



# Loops

Forest Agostinelli  
University of South Carolina

# Outline

- While loops
- Do-while loops
- For loops
- Control within loops

# While Loops

- While-statement
- If the Boolean expression is “true” then the body of the while-statement is executed until it is false
- Putting curly braces “{}” to denote the body of the while-statement is strongly encouraged
- Do not put a semicolon “;” after the parenthesis
  - It will ignore the Boolean expression
- Spoken
  - “while this is true then keeping doing that”

## Syntax

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

## Examples

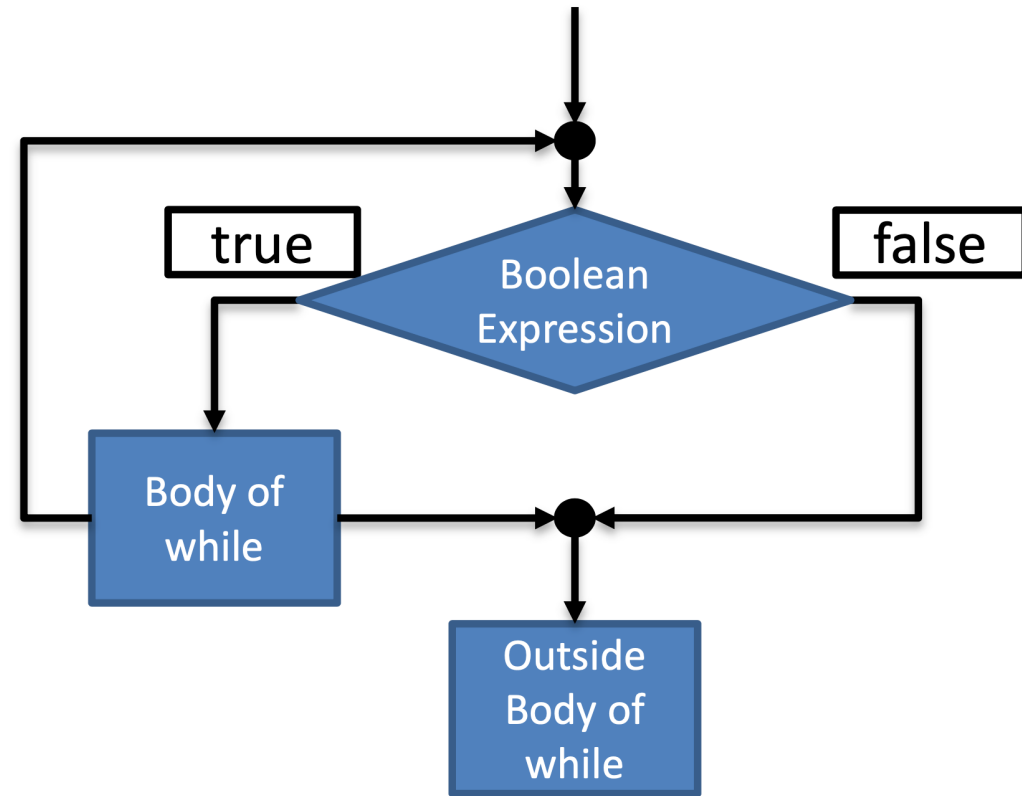
```
int a = 0;  
while(a < 10)  
{  
    System.out.println(a);  
    a++;  
}
```

# While Loops

## Syntax

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

## General While-Statement Flow Chart





# Example

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

    public static final int UPPER_NUMBER = 100;
    public static void main(String[] args) {
        Scanner keyboard = new Scanner(System.in);
        Random r = new Random();
        int secretNumber = r.nextInt(UPPER_NUMBER);
        System.out.println("I'm thinking of a number from 0 to "+(UPPER_NUMBER-1)+"\nGuess the
number!");
        int guessNumber = 0;
        boolean correctGuess = false;
        while(!correctGuess)
        {
            guessNumber = keyboard.nextInt();
            if(guessNumber > secretNumber)
            {
                System.out.println("That's too high!");
            }
            else if(guessNumber < secretNumber)
            {
                System.out.println("That's too low!");
            }
            else
            {
                System.out.println("That's correct!");
                correctGuess = true;
            }
        }
    }
}
```

# Infinite Loops

- Loop's Boolean expressions must eventually evaluate to "false"
- If this does not happen it creates a logic error called an "Infinite Loop"
- The body of the loop keeps running until the program is terminated
- Common Causes
  - Off by one errors
  - Incorrect bounds
  - Round off Errors
- Floating point types (float and double) should use "<=" or ">=" instead of "=="

## Example

