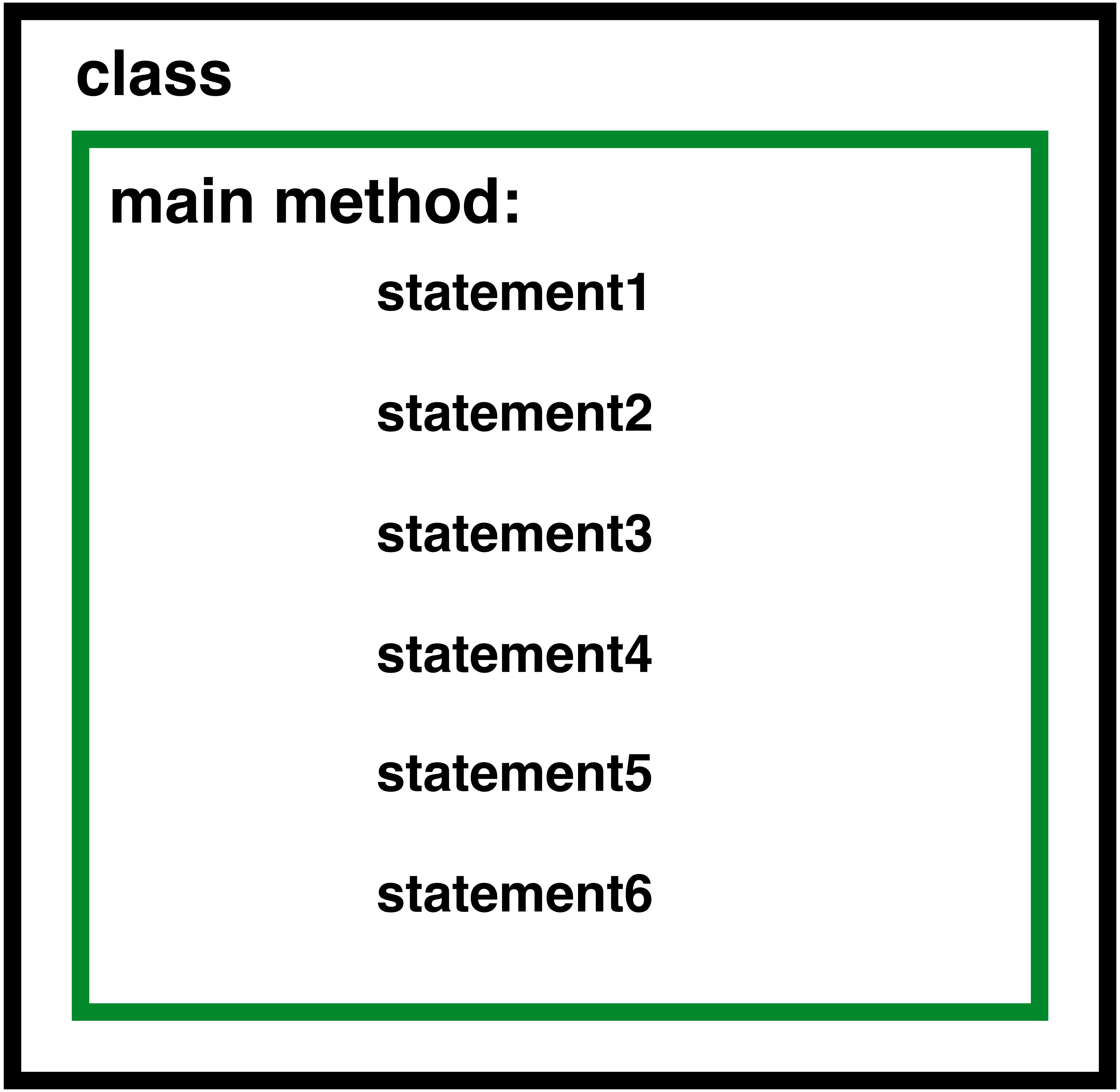
Mix Step:

4. Stir dry ingredients together in a bowl and add to mixer alternately with milk, starting with the flour and ending with the flour.
5. Mix in vanilla.
6. Pour into a greased and floured tube pan and bake for 1 to 1

Bake Step:

7. 1/2 hours, until a toothpick inserted in the center of the cake comes out clean.

