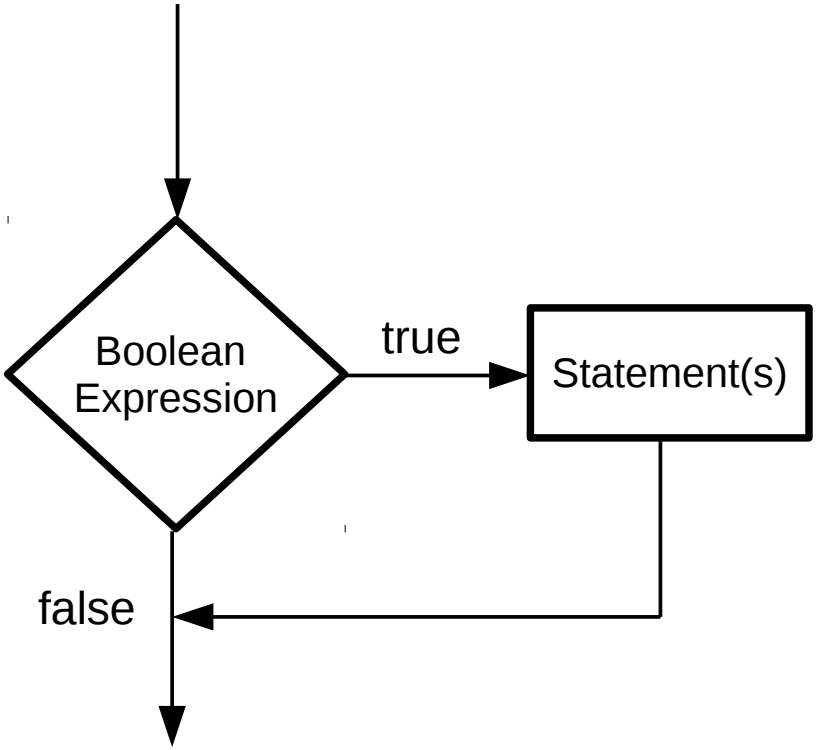
CIS 351-Data Structure-Loop

Farzana Rahman
Syracuse University

Slides prepared by Farzana Rahman

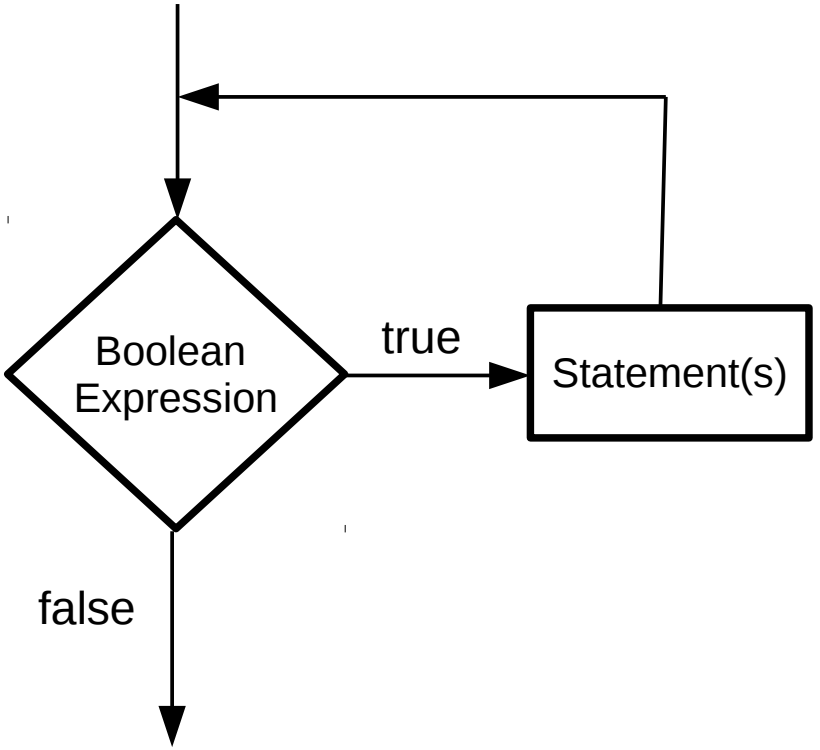
Reminder: if-statements

```
if (BooleanExpression) {
   Statement(s)
}
```