```
int a = 10;
while(a > 0)
{
    System.out.println(a);
    a++;
}
//Another Example
double j = 10.0;
while(j != 0.0)
{
    j -= 0.1;
    System.out.println(j);
}
```

# Nested Loops

- Loops can be nested within the body of another loop
  - Much like branching statements
- Loops looping other loops can be full of logic errors

## Syntax

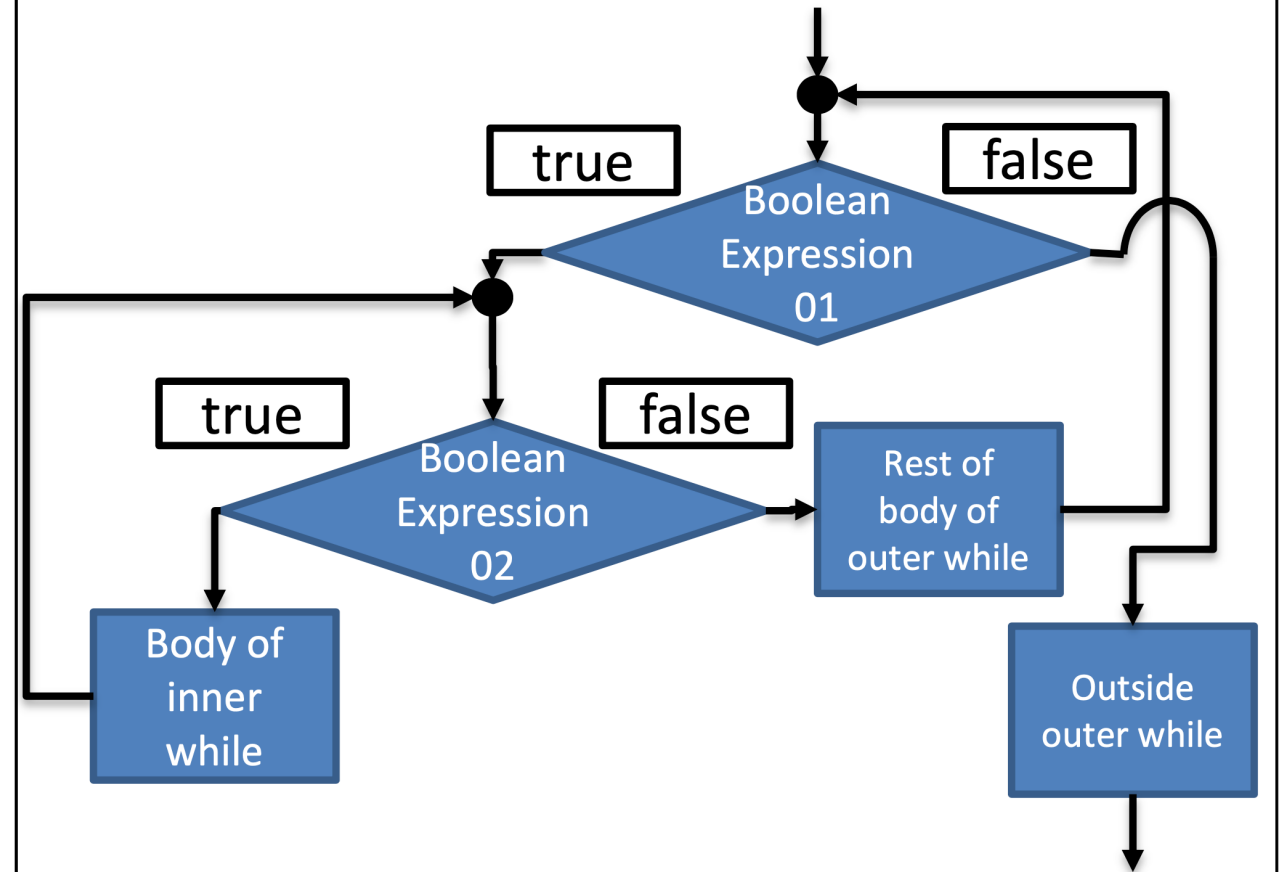
```
while(<<Boolean expression 01>>)  
{  
    while(<<Boolean expression 02>>)  
    {  
        ...  
    }  
}  
//Do-while can also be substituted
```

# Nested Loops

## Syntax