Static Method



class

main method:

statement1

statement2

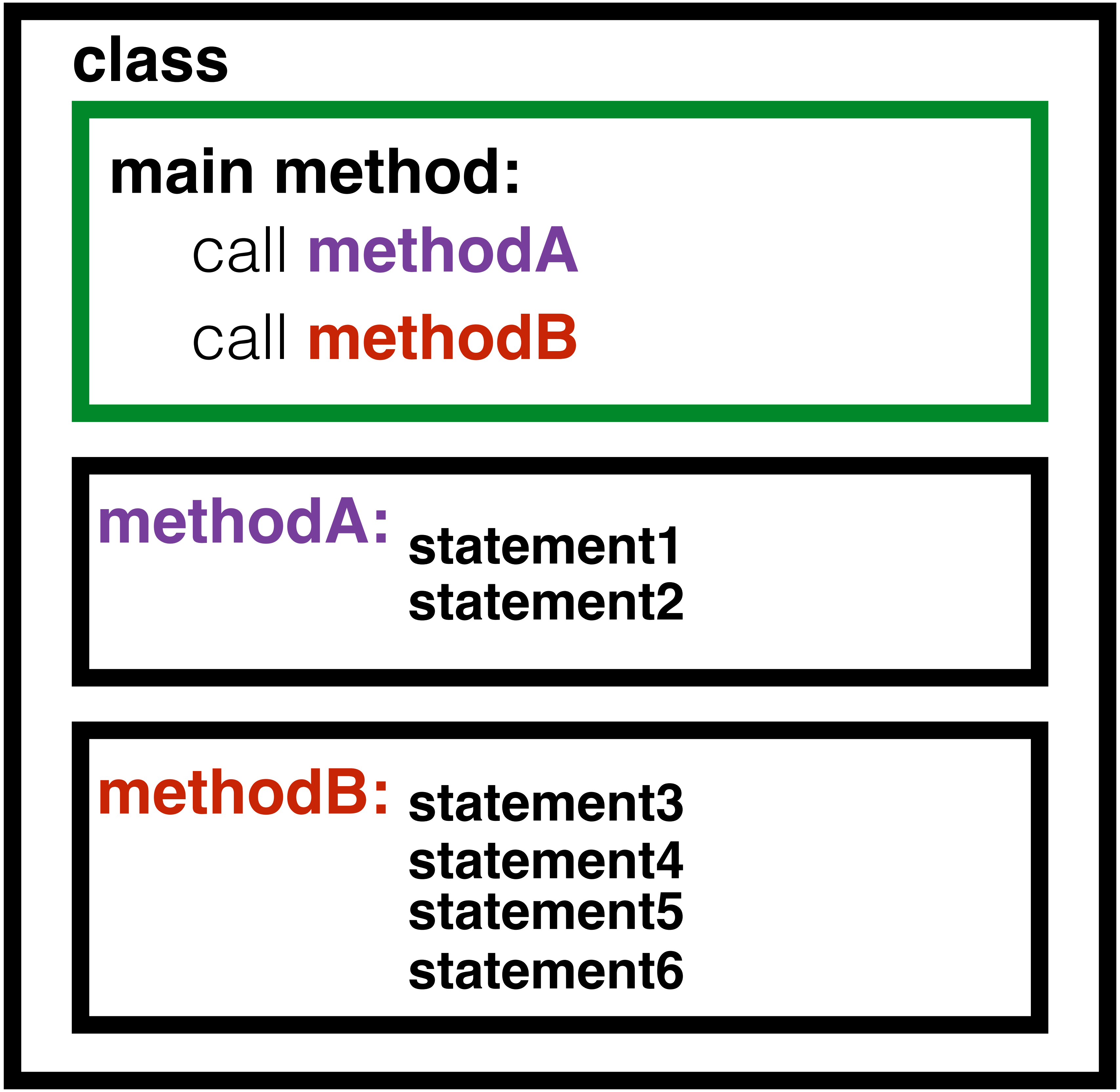
statement3

statement4

statement5

statement6

Static Method



class

main method:

call methodA

call methodB

methodA: statement1

statement2

methodB: statement3

statement4

statement5

statement6

Static Method

Syntax:

```
public static void methodName() {
    … statements …
}
```

Example:

```
public static void printMessage() {
    System.out.println(“This is the error”);
    System.out.println(“in the lab.”);
}
```

Static Method

```
class Example {
    public static void main(String[] args) {
        printMessage();
        printHello();
    }

    public static void printMessage() {
        System.out.println(“This is the error”);
        System.out.println(“in the lab.”);
    }

    public static void printHello() {
        System.out.println(“Hello”);
        System.out.println(“World!”);
    }
}
```

Static Method

You can call the same static method
as many times as you want

```
class Example {
    public static void main(String[] args) {
        printMessage();
        printMessage();
        printMessage();
    }
    public static void printMessage() {
        System.out.println(“This is the error”);
        System.out.println(“in the lab.”);
    }
}
```

Static Method

a method can call other static method

```
class Example {
     public static void main(String[] args) {
          printMessage();
     }

     public static void printMessage() {
          System.out.println(“This is the error”);
          printHello();
     }

     public static void printHello() {
          System.out.println(“Hello”);
          System.out.println(“Hello”)
     }
}
```

Static Method

Static Method Principle

Place statements into a static method if:

1. The statements are related structurally
2. The statements are repeated

Don’t create a static method if:

1. A single print or println statement
2. Blank lines like System.out.println();
   (Put them in main method)
3. Unrelated or weakly related statements
   (splitting them into 2 smaller methods)

