Announcements

- Later – No Lab!
  - Time to work on Programs 1 and 2 independently

- Program 1 due today

- Program 2 due Monday

- Office hours today
  - 1–2 PM
Questions?
If–Else or Switch Statement

- Use a switch statement when you have more than 2 conditions on a single variable

- Example: Weekdays – if you have a different action to perform for each day of the week, use a switch statement
If–Else or Switch Statement

- Use an if–else for all other scenarios:
  - More than one variable you’re testing (multiple conditions)
  - Testing for a range of values
  - Variable is not an int or char

- Example: Grades – each grade (A, B, C, D, F) has a range of values that reflect each grade letter
  - if (grade >= 90) {
    // A
  } else if (grade >= 80) {
    // B
  ...
Loops

- Loop – part of program that repeats
- Body – statements being repeated
- Iteration – each repetition of body
- Stopping condition
Types of Loops

- **while**
  - Safest choice
  - Not always most elegant
  - Loop iterates 0 or more times

- **do–while**
  - Loop iterates at least ONCE

- **for**
  - Numeric computation changes by equal amount
The **while** Statement

- Also called a **while** loop
- A **while** statement repeats while a controlling boolean expression remains true
- The loop body typically contains an action that ultimately causes the controlling boolean expression to become false.
While Loops

Evaluate Boolean Expression

true

false

Execute Body

End Loop
While Loops

```java
while (count <= number) {
    System.out.print(count + "", "");
    count++;
}
```
While Loops

- Syntax

  while (Boolean_Expression)
    Body_Statement

  or

  while (Boolean_Expression)
  { 
    First_Statement
    Second_Statement
    ...
  }
The **do-while** Statement

- Also called a **do-while** loop
- Similar to a **while** statement, except that the loop body is executed at least once
- Syntax
  
  ```
  do
  Body_Statement
  while (Boolean_Expression);
  ```
  
  Don’t forget the semicolon!
Do-While Loops

1. Evaluate Boolean Expression
2. Execute Body
3. Evaluate Boolean Expression
   - If true, go back to Execute Body
   - If false, End Loop
Do–while Loops

do
{
    System.out.print(count + ", ");
    count++;
} while (count <= number);
First, the loop body is executed.

Then the boolean expression is checked.
- As long as it is true, the loop is executed again.
- If it is false, the loop is exited.

Equivalent **while** statement

```
Statement(s)_S1
while (Boolean_Condition)
    Statement(s)_S1
```
Loop Practice

Write a while loop or a do-while loop that will compute the sum of the first n positive odd numbers. For example, if n is 5, you should compute $1 + 3 + 5 + 7 + 9$. 
The **for** Statement

- A **for** statement executes the body of a loop a fixed number of times.
- Also known as a for loop.
- Example
  ```java
  for (count = 1; count <= 3; count++) {
    System.out.println(count);
  }
  ```
Syntax

```
for (Initialization, Condition, Update)  
  Body_Statement
```

- **Body_Statement** can be either a simple statement or a compound statement in `{}`.

- Corresponding **while** statement

```
Initialization
while (Condition)  
  Body_Statement_Including_Update
```
For Loops

Evaluate Boolean Expression

Execute Initialization

Evaluate Boolean Expression

true -> Execute Body
false -> End Loop

Execute Update Action
For Loops

```java
for (countDown = 3; countDown >= 0; countDown--)
{
    System.out.println(countDown);
    System.out.println("and counting.");
}
```
For Loops

- Possible to declare variables within a `for` loop

```java
int sum = 0;
for (int n = 1 ; n <= 10 ; n++) {
    sum = sum + n * n;
}
```

- Note that variable `n` is local to the loop
Loop Practice

Write a for loop that will compute the sum of the first n positive even numbers. For example, if n is 5, you should compute $2 + 4 + 6 + 8 + 10$. 
Types of Loops

- **while**
  - Safest choice
  - Not always most elegant
  - Loop iterates 0 or more times

- **do-while**
  - Loop iterates at least ONCE

- **for**
  - Numeric computation changes by equal amount
Bugs

- Problem with program preventing correct execution

- Two most common mistake in loops
  - Off-by-one errors
  - Infinite Loops!!!!!!
A loop which repeats without ever ending is called an infinite loop.

If the controlling boolean expression never becomes false, a loop will repeat without ending.
count = 1;
while (count <= num)
{
    System.out.print(count + " ", ");
    //count++;  
}
Infinite Loops

count = 1;
while (count <= num);
{
    System.out.print(count + " , ");
    count++; 
}

int count;
// initializing action; boolean expression; update action
for (count = 1; count >= num; count++)
{
    System.out.print(count + " , ");
}