```
while(<<Boolean expression 01>>)  
{  
    while(<<Boolean expression 02>>)  
    {  
        //Body of inner while  
    }  
    //Rest of body of outer while  
}  
//Outside outer while
```

## Nested Loop Flow Chart



# Outline

- While loops
- Do-while loops
- For loops
- Control within loops

# Do-While Loops

- Do-while-statement
- The body of a do-while runs at least once
  - The body of a while may never run at all
- After running the body of the do-while, If the Boolean expression is “true” then the body of the do-while-statement is executed until it is false
- Putting curly braces “{}” to denote the body of the while-statement is strongly encouraged
- Put a semicolon “;” after the parenthesis
  - Otherwise it is a syntax error
- Spoken
  - “do that while this is true”

## Syntax

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

## Examples

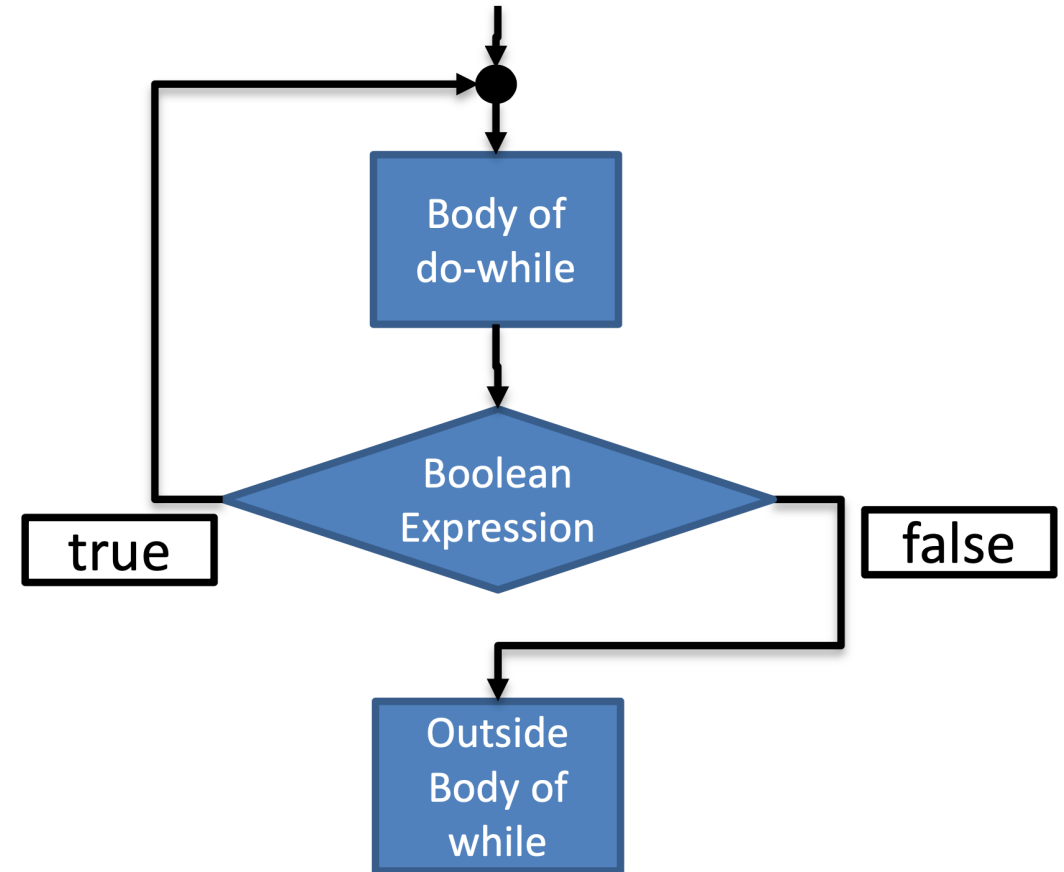
```
int a = 10;
do
{
    System.out.println(a);
    a++;
}while(a < 10);//Yes put the semicolon here
```

# Do-While Loops

## Syntax

