Reading

- This week: Chapter 6 all (6.1-6.4)
- second edition: Chap 7
News

■ Welcome back!
■ Midterms returned last time
  ■ get yours after class if you didn't already
■ Department news
Department of Computer Science Undergraduate Events

Events this week

Resume & Cover Letter Drop-In Session
Date: Wed., Mar 3
Time: 12 – 3 pm (20 mins. sessions)
Location: Rm 255, ICICS/CS

Find a Job Fast! Info Session
Date: Thurs., Mar 4
Time: 12:30 – 1:45 pm
Location: DMP 201
Registration: Email dianejoh@cs.ubc.ca

Townhall Meeting – 1st Year CS Students
Date: Thurs., Mar 4
Time: 12:30 - 2 pm
Location: DMP 310
Lunch will be provided!

Faculty Talk – Son Vuong
Title: Mobile Learning via LIVES
Date: Thurs., Mar 4
Time: 12:30 – 1:45 pm
Location: DMP 201

Events next week

Townhall Meeting – Combined
Majors/Honours, BA, B.Comm in CS
Date: Thurs., Mar 11
Time: 12:30 – 2 pm
Location: DMP 310
Lunch will be provided!

CS Distinguished Lecture Series –
Featuring David Parkes
Title: Incentive Mechanism
Engineering in the Internet Age
Date: Thurs., Mar 11
Time: 3:30 – 4:50 pm
Location: DMP 110

CSSS Movie Night –
“Zombieland” & “Iron Man”
Date: Thurs., Mar 11
Time: 6 – 10 pm
Location: DMP 310
Free pop & popcorn!
Recap: While Statement

\textbf{while} (boolean expression)
  
  body

- 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 \(^5\)
Recap: *If Versus While Statements*

- How can loop boolean change from false to true?

![Diagram](attachment://diagram.png)
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");
    }
}
```

- **while** statement
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");
    }
}
```

- **boolean expression**
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");
    }

    }
}
```

- `while` statement body
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");
    }
}

- statement after while
  - control flow resumes here when boolean is false
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");
    }
}
```

- trace what happens when execute
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");
    }
}
```

`limit` 3
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");
    }
}
```

`limit` 3  `counter` 1
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");
    }
}
```

**limit** 3  **counter** 1  Is counter <= limit? yes
Using \texttt{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");
    }
}
```

\begin{tabular}{ll}
\textbf{limit} & 3 \\
\textbf{counter} & 1 \\
\end{tabular}

Is counter $\leq$ limit? yes

"The square of 1 is 1" printed on monitor
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");
    }
}
```

```
limit 3  counter 2
```
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");
    }
}
```

**limit** 3  **counter** 2  **Is counter <= limit?** yes
Using **while** Statements

```java
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");
    }
}
```

Limit: 3  Counter: 2  Is counter <= limit? yes

"The square of 2 is 4" printed on monitor
Using \textbf{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));
            \textcolor{red}{counter = counter + 1;}
        }
        System.out.println("End of demonstration");
    }
}
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");
    }
}

limit 3  counter 3  Is counter <= limit? yes
Using \textbf{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");
    }
}
```

\begin{itemize}
\item limit \, 3 \, \text{counter} \, 3 \, \text{Is counter} \leq \text{limit? yes}
\end{itemize}

"The square of 3 is 9" printed on monitor
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");
    }
}
```

limit: 3  counter: 4
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");
    }
}
```

limit 3    counter 4    Is counter <= limit? NO!
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!

“End of demonstration" printed on monitor
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.
- ...and so on...
Climbing Stairs Again

while (I’m not at the top of the stairs)
{
    Climb up one step
}

- Climbing stairs is a while loop!
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");
    }
}
```

■ **change** termination condition
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");
    }
}
```

- **change** termination condition
- **body of loop never executed**
Using **while** Statements

```java
public class WhileDemo {
    public static void main (String[] args) {
        int limit = 3;
        int counter = 1;

        while (counter >= counter) {
            System.out.println("The square of " + counter + " is " + (counter * counter));
            counter = counter + 1;
        }
        System.out.println("End of demonstration");
    }
}
```

- change termination condition
  - always true
Infinite Loops

public class WhileDemo
{
    public static void main (String[] args)
    {
        int limit = 3;
        int counter = 1;

        while (counter >= counter)
        {
            System.out.println("The square of " + counter + " is " + (counter * counter));
            counter = counter + 1;
        }
        System.out.println("End of demonstration");
    }
}

■ if termination condition always true, loop never ends
  ■ infinite loop goes forever
Infinite Loops

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");
    }
}

■ good termination condition
■ but process never gets closer to condition
Infinite Loops

public class WhileDemo
{
    public static void main (String[] args)
    {
        int limit = 9;
        int counter = 0;

        while (counter != limit)
        {
            System.out.println("The square of " + counter + " is " + (counter * counter));
            counter = counter + 2;
        }
        System.out.println("End of demonstration");
    }
}

- process gets closer to termination condition
- but never satisfies condition, keeps going past it
Another **while** Example