Static Method

Static Method Question:

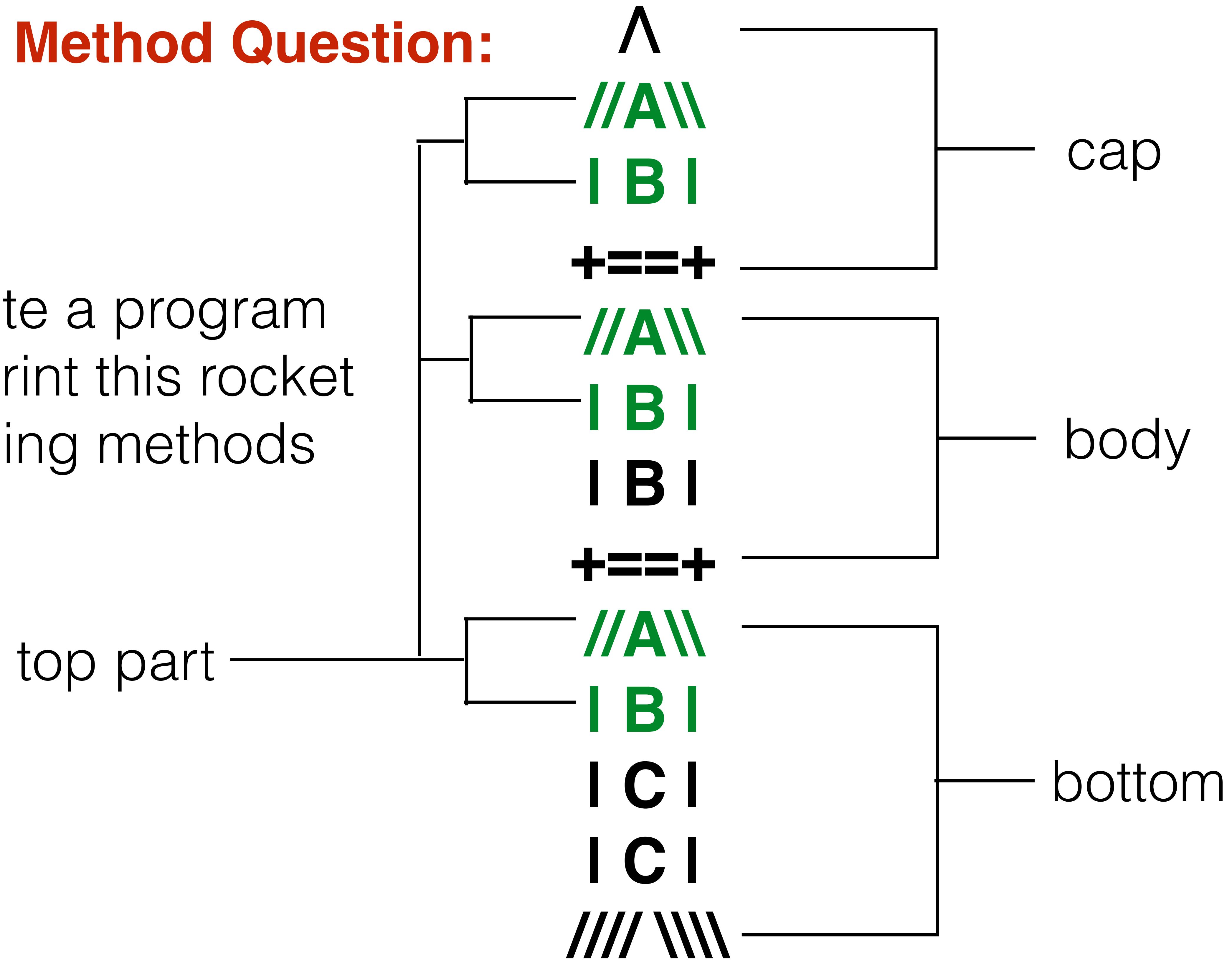
write a program
to print this rocket
using methods

```
 ^
//A\\
| B |
+==+
//A\\
| B |
| B |
+==+
//A\\
| B |
| C |
| C |
//// \\\\
```

Static Method

Static Method Question:

write a program
to print this rocket
using methods



```
 ^
//A\\
| B |
+==+
//A\\
| B |
| B |
+==+
//A\\
| B |
| C |
| C |
//// \\\\
```