```
do  
{  
    //Body of the do-while  
}while(<<Boolean expression>>);  
  
//Outside Body of the do-while
```

## General Do-While-Statement Flow Chart



# Example

```
/*
 * Written by JJ Shepherd
 */
import java.util.Scanner;
import java.util.Random;
public class NumberGuesser01 {

    public static final int UPPER_NUMBER = 100;
    public static void main(String[] args) {
        Scanner keyboard = new Scanner(System.in);
        Random r = new Random();
        boolean playAgain = true;
        do
        {
            int secretNumber = r.nextInt(UPPER_NUMBER);
            System.out.println("I'm thinking of a number from 0 to "+(UPPER_NUMBER-1)+"\nGuess
the number!");
            int guessNumber = 0;
            boolean correctGuess = false;
            while(!correctGuess)
            {
                guessNumber = keyboard.nextInt();
                if(guessNumber > secretNumber)
                {
                    System.out.println("That's too high!");
                }
                else if(guessNumber < secretNumber)
                {
                    System.out.println("That's too low!");
                }
                else
                {
                    System.out.println("That's correct!");
                    correctGuess = true;
                }
            }
            System.out.println("Enter \"true\" to play again");
            playAgain = keyboard.nextBoolean();
        }while(playAgain);
    }
}
```



# Outline

- While loops
- Do-while loops
- For loops
- Control within loops

# For Loops

- For-statement
- Counting Loop
- Special kind of While-Statement
- Arguments require 3 parts, separated by semicolons
  - Init: This initialization of a counting variable. Only runs once.
  - Boolean Expression: Just like before
  - Update: Updates the counting variable after all other statements in the body have run
- Putting curly braces “{}” to denote the body of the for-statement is strongly encouraged
- Do not put a semicolon “;” after the parenthesis
- Spoken
  - “Do that for this many times”

## Syntax

