



# Branching Statements

Forest Agostinelli

University of South Carolina

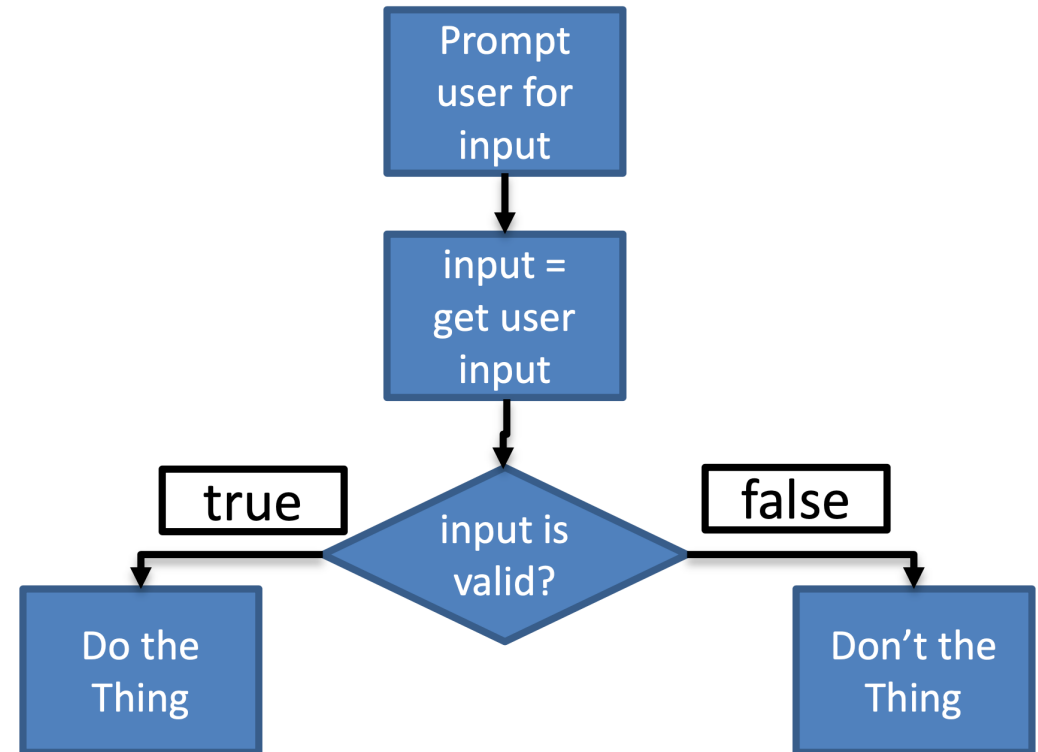
# Outline

- If/else statements
- Boolean expressions
- Nested statements
- Else-if statements
- Example programs

# Flow Control

- Flow of control is the order in which a program performs actions.
- A **branching statement** chooses between two or more possible actions.
- A **loop statement** repeats an action until a stopping condition occurs.
- Flow Charts diagram the flow of a program
  - Boxes are Statements
  - Diamonds are Decisions
    - True branch
    - False branch
  - Arrows indicate the flow of statements and decisions
  - Pseudocode is mostly used

Flow Chart Example



# If Statements

- If-statement
- If the Boolean expression is “true” then the body of the if-statement is executed, and otherwise is ignored
- Putting curly braces “{}” to denote the body of the if-statement is strongly encouraged
- Do not put a semicolon “;” after the parenthesis
  - It will ignore the Boolean expression
- Spoken
  - “if this is true then do this”

## Syntax

```
if(<<Boolean expression>>)  
{  
    //Body of the if-statement  
}  
//Outside Body of the if-statement
```