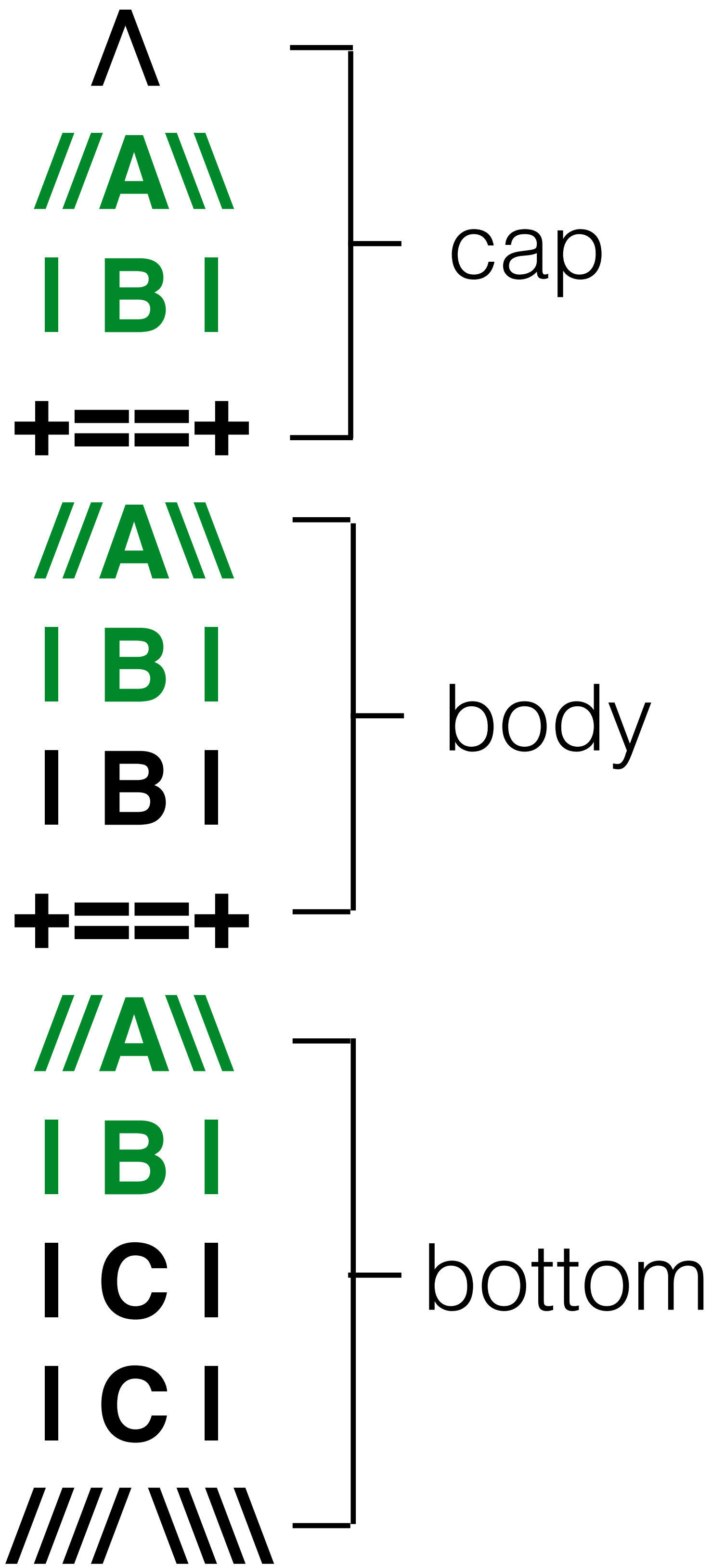
cap

body

bottom

top part

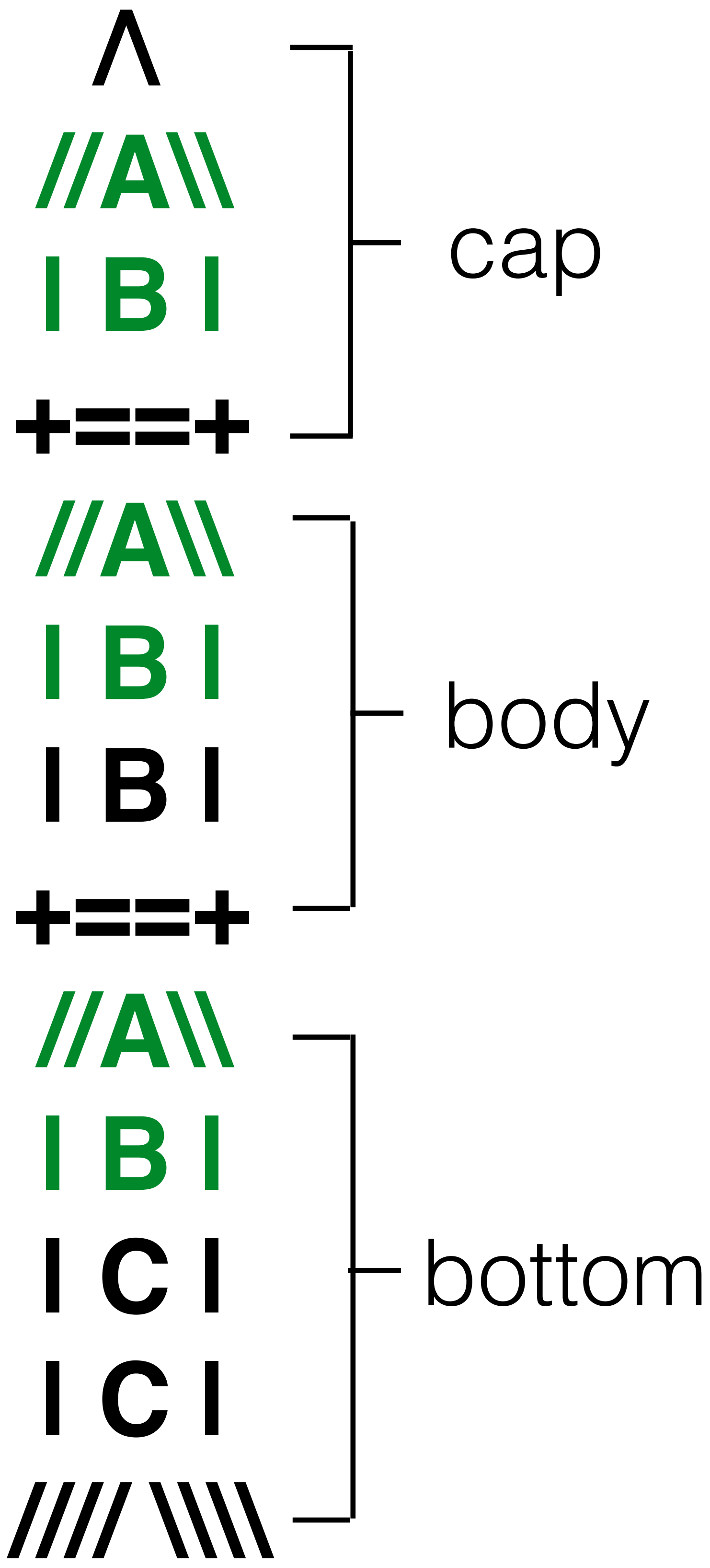
Static Method



```
  Λ
//A\\
I B I
+==+
//A\\
I B I
I B I
+==+
//A\\
I B I
I C I
I C I
//// \\\\
```

```
public static void topPart() {
        System.out.println(“ //A\\”);
        System.out.println(“ I B I”);
}
```

Static Method