```
for(<<Init>> ; <<Boolean expression>> ; <<Update>>)  
{  
    //Body of the for-statement  
}  
//Outside Body of the for-statement
```

## Examples

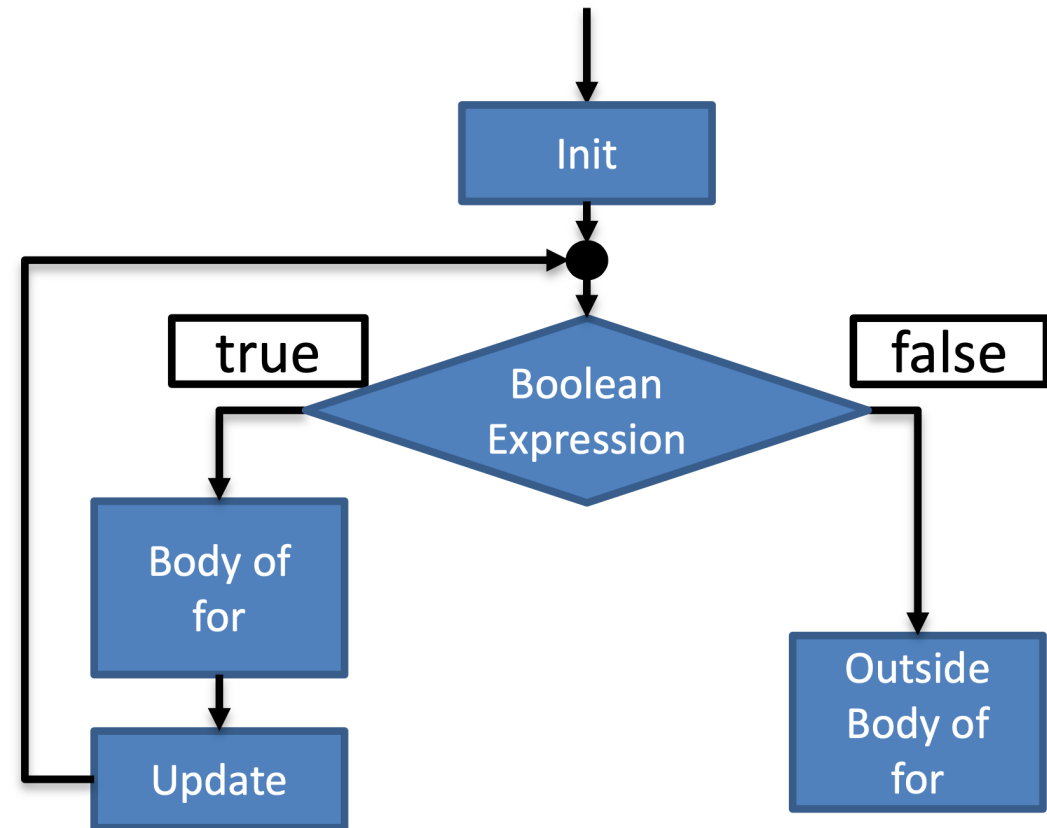
```
for(int i=0; i<10; i++)  
{  
    System.out.println(i);  
}
```

# For Loops

## Syntax

```
for(<<Init>> ; <<Boolean expression>> ; <<Update>>)  
{  
    //Body of the for-statement  
}  
//Outside Body of the for-statement
```

## General For-Statement Flow Chart



# For Loops

## For-Loop

```
for(<<Init>> ; <<Boolean expression>> ; <<Update>>)
{
    //Body of the for-statement
}
//Outside Body of the for-statement
```

## While-Loop

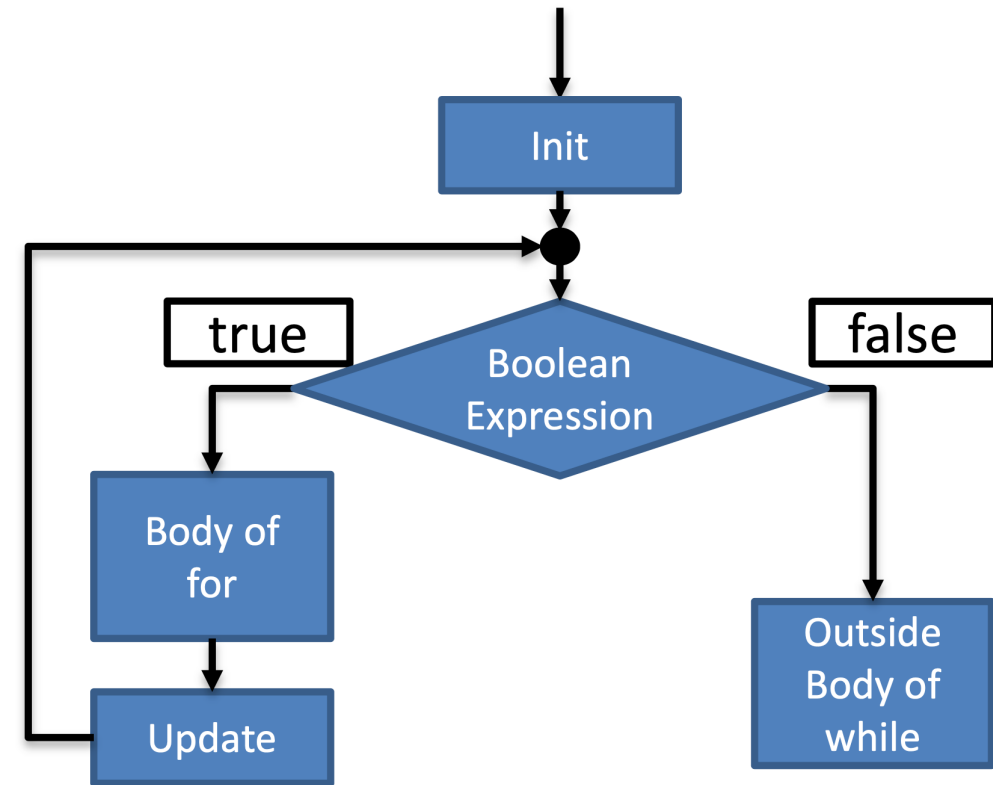
```
<<Init>>;
while(<<Boolean expression>>)
{
    //Body of the while-statement
    <<Update>>;
}
//Outside Body of the while-statement
```

# For Loops

## Syntax