## Examples

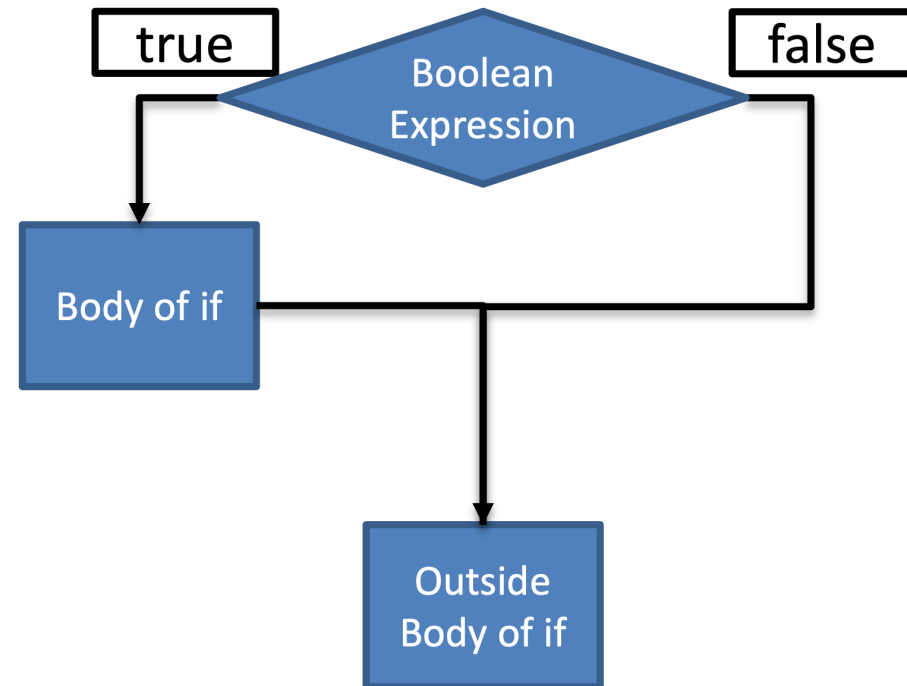
```
if(a == b)  
{  
    System.out.println(“a is equal to b”);  
}
```

# If Statements

## Syntax

```
if(<<Boolean expression>>)  
{  
    //Body of the if-statement  
}  
//Outside Body of the if-statement
```

## General If-Statement Flow Chart



# If/else Statements

- Else-statement
- Requires a preceding if-statement
  - If-statements do not require an else-statement
- If the Boolean expression is “false” then the body of the else-statement is executed, and otherwise is ignored
- Putting curly braces “{}” to denote the body of the else-statement is strongly encouraged
- Spoken:
  - “if this is true then do this, otherwise (else) do that”

## Syntax

```
if(<<Boolean expression>>)  
{  
    //Body of the if-statement  
}  
else  
{  
    //Body of the else-statement  
}
```

## Examples

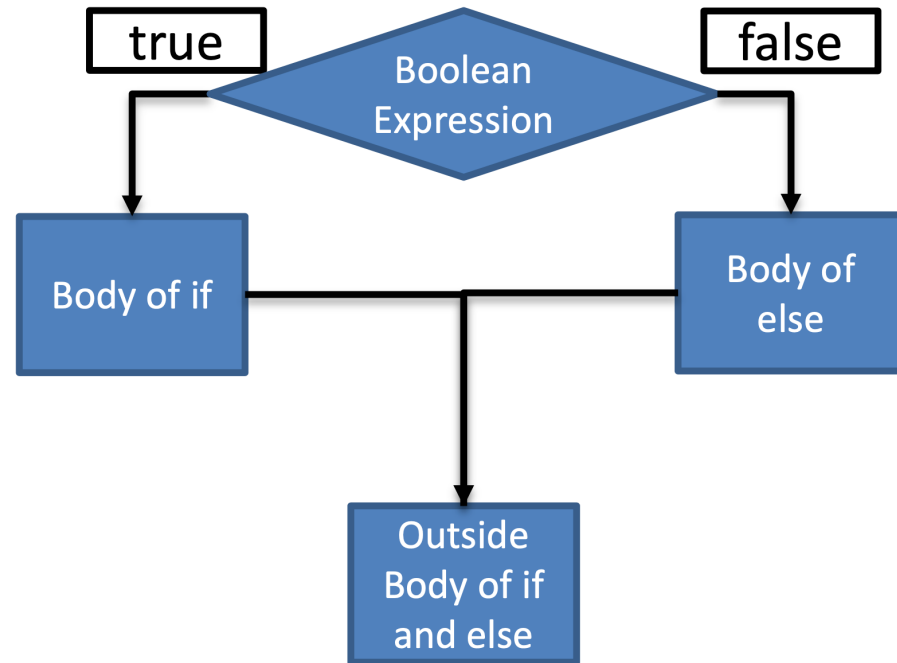
```
if(a == b)  
{  
    System.out.println(“a is equal to b”);  
}  
else  
{  
    System.out.println(“a is not equal to b”);  
}
```