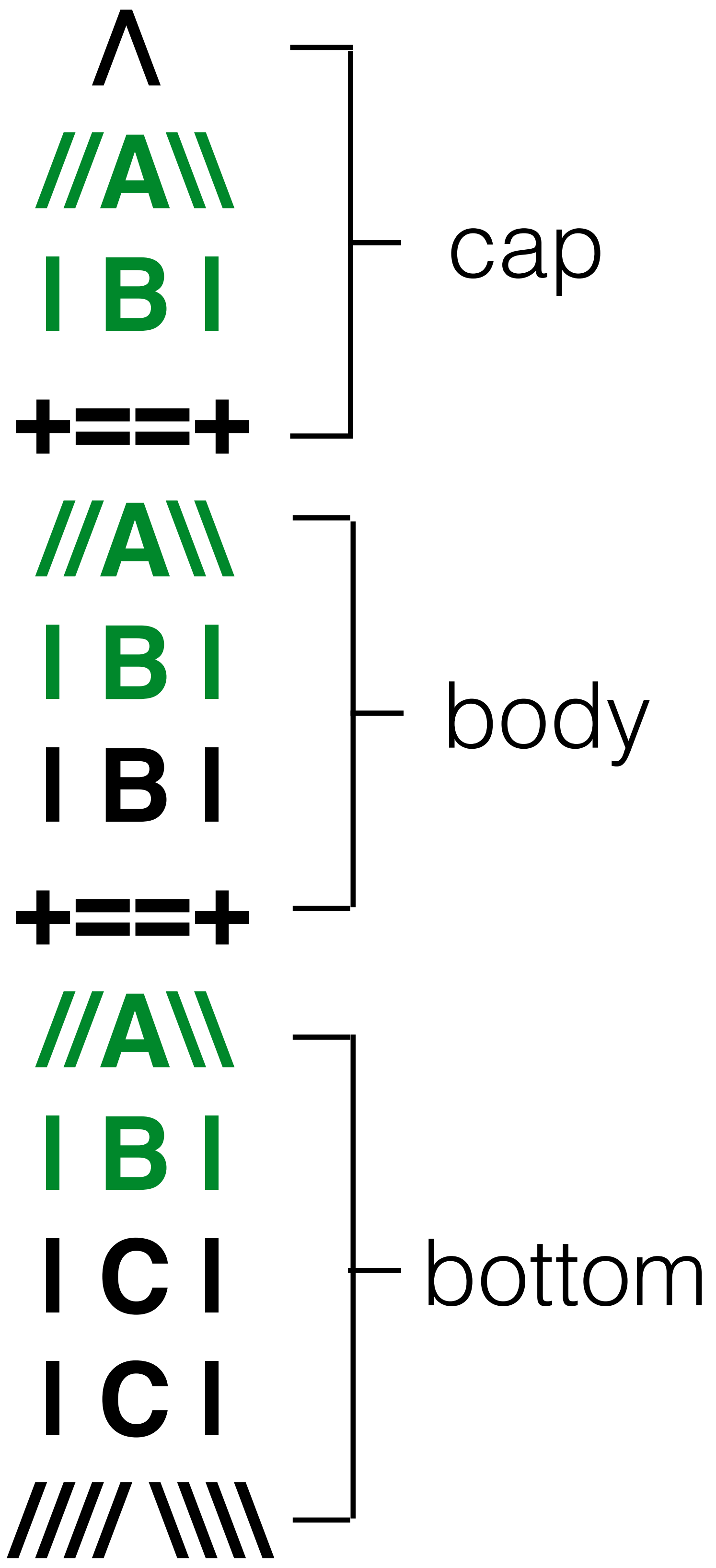


```
 /\
//A\\
| B |
+==+
//A\\
| B |
| B |
+==+
//A\\
| B |
| C |
| C |
//// \\\\
```

```
public static void topPart() {
        System.out.println(“ //A\\\\”);
        System.out.println(“ | B |”);
}
```

```
public static void cap() {
        System.out.println(“  /\\”);
        topPart();
        System.out.println(“+==+”);
}
```