```java
public class PrintFactorials {
    public static void main (String[] args) {
        int limit = 10;
        int counter = 1;
        int product = 1;

        while (counter <= limit) {
            System.out.println("The factorial of " + counter + " is " + product);
            counter = counter + 1;
            product = product * counter;
        }
        System.out.println("End of demonstration");
    }
}
```

- **accumulate product**
public class BeerSong {
    public static void main(String[] args) {
        int beerNum = 99;
        String word = "bottles";
        while (beerNum > 0) {
            if (beerNum == 1) {
                word = "bottle";
            }
            System.out.println(beerNum + " " + word + " of beer on the wall.");
            System.out.println(beerNum + " " + word + " of beer.");
            System.out.println("If one of the bottles");
            System.out.println("should happen to fall...");
            beerNum = beerNum - 1;

            if (beerNum < 1) {
                System.out.println("No more bottles of beer on the wall.");
            }
        }
    }
}
import java.util.Scanner;

public class BeerSong2 {
    public static void main (String[] args) {
        int beerNum = 99;
        String word = "bottles";

        boolean keepgoing = true;
        String answer;
        Scanner in = new Scanner(System.in);

        while ((beerNum > 0) && keepgoing) {
            if (beerNum == 1) {
                word = "bottle";
            }

            System.out.println(beerNum + " " + word + " of beer on the wall.");
            System.out.println(beerNum + " " + word + " of beer.");
            System.out.println("If one of the bottles");
            System.out.println("should happen to fall... ");
            beerNum = beerNum - 1;
        }
    }
}
Fun With Loops

```java
System.out.println("Continue? (y/n): ");
answer = in.nextLine();
if (answer.equals("n"))
{
    keepgoing = false;
}

if (beerNum < 1)
{
    System.out.println("No more bottles of beer on the wall.");
}
```
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");
    }
}

■ Equivalent to...
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");
    }
}

...this loop using 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**
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");
    }
}

- **Initialization**: first part
  - executed only one time, at beginning
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");
    }
}

- boolean expression: second part
  - evaluated just before loop body, like in while
public class ForDemo
{
    public static void main (String[] args)
    {
        for (int counter = 1; counter <= 3; counter = counter + 1) { /* Increment: third part */
            System.out.println("The square of " + counter + " is " + (counter * counter));
        }
        System.out.println("End of demonstration");
    }
}

- **Increment**: third part
  - executed at end of loop body
- Despite name, arbitrary calculation allowed
  - could decrement, for example!
For Versus While Statement

how for statement works

- initialization
- boolean expression
  - true
  - statement
  - increment
  - false

how while statement works

- boolean expression
  - true
  - statement
  - false
**For Versus While Statement**

how for statement works

- initialization
- boolean expression
- true → statement
- increment
- false

how while statement works

- boolean expression
- true → statement
- false

- flowcharts can be somewhat deceptive
- need initialization and incrementing/modifying in while loop too
- although syntax does not require it in specific spot
For Versus While Statement

- Anything that can be done with one type of loop can be done with another
  - for and while are equivalent

- For statement convenient when
  - loop should be executed specific number of times
  - number can be determined before loop starts

- While statement convenient when
  - don't know yet how many times to execute loop body
  - but can check if it's time to end loop as you go
Four Things Needed In Any Loop

- Give starting values to one or more variables used in loop

initialize

test

true

do useful stuff

false

get closer to termination

how loops work in general
Four Things Needed In Any Loop

- Give starting values to one or more variables used in loop
- Test to see when looping stops

initialize

test

true

false
do useful stuff

get closer to termination

how loops work in general
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

how loops work in general
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
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");
    }
}

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

        do
        {
            System.out.println("The square of " + counter + " is " + (counter * counter));
            counter = counter + 1;
        } while (counter <= limit);

        System.out.println("End of demonstration");
    }
}

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

Order of four things can change, but need them all