# If/else Statements

## Syntax

```
if(<<Boolean expression>>)  
{  
    //Body of the if-statement  
}  
else  
{  
    //Body of the else-statement  
}  
//Outside of if and else
```

## General If-Else-Statement Flow Chart



# Conditional Operator

- The “?” and the “:” together are called the conditional operator
  - AKA ternary operator
  - AKA “One-line if”
- The value after the “?” corresponds to the body of the if-statement
- The value after the “:” corresponds to the body of the else-statement
- Can be useful for writing print statements

## Syntax

```
<<var>> = (<<Boolean expression>>)?<<value 1>>:<<value 2>>;
```

## Example

```
maxValue = (value1 >= value2) ? Value1 : value2;
```

```
System.out.println("a is " + ((a >= 0) ? "positive" : "negative"));
```



# Outline

- If/else statements
- Boolean expressions
- Nested statements
- Else-if statements
- Example programs

# Boolean Expressions

- True or False Value
- Common Boolean Operators
  - “==” : Equal to
  - “!=” : Not Equal
  - “<” : strictly less than
  - “>” : strictly greater than
  - “<=” : less than or equal to
  - “>=” : greater than or equal to

## Syntax

```
<<value>> <<Boolean operator>> <<value>>;
```

## Examples

```
boolean a = 12 > 3;  
if(a)//Or a == true  
{  
    System.out.println("Here");  
}  
else  
{  
    System.out.println("Not here");  
}
```

# Compound Boolean Expressions

- Combines multiple Boolean expressions
- Common Compound Boolean Expression Operators
  - “&&” : AND – both must be true to yield true
  - “||” : OR – only one must be true to yield true

## Syntax

```
<<Boolean expression>> <<operator>> <<Boolean expression>>;
```

## Examples

```
boolean a = 2 != 0 && 12 > 3;  
if(a)//Or a == true  
{  
    System.out.println("Here");  
}  
else  
{  
    System.out.println("Not here");  
}
```

# Truth Table

A	B	A && B	A    B
TRUE	TRUE	TRUE	TRUE
TRUE	FALSE	FALSE	TRUE
FALSE	TRUE	FALSE	TRUE
FALSE	FALSE	FALSE	FALSE

# Negating Boolean Expressions

- The not operator is “!” and is used to negate the value of a Boolean expression

Negated Expression	Equivalent Expression
!(A < B)	A >= B
!(A <= B)	A > B
!(A > B)	A <= B
!(A >= B)	A < B
!(A == B)	A != B
!(A != B)	A == B
!(A && B)	!A    !B
!(A    B)	!A && !B

De Morgan's laws

# Using “==” in Practice

- |   |  |
|---|--|
| <ul style="list-style-type: none"><li>• The operator “==” is great for determining if two values are equal in some cases, but not all</li><li>• Great to use when comparing integer values</li><li>• Not great to use when comparing floating-point values<ul style="list-style-type: none"><li>– Round of errors</li><li>– Use a combination of <math>\geq</math> and <math>\leq</math> with a tolerance</li></ul></li></ul> | <ul style="list-style-type: none"><li>• Great for comparing memory addresses of Objects<ul style="list-style-type: none"><li>– Check if objects are NULL</li><li>– Check if two identifiers reference the same place in memory</li></ul></li><li>• Not great for comparing contents of an object<ul style="list-style-type: none"><li>– Use the “.equals()” method instead</li></ul></li></ul> |
|---|--|