Static Method



```
 Λ
//A\\
| B |
+==+
//A\\
| B |
| B |
+==+
//A\\
| B |
| C |
| C |
/// \\\
```

cap

body

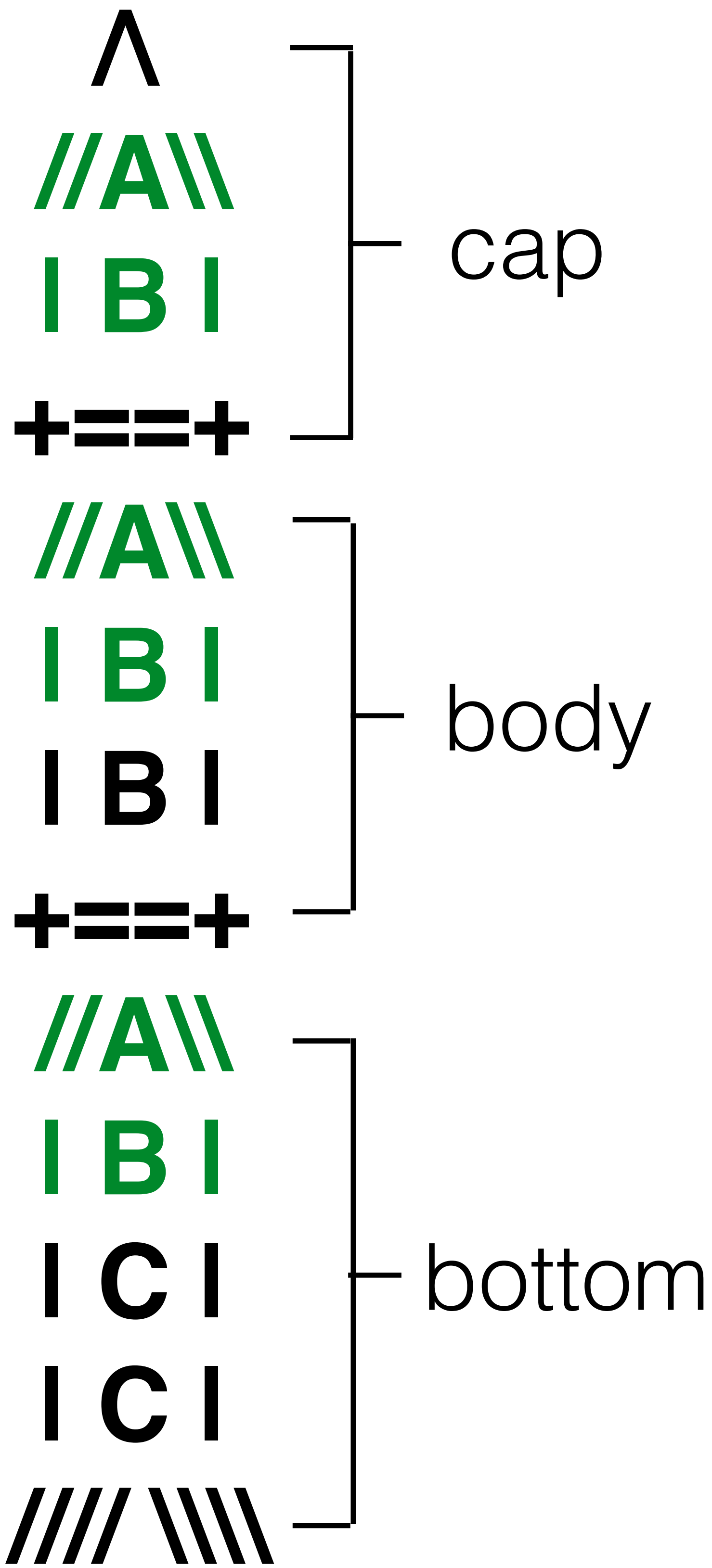
bottom

```
public static void topPart() {
          System.out.println(“ //A\\\\”);
          System.out.println(“ | B |”);
}

public static void cap() {
          System.out.println(“  /\\”);
          topPart();
          System.out.println(“+==+”);
}
```

```
public static void body() {
          topPart();
          System.out.println(“ | B |”);
          System.out.println(“+==+”);
}
```

