Recap: While Statement

while (boolean expression)

- simplest form of loop in Java
- body of loop can be
  - single statement
  - whole block of many statements in curly braces
- control flow:
  - body executed if expression is true
  - then boolean expression evaluated again
  - if expression still true, body executed again
  - repetition continues until expression false
  - then processing continues with next statement after loop

Recap: If Versus While Statements

- how if statement works
- boolean expression

  - true
  - how if statement works
  - boolean expression
  - false

  - how while statement works

Using while Statements

public class WhileDemo {
  public static void main (String[] args) {
    int limit = 3;
    int counter = 1;
    while (counter <= limit) {
      System.out.println("The square of " + counter + " is " + (counter * counter));
      counter = counter + 1;
    }
    System.out.println("End of demonstration");
  }
}

Using while Statements

- while statement
- control flow resumes here when boolean is false

Using while Statements

- statement after while
- control flow resumes here when boolean is false

Using while Statements

- trace what happens when execute

Using while Statements

- limit

Using while Statements

- limit
- counter

```
int limit = 3;
int counter = 1;
while (counter <= limit) {
  System.out.println("The square of " + counter + " is " + (counter * counter));
  counter = counter + 1;
}
System.out.println("End of demonstration");
```
Using while Statements

```java
public class WhileDemo {  
  public static void main (String[] args)  
  {    
    int limit = 3;    
    int counter = 1;    
    while (counter <= limit)    
    {      
      System.out.println("The square of " + counter + " is " + (counter * counter));      
      counter = counter + 1;    
    }    
    System.out.println("End of demonstration");  
  }
}
```

Is counter <= limit? yes

**Infinite Loops**

```java
public class WhileDemo {  
  public static void main (String[] args)  
  {    
    int limit = 3;    
    int counter = 1;    
    while (counter <= limit)    
    {      
        System.out.println("The square of " + counter + " is " + (counter * counter));      
        counter = counter + 1;    
    }    
    System.out.println("End of demonstration");  
  }
}
```

Is counter <= limit? NO!

Climbing Stairs Again

- Am I at the top of the stairs?
  - No
  - Climb up one step
  - Am I at the top of the stairs?
    - No
    - Climb up one step
    - Am I at the top of the stairs?
      - No
      - Climb up one step
      - Am I at the top of the stairs?
        - No
        - Climb up one step
        - Am I at the top of the stairs?
          - Yes

```
```

**Infinite Loops**

```java
public class WhileDemo {  
  public static void main (String[] args)  
  {    
    int limit = 3;    
    int counter = 1;    
    while (counter <= limit)    
    {      
        System.out.println("The square of " + counter + " is " + (counter * counter));      
        counter = counter + 1;    
    }    
    System.out.println("End of demonstration");  
  }
}
```

Is counter <= limit? NO!

```
```
Another while Example

public class PrintFactorials
{
    public static void main(String[] args)
    {
        int limit = 10;
        int counter = 1;
        int product = 1;
        while (counter <= limit)
        {
            System.out.print("factorial of "+counter+");
            product = product * counter;
        }
        System.out.println("End of demonstration");
    }
}

Fun With Loops

public class BeerSong
{
    public static void main(String[] args)
    {
        int beerNum = 99;
        String word = "bottles";
        while (beerNum > 0)
        {
            System.out.println(""+beerNum+" of beer on the wall.");
            beerNum = beerNum - 1;
        }
        System.out.println("No more bottles of beer on the wall.");
    }
}

Other Loop Statements

public class ForDemo
{
    public static void main(String[] args)
    {
        for (int counter = 1; counter <= 3; counter = counter + 1)
        {
            System.out.println("The square of "+counter+" is "+(counter*counter));
        }
        System.out.println("End of demonstration");
    }
}

For Statement

public class ForDemo
{
    public static void main(String[] args)
    {
        for (int counter = 1; counter <= 3; counter = counter + 1)
        {
            System.out.println("The square of "+counter+" is "+(counter*counter));
        }
        System.out.println("End of demonstration");
    }
}

For Statement

public class ForDemo
{
    public static void main(String[] args)
    {
        for (int counter = 1; counter <= 3; counter = counter + 1)
        {
            System.out.println("The square of "+counter+" is "+(counter*counter));
        }
        System.out.println("End of demonstration");
    }
}

For Versus While Statement

public class ForDemo
{
    public static void main(String[] args)
    {
        for (int counter = 1; counter <= 3; counter = counter + 1)
        {
            System.out.println("The square of "+counter+" is "+(counter*counter));
        }
        System.out.println("End of demonstration");
    }
}

Four Things Needed In Any Loop

- Give starting values to one or more variables used in loop
- Test to see when looping stops
- Flowcharts can be somewhat deceptive
- Need initialization and incrementing/modifying in while loop too
- Although syntax does not require it in specific spot

Four Things Needed In Any Loop

- Give starting values to one or more variables used in loop
- Test to see when looping stops
- Flowcharts can be somewhat deceptive
- Need initialization and incrementing/modifying in while loop too
- Although syntax does not require it in specific spot

For Statement

- Header has three parts
  - separated by semicolons
- Initialization: first part
  - executed only one time, at beginning
- boolean expression: second part
  - evaluated just before loop body, like in while
- Increment: third part
  - executed at end of loop body
- Despite name, arbitrary calculation allowed
  - could decrement, for example!
Four Things Needed In Any Loop

- Give starting values to one or more variables used in loop
- Test to see when looping stops
- One or more useful operations here
- Change something to move process closer to termination

How loops work in general

Yet Another Loop Statement

- do version: not quite equivalent
- termination test at end, so body executed at least once

Do Statement

- Body always executed at least once

Four Things Needed In Any Loop

- Give starting values to one or more variables used in loop
- Test to see when looping stops
- One or more useful operations here
- Change something to move process closer to termination

How loops work in general

Do Statement

- Body always executed at least once