```
for(<<Init>> ; <<Boolean expression>> ; <<Update>>)  
{  
    //Body of the for-statement  
}  
//Outside Body of the for-statement
```

## General For-Statement Flow Chart



# Quick Quiz

- What is the output of this code?

```
int length = 2;
int width = 3;
for (int i=0; i<length; i++) {
    for (int j=0; j<width; j++) {
        System.out.print("*");
    }
    System.out.println();
}
```

# Quiz Answer

```
for(int i=0;i<length;i++)
{
    for(int j=0;j<width;j++)
    {
        System.out.print("*");
    }
    System.out.println();
}
```

## Variable Values

```
length = 2
width = 3;
i = DOES NOT EXIST
j = DOES NOT EXIST
```

## Console

# Quiz Answer

```
→ for(int i=0;i<length;i++)  
{  
    for(int j=0;j<width;j++)  
    {  
        System.out.print("*");  
    }  
    System.out.println();  
}
```

## Variable Values

```
length = 2  
width = 3;  
i = DOES NOT EXIST  
j = DOES NOT EXIST
```

## Console



# Quiz Answer

```
→ for(int i=0; i<length; i++)  
{  
    for(int j=0; j<width; j++)  
    {  
        System.out.print("*");  
    }  
    System.out.println();  
}
```

## Variable Values

```
length = 2  
width = 3;  
i = 0  
j = DOES NOT EXIST
```

## Console

# Quiz Answer

```
→ for(int i=0; i<length; i++)  
{  
    for(int j=0; j<width; j++)  
    {  
        System.out.print("*");  
    }  
    System.out.println();  
}
```

## Variable Values

```
length = 2  
width = 3;  
i = 0  
j = DOES NOT EXIST
```

## Console

# Quiz Answer

```
for(int i=0;i<length;i++)
{
    →for(int j=0;j<width;j++)
    {
        System.out.print("*");
    }
    System.out.println();
}
```

## Variable Values

```
length = 2
width = 3;
i = 0
j = DOES NOT EXIST
```

## Console

# Quiz Answer

```
for(int i=0;i<length;i++)
{
    → for(int j=0;j<width;j++)
        {
            System.out.print("*");
        }
    System.out.println();
}
```

## Variable Values

```
length = 2
width = 3;
i = 0
j = 0
```

## Console

# Quiz Answer

```
for(int i=0;i<length;i++)
{
    →for(int j=0;j<width;j++)
    {
        System.out.print("*");
    }
    System.out.println();
}
```

## Variable Values

```
length = 2
width = 3;
i = 0
j = 0
```

## Console

# Quiz Answer

```
for(int i=0;i<length;i++)
{
    for(int j=0;j<width;j++)
    {
        →System.out.print("*");
    }
    System.out.println();
}
```

## Variable Values

```
length = 2
width = 3;
i = 0
j = 0
```

## Console

```
*
```

# Quiz Answer

```
for(int i=0;i<length;i++)
{
    →for(int j=0;j<width;j++)
    {
        System.out.print("*");
    }
    System.out.println();
}
```

## Variable Values