Static Method



```
  /\
 //A\\
 | B |
 +==+
 //A\\
 | B |
 | B |
 +==+
 //A\\
 | B |
 | C |
 | C |
//// \\\\
```

cap

body

bottom

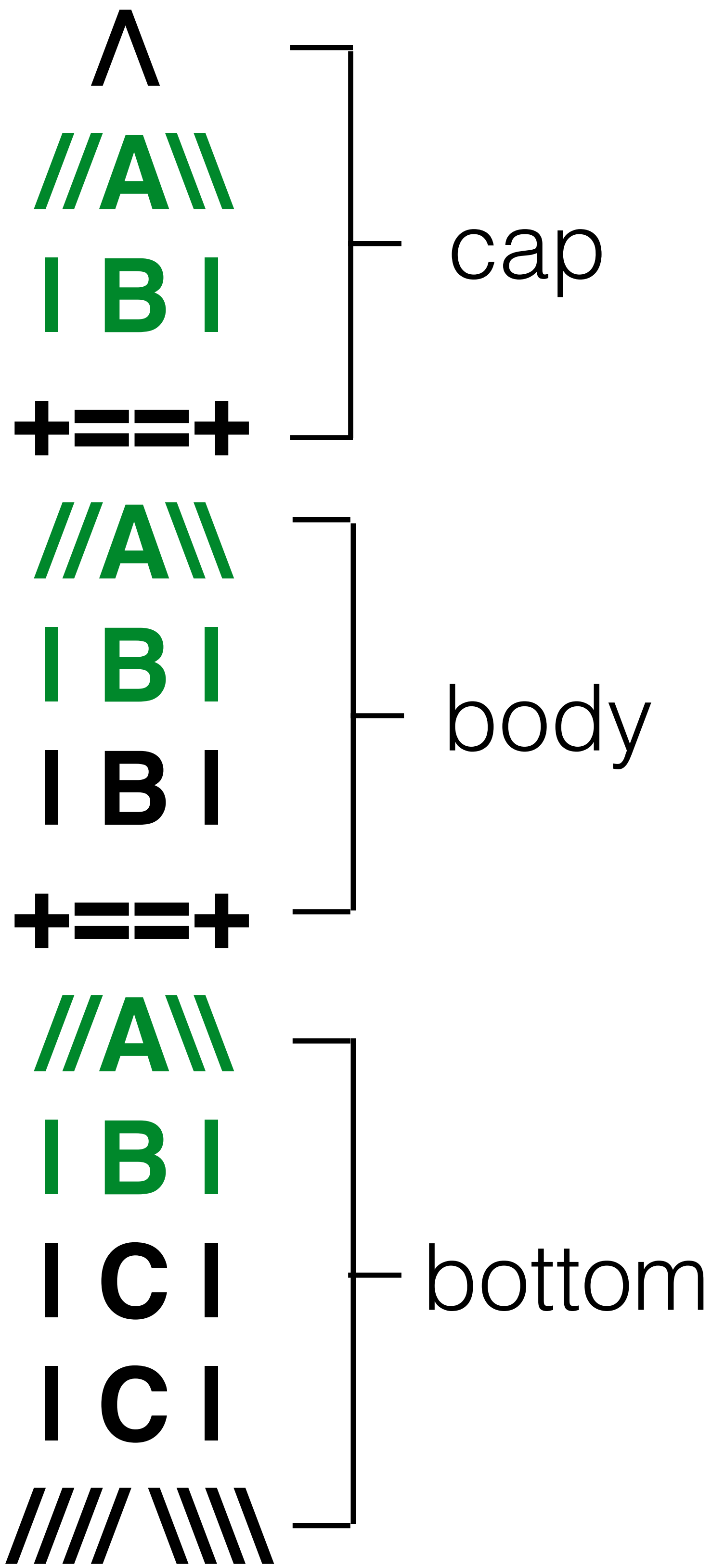
```
public static void topPart() {
        System.out.println(" //A\\\\");
        System.out.println(" | B |");
}

public static void cap() {
        System.out.println("  /\\");
        topPart();
        System.out.println("+==+");
}

public static void body() {
        topPart();
        System.out.println(" | B |");
        System.out.println("+==+");
}
```