# Outline

- If/else statements
- Boolean expressions
- **Nested statements**
- Else-if statements
- Example programs

# Nested Statements

- Some statements can be nested within the body of another statement
- Nested if-statements are if-statements within the body of another if-statement
  - Same is true with if-else statements
  - Else statements still must have a corresponding if-statement
- Very useful when testing a combination of conditions

## Syntax

```
if(<<Boolean expression>>)  
{  
    if(<<Boolean expression>>)  
    {  
        ...  
    }  
}
```

## Example

```
if(a == b)  
{  
    System.out.println("a is equal to b");  
    if(b == c)  
    {  
        System.out.println("b is equal to c");  
    }  
    else  
    {  
        System.out.println("b is not equal to c");  
    }  
}
```

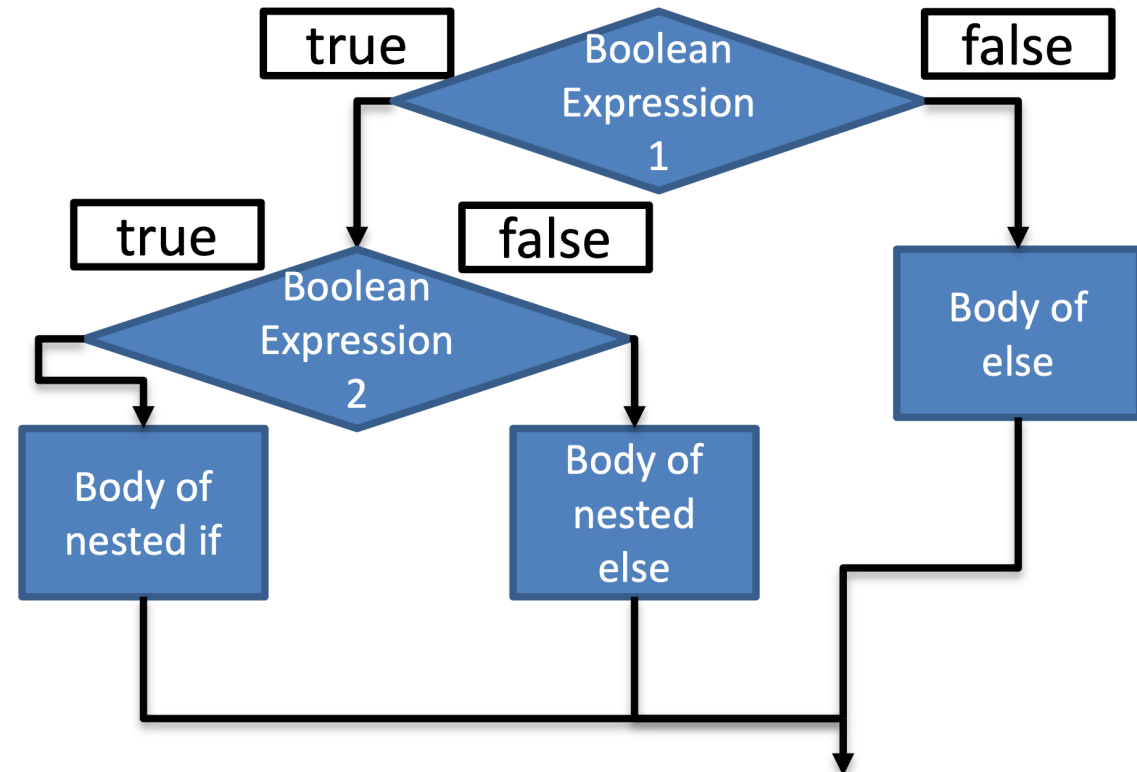


# Nested Statements

## Syntax

```
if(<<Boolean expression 1>>)  
{  
    if(<<Boolean expression 2>>)  
    {  
        //Body of nested if  
    }  
    else  
    {  
        //Body of nested else  
    }  
}  
else  
{  
    //Body of else  
}
```

