We Continue

GEEN163

“I have always wished for my computer to be as easy to use as my telephone; my wish has come true because I can no longer figure out how to use my telephone.”

Bjarne Stroustrup

creator of C++

Second Exam

- The second exam in COMP163 will be on Wednesday, October 17
- The exam will cover topics since the first exam
  - IF statements
  - Loops
  - Reading and writing files

`for` Loop

- The `for` loop has three parts separated by semicolons
  - **initialization** – Done once before the loop
  - **test** – Loop executed when test is true
  - **update** – Done at the end of the loop before test

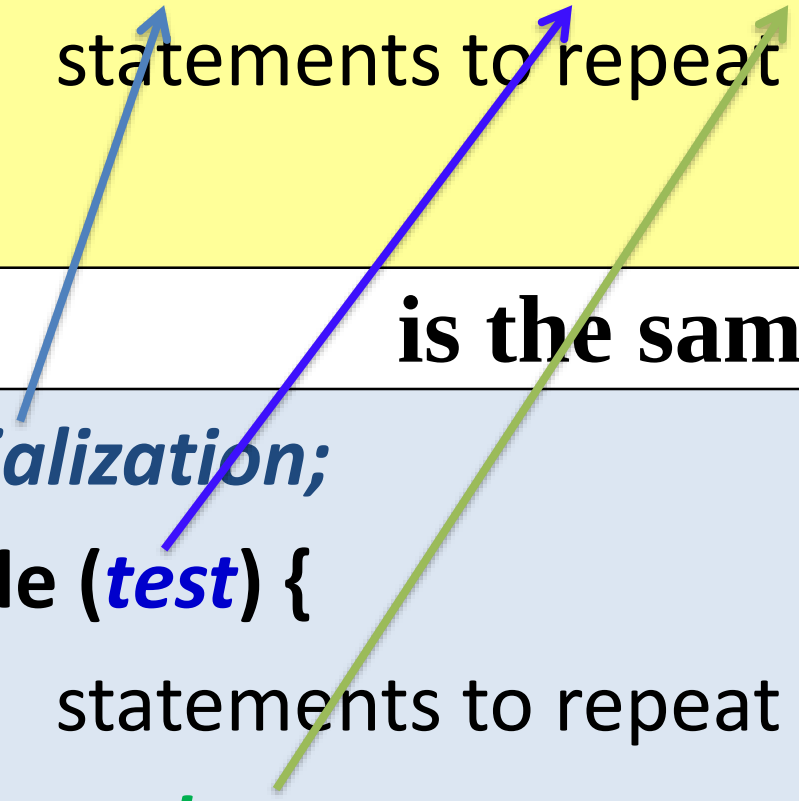
```