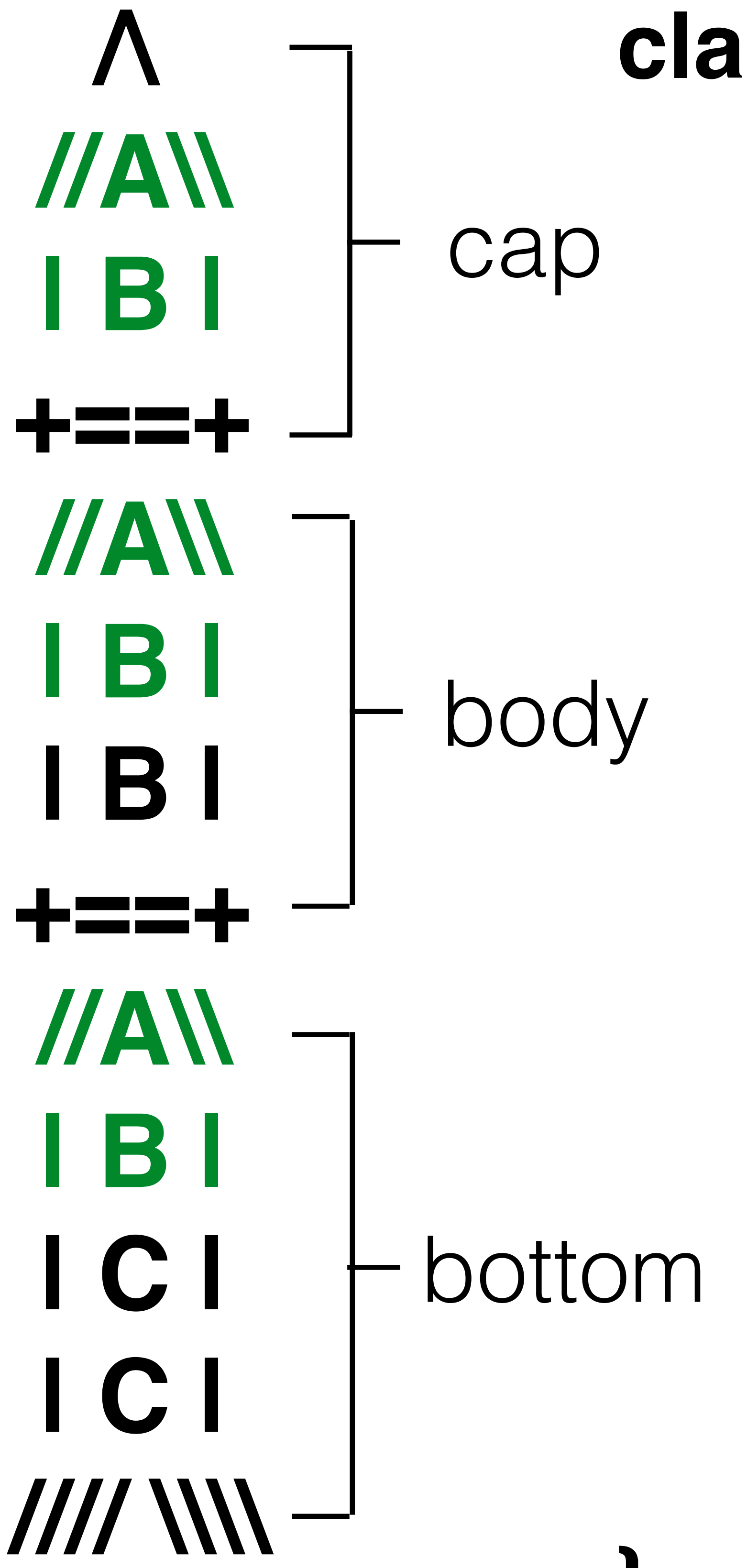
```
public static void bottom() {
        topPart();
        System.out.println(" | C |");
        System.out.println(" | C |");
        System.out.println("////\\\\\\\");
}
```

Static Method



```
public static void topPart() {
        System.out.println(" //A\\\\");
        System.out.println(" | B |");
}
public static void cap() {
        System.out.println("  Λ\");
        topPart();
        System.out.println("+==+");
}
public static void body() {
        topPart();
        System.out.println(" | B |");
        System.out.println("+==+");
}
public static void bottom() {
        topPart();
        System.out.println(" | C |");
        System.out.println(" | C |");
        System.out.println("///Λ\\\\\");
}
```

Static Method



```
 ^
//A\\
| B |
+==+
//A\\
| B |
| B |
+==+
//A\\
| B |
| C |
| C |
//// \\\\
```

```
class Example {
    public static void main(String[] args) {
        cap();
        body();
        bottom();
    }
    public static void topPart() {
        ...
    }
    public static void cap() {
        ...
    }
    public static void body() {
        ...
    }
    public static void bottom() {
        ...
    }
}
```

Now write the code **without** redundancy