## General Nested If-Statement Flow Chart



# Nested Statements

- You may omit the curly braces whenever an if-statement or else-statement has exactly ONE statement in its body
- An if-else is considered as one statement in Java
- This may cause logic errors if not careful
- It's a good idea to put the curly braces to clearly define the body of the statements
- The 2 examples do not have the same logic

```
if(a == b)
{
    if(b == c)
        e = f;
}
else
    e = g;
```

else applies to outer if

```
if(a == b)
    if(b == c)
        e = f;
else
    e = g;
```

else applies to inner if

# Outline

- If/else statements
- Boolean expressions
- Nested statements
- Else-if statements
- Example programs

# Else-if Statement

- The else-if statement is a shorthand for an if-statement in the body of the else of another if-statement
- Else-if statements require first an if-statement
  - Just like an else-statement
- Can have multiple else-if statements in succession
- For the statements in an else-if to run the previous if's or else-if's must be false
  - Else-if's should only be used when the conditions are dependent

## Syntax

```
if(<<Boolean expression>>)  
{  
    //Body of if-statement  
}  
else if(<<Boolean expression>>)  
{  
    //Body of else-if statement  
}  
else  
{  
    //Body of else statement  
}
```

## Example

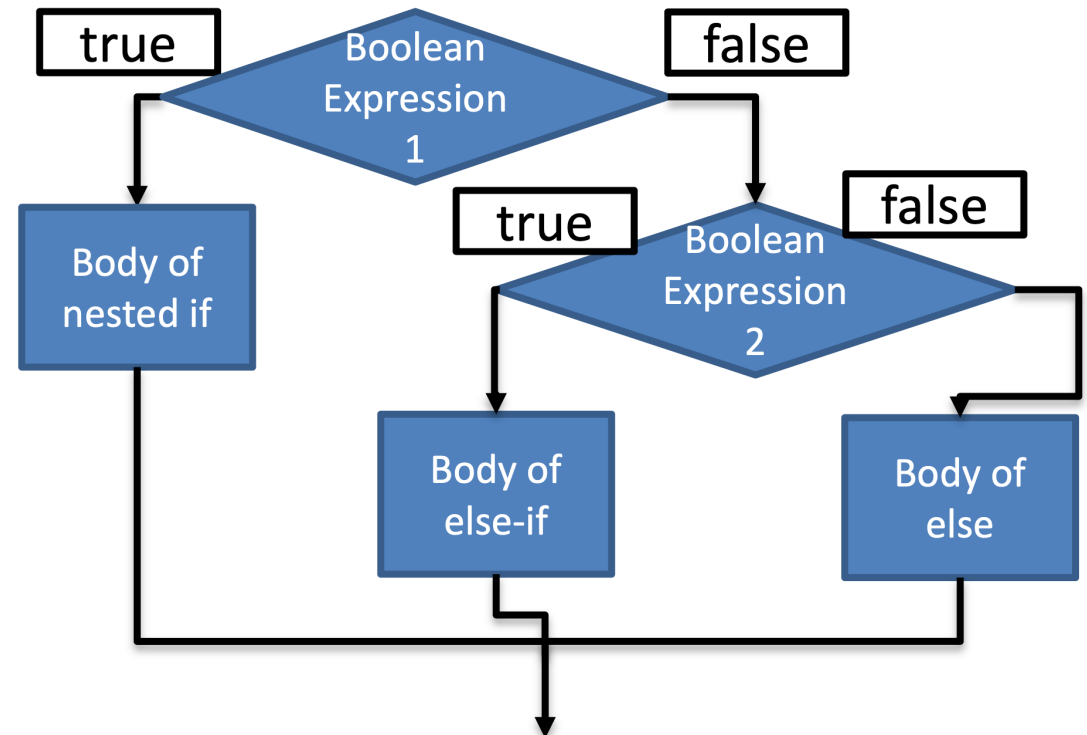
```
if(a == b)  
{  
    System.out.println("a is equal to b");  
}  
else if(a == c)  
{  
    System.out.println("a is equal to c, and not b");  
}  
else  
{  
    System.out.println("a is not equal to b or c");  
}
```