```
length = 2
width = 3;
i = 0
j = 1
```

## Console

```
*
```

# Quiz Answer

```
for(int i=0;i<length;i++)
{
    →for(int j=0;j<width;j++)
        {
            System.out.print("*");
        }
    System.out.println();
}
```

## Variable Values

```
length = 2
width = 3;
i = 0
j = 1
```

## Console

```
*
```



# Quiz Answer

```
for(int i=0;i<length;i++)
{
    for(int j=0;j<width;j++)
    {
        →System.out.print("*");
    }
    System.out.println();
}
```

## Variable Values

```
length = 2
width = 3;
i = 0
j = 1
```

## Console

```
**
```

# Quiz Answer

```
for(int i=0;i<length;i++)
{
    →for(int j=0;j<width;j++)
    {
        System.out.print("*");
    }
    System.out.println();
}
```

## Variable Values

```
length = 2
width = 3;
i = 0
j = 2
```

## Console

```
**
```

# Quiz Answer

```
for(int i=0;i<length;i++)
{
    →for(int j=0;j<width;j++)
    {
        System.out.print("*");
    }
    System.out.println();
}
```

## Variable Values

```
length = 2
width = 3;
i = 0
j = 2
```

## Console

```
**
```

# Quiz Answer

```
for(int i=0;i<length;i++)
{
    for(int j=0;j<width;j++)
    {
        →System.out.print("*");
    }
    System.out.println();
}
```

## Variable Values

```
length = 2
width = 3;
i = 0
j = 2
```

## Console

```
***
```

# Quiz Answer

```
for(int i=0;i<length;i++)
{
    →for(int j=0;j<width;j++)
    {
        System.out.print("*");
    }
    System.out.println();
}
```

## Variable Values

```
length = 2
width = 3;
i = 0
j = 3
```

## Console

```
***
```

# Quiz Answer

```
for(int i=0;i<length;i++)
{
    →for(int j=0;j<width;j++)
        {
            System.out.print("*");
        }
    System.out.println();
}
```

## Variable Values

```
length = 2
width = 3;
i = 0
j = 3
```

## Console

```
***
```

# Quiz Answer

```
for(int i=0;i<length;i++)
{
    for(int j=0;j<width;j++)
    {
        System.out.print("*");
    }
    → System.out.println();
}
```

## Variable Values

```
length = 2
width = 3;
i = 0
j = DOES NOT EXIST
```

## Console

```
***
```

# Quiz Answer

```
→ for(int i=0;i<length;i++)  
{  
    for(int j=0;j<width;j++)  
    {  
        System.out.print("*");  
    }  
    System.out.println();  
}
```

## Variable Values

```
length = 2  
width = 3;  
i = 1  
j = DOES NOT EXIST
```

## Console

```
***
```



# Quiz Answer

```
for(int i=0;i<length;i++)  
{  
    → for(int j=0;j<width;j++)  
        {  
            System.out.print("*");  
        }  
    System.out.println();  
}
```

## Variable Values

```
length = 2  
width = 3;  
i = 1  
j = 0
```

## Console

```
***
```

# Quiz Answer

```
for(int i=0;i<length;i++)
{
    →for(int j=0;j<width;j++)
    {
        System.out.print("*");
    }
    System.out.println();
}
```

## Variable Values

```
length = 2
width = 3;
i = 1
j = 0
```

## Console

```
***
```

# Quiz Answer

```
for(int i=0;i<length;i++)
{
    for(int j=0;j<width;j++)
    {
        →System.out.print("*");
    }
    System.out.println();
}
```

## Variable Values

```
length = 2
width = 3;
i = 1
j = 0
```


## Console

```
***
*
```

A Few Steps Later...

# Quiz Answer

```
for(int i=0;i<length;i++)  
{  
    for(int j=0;j<width;j++)  
    {  
        System.out.print("*");  
    }  
    System.out.println();  
}
```



## Variable Values