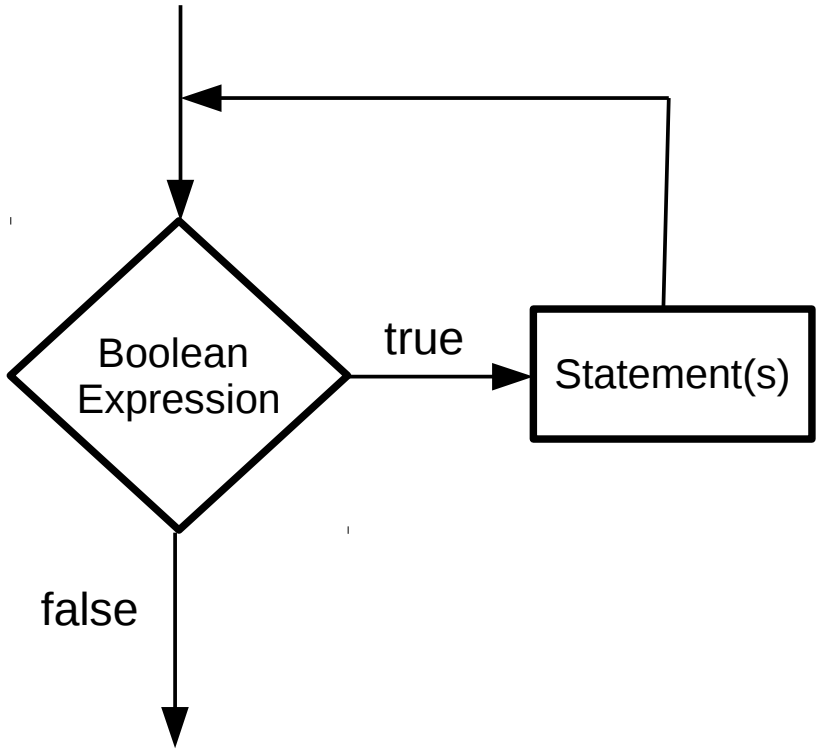
While Loops

```
while (BooleanExpression) {
   Statement(s)
}
```



While Loops

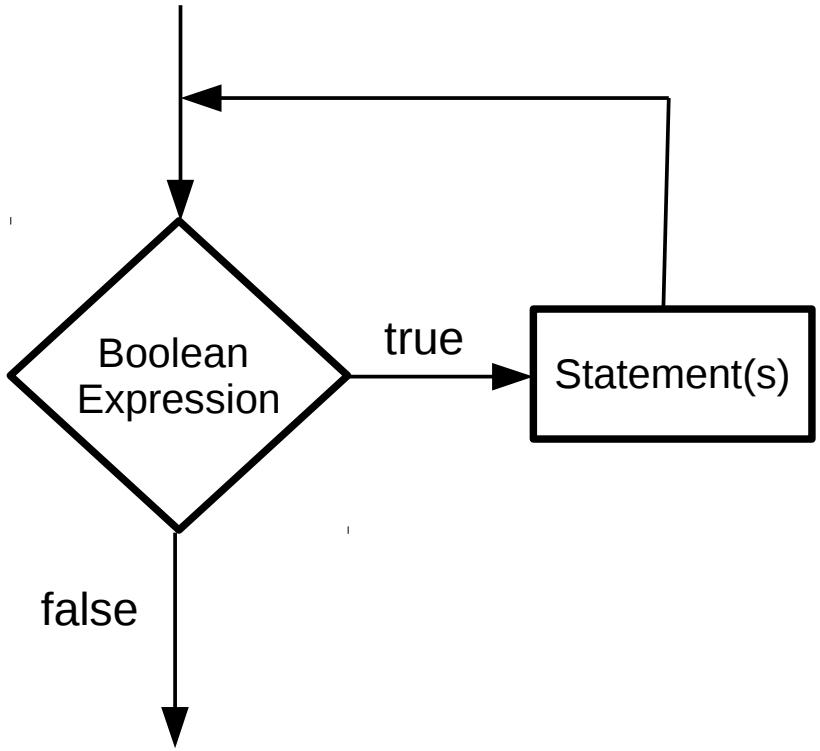
```
int a = 0;
while (a < 5) {
   System.out.println(“Hello.”);
}
```



Is there a problem?

While Loops

```
int a = 0;
while (a < 5) {
   System.out.println(“Hello.”);
   a++;
}
```



The body of every while loop should contain instruction(s) that can change the truth value of the logical expression

If for Input Validation

- We can use an if-statement to make sure that the user enters valid data:

```
System.out.print("Withdrawl amount: ");
amount = input.nextDouble();

if (amount < 1.0 || amount > 300.0) {
   System.out.println("Bad withdrawal amount!");
   System.exit(0); // Exits the application.
}

System.out.printf("Here are your %.2f dollars.", amount);
```

- Problem: user only gets one shot.