# Else-if Statement

## Syntax

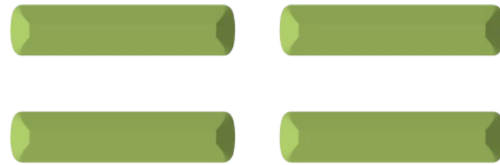
```
if(<<Boolean expression 1>>)  
{  
    //Body of if  
}  
else if(<<Boolean expression 2>>)  
{  
    //Body of else-if  
}  
else  
{  
    //Body of else  
}
```

## General Nested Else-If-Statement Flow Chart



# Else-if Logic

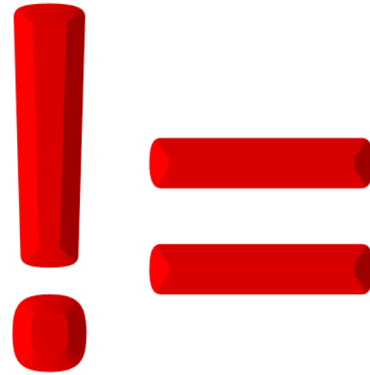
```
if (Boolean_Expression_1)
{
    Statement_1
}
else if (Boolean_Expression_2)
{
    Statement_2
}
else if (Boolean_Expression_3)
{
    Statement_3
}
else
{
    Default_Statement
}
```



```
if (Boolean_Expression_1)
{
    Statement_1
}
else
{
    if (Boolean_Expression_2)
    {
        Statement_2
    }
    else
    {
        if(Boolean_Expression_3)
        {
            Statement_3
        }
        else
        {
            Default_Statement
        }
    }
}
```

# Else-if Logic

```
if (Boolean_Expression_1)
{
    Statement_1
}
else if (Boolean_Expression_2)
{
    Statement_2
}
else if (Boolean_Expression_3)
{
    Statement_3
}
else
{
    Default_Statement
}
```



```
if (Boolean_Expression_1)
{
    Statement_1
}
if (Boolean_Expression_2)
{
    Statement_2
}
if (Boolean_Expression_3)
{
    Statement_3
}
else
{
    Default_Statement
}
```

# Exit Method

- The exit method immediately stops the program
- Can be used when there arises an error that will prevent a program from working
- The value “0” is generally used when the program is exiting under normal conditions

## Syntax

```
System.exit(<<Integer Value>>);
```

## Example

```
if(!validInput)
{
    System.exit(0);
}
```



# Short Circuit Evaluation

- Sometimes only a part of a Boolean expression needs to be evaluated to determine the entire value
  - If the first operand of an `&&` is false, then the entire expression is false
  - If the first operand of an `||` is true, then the entire expression is true
- Sometimes called Lazy Evaluation
- Very efficient and sometimes necessary

## Example 1

```
boolean a = false;
boolean b = true;
if(a && b)//b is never checked
{
}
}
```

## Example 2

```
boolean a = true;
boolean b = false;
if(a || b)//b is never checked
{
}
}
```

# Precedence

Highest Precedence	
Unary Operators	<code>+, -, ++, --, !</code>
Binary Arithmetic	<code>*, /, %</code>
Binary Arithmetic	<code>+, -</code>
Boolean Operators	<code>&lt;, &gt;, &lt;=, &gt;=</code>
Boolean Operators	<code>==, !=</code>
Boolean Operator	<code>&amp;</code>
Boolean Operator	<code> </code>
Boolean Operator	<code>&amp;&amp;</code>
Boolean Operator	<code>  </code>
Lowest Precedence	