```
length = 2  
width = 3;  
i = DOES NOT EXIST  
j = DOES NOT EXIST
```

## Console

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

# Loops Summary

- While
  - Body runs 0 to many times
  - Great for “ask-before-iterating”
- Do-While
  - Body runs 1 to many times
  - Great for “ask-before-iterating”
- For-Loop
  - Body runs a countable number of times
  - Great for “count-controlled” situations

# Example

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

    public static void main(String[] args) {
        Scanner keyboard = new Scanner(System.in);
        System.out.println("I can draw a box using stars (*)\nGive me a length followed by a
width, where both are greater than 0");

        int length = keyboard.nextInt();
        int width = keyboard.nextInt();

        if(length <= 0 || width <= 0)
        {
            System.out.println("That is invalid!");
            System.exit(0);
        }
        for(int i=0;i<length;i++)
        {
            for(int j=0;j<width;j++)
            {
                System.out.print("*");
            }
            System.out.println();
        }
    }
}
```

# Example

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

    public static void main(String[] args) {
        Scanner keyboard = new Scanner(System.in);

        System.out.println("I'm an octopus, so I like the number 8.\nEnter a positive value and
I'll count up by 8's! Blub blub");
        int numberInput = keyboard.nextInt();

        if(numberInput < 0)
        {
            System.out.println("That's not valid!");
            System.exit(0);
        }

        for(int i=0;i<numberInput;i+=8)
        {
            System.out.println(i);
        }
        /*
        for(int i=0;i<numberInput;i++)
        {
            if(i%8 == 0)
            {
                System.out.println(i);
            }
        }
        */
    }
}
```



# Outline

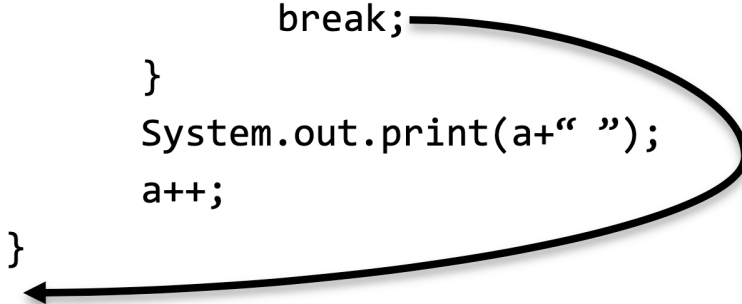
- While loops
- Do-while loops
- For loops
- Control within loops

# Break Statement

- The statement “break” immediately stops a loop.
- Once the break statement is reached the it will run the next statements outside the body of the loop.
  - “Jumps out of the loop”
- For nested loops it stops the loop whose body the break statement is found.

## Break Example

```
int a = 0;
while(true)
{
    if(a >= 5)
    {
        break;
    }
    System.out.print(a+" ");
    a++;
}
```



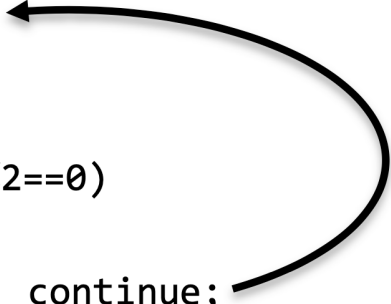
//Output: 0 1 2 3 4

# Continue

- The statement “continue” stops the current loop iterations and starts a new one.
- Once the continue statement is reached it immediately starts the loop again.
  - “Jumps back to the start of the loop and continues”
- For nested loops it continues the loop whose body the continue statement is found.

## Continue Example

```
int a = 0;
while(a<10)
{
    a++;
    if(a%2==0)
    {
        continue;
    }
    System.out.print(a+ " ");
}
//Output: 1 3 5 7 9
```



# Sentinel Values

- Special value(s) used signal the end of an algorithm.
- We can use these to stop loops.
- Sentinel values should be selected so that they are distinct from other valid values.

## Sentinel Value Example

```
//Find the average of positive values
int value = 0;
int sum = 0;
int count = 0;
while(true)
{
    value = keyboard.nextInt();
    if(value < 0)//The sentinel values are negative
    {
        break;
    }
    sum += value;
    count++;
}
int average = sum / count;
System.out.println(average);
//If the input was 2 4 6 8 10 -1 then it would output 6
```