While Loop for Input Validation

- Use a while loop to keep asking *while* the user still hasn't entered a valid number:

```
System.out.print("Withdrawl amount: ");
amount = input.nextDouble();

while (amount < 1.0 || amount > 300.0) {
   System.out.println("Amount must be $1.00 - $300.00.");
   System.out.print("Withdrawl amount: ");
   amount = input.nextDouble();
}

System.out.printf("Here are your %.2f dollars.", amount);
```

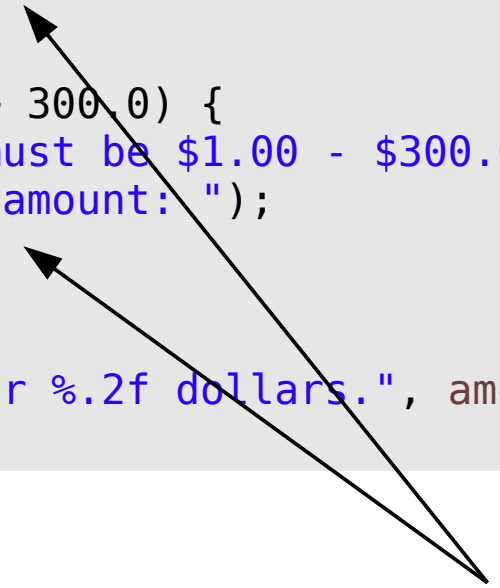
While Loop for Input Validation

- Use a while loop to keep asking *while* the user still hasn't entered a valid number:

```
System.out.print("Withdrawl amount: ");
amount = input.nextDouble();

while (amount < 1.0 || amount > 300.0) {
   System.out.println("Amount must be $1.00 - $300.00.");
   System.out.print("Withdrawl amount: ");
   amount = input.nextDouble();
}

System.out.printf("Here are your %.2f dollars.", amount);
```

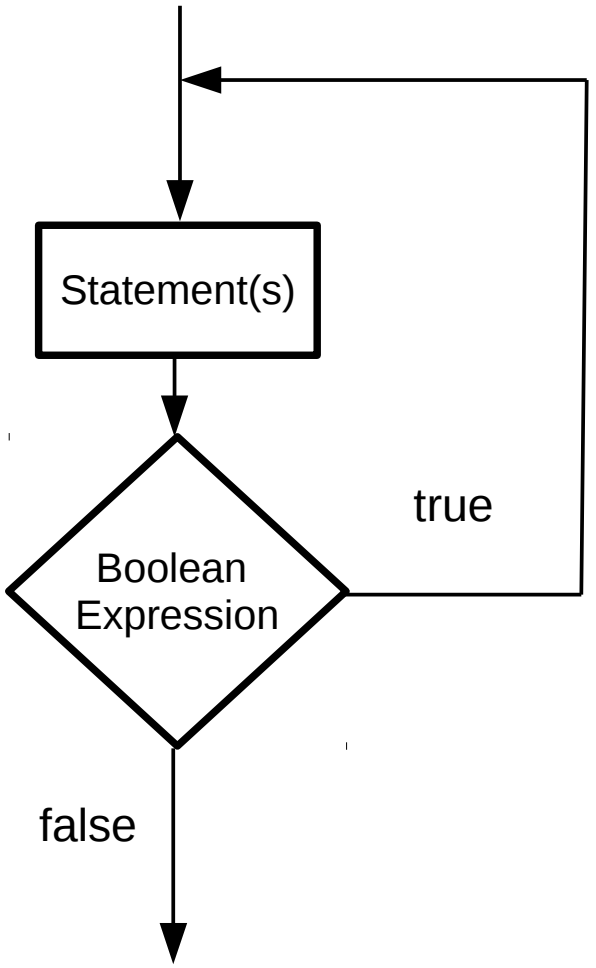


Ugly that we repeat this code

Do-While Loops

```
do {
   Statement(s)
} while (BooleanExpression);
```

- Referred to as a *post-test loop*, because the test is performed after they loop body



Do-While Loop for Input Validation

- No more code repetition:

```
do {

   System.out.println("Amount must be $1.0 - $300.00");
   System.out.print("Withdrawl amount: ");
   amount = input.nextDouble();

} while (amount < 1.0 || amount > 300.0);

System.out.printf("Here are your %.2f dollars.", amount);
```

Counting Loops

- Common to write loops that execute some fixed number of times:

```
int frame = 1;

while (frame <= 10)
{
   // Get bowling scores for this frame.
   // Do some fancy calculations.
   // Show a turkey animation if needed...
   frame++;
}
```

Counting Loops

- Common to write loops that execute some fixed number of times:

```
int frame = 1;

while (frame <= 10)
{
   // Get bowling scores for this frame.
   // Do some fancy calculations.
   // Show a turkey animation if needed...
   frame++;
}
```

We need to look in *three* different places to figure out what this loop is doing.

For Loops

- For loops provide more concise syntax for the same logic:

```
int frame = 1;

while (frame <= 10)
{
   // Get bowling scores for this frame.
   // Do some fancy calculations.
   // Show a turkey animation if needed...
   frame++;
}
```

```
for (int frame = 1; frame <= 10; frame++)
{
   // Get the latest scores.
   // Do some fancy calculations.
   // Show a turkey animation if needed...
}
```