# Outline

- If/else statements
- Boolean expressions
- Nested statements
- Else-if statements
- Example programs

# Math Quiz Program

```
/*
 * Written by JJ Shepherd
 */
import java.util.Random;
import java.util.Scanner;
public class MathChallenge03 {

    public static final int RNG_RANGE = 100;
    public static void main(String[] args) {
        Scanner keyboard = new Scanner(System.in);
        Random r = new Random();
        System.out.println("Are you ready for a math challenge!?\nEnter \"yes\" to continue.");

        String play = keyboard.nextLine();

        if(play.equalsIgnoreCase("yes"))//play.equals("yes")
        {
            System.out.println("LET THE CHALLENGE BEGIN!!!");
        }
        else
        {
            System.out.println("Aw...");
            System.exit(0);//Immediately stops the program here
        }

        int num1 = r.nextInt(RNG_RANGE);
        int num2 = r.nextInt(RNG_RANGE);
        int result = num1 + num2;
        System.out.println("What is "+num1+" + "+num2+"?");

        int answer = keyboard.nextInt();

        if(answer == result)
        {
            System.out.println("You win!!!");
        }
        else
        {
            System.out.println("SORRY! NOPE!");
        }

    }
}
```

# Box Picking

```
/*
 * Written by JJ Shepherd
 */
import java.util.Scanner;
public class LetsMakeADecision {

    public static final String BOX1 = "ET for the Atari";
    public static final String BOX2 = "Floppy Disk";
    public static final String BOX3 = "BRAND NEW CPU!";
    public static void main(String[] args) {
        Scanner keyboard = new Scanner(System.in);
        String c1Prize = "nothing";
        String c2Prize = "nothing";

        System.out.println("Let's make a decision!\nContestant 1, will you choose \"box1\",
        \"box2\", or \"box3\"?");

        String answer = keyboard.nextLine();

        if(answer.equalsIgnoreCase("box1"))
        {
            c1Prize = BOX1;
            System.out.println("Contestant 2, will you choose \"box2\" or \"box3\"?");
            answer = keyboard.nextLine();
            if(answer.equalsIgnoreCase("box2"))
            {
                c2Prize = BOX2;
            }
            else if(answer.equalsIgnoreCase("box3"))
            {
                c2Prize = BOX3;
            }
            else
            {
                System.out.println("THAT'S NOT VALID!!! YOU GET NOTHING!!!");
            }
        }
    }
}
```

# Box Picking Game

```
else if(answer.equalsIgnoreCase("box2"))
{
    c1Prize = BOX2;
    System.out.println("Contestant 2, will you choose \"box1\" or \"box3\"?");
    answer = keyboard.nextLine();
    if(answer.equalsIgnoreCase("box1"))
    {
        c2Prize = BOX1;
    }
    else if(answer.equalsIgnoreCase("box3"))
    {
        c2Prize = BOX3;
    }
    else
    {
        System.out.println("THAT'S NOT VALID!!! YOU GET NOTHING!!!");
    }
}
```

# Box Picking Game

```
else if(answer.equalsIgnoreCase("box3"))
{
    c1Prize = BOX3;
    System.out.println("Contestant 2, will you choose \"box1\" or \"box2\"?");
    answer = keyboard.nextLine();
    if(answer.equalsIgnoreCase("box1"))
    {
        c2Prize = BOX1;
    }
    else if(answer.equalsIgnoreCase("box2"))
    {
        c2Prize = BOX2;
    }
    else
    {
        System.out.println("THAT'S NOT VALID!!! YOU GET NOTHING!!!");
    }
}
else
{
    System.out.println("THAT'S NOT VALID!!! YOU BOTH GET NOTHING!!!");
    System.exit(0);
}
System.out.println("Contestant 1 won: "+c1Prize+"\nContestant 2 won: "+c2Prize+"\nThank
you for playing!");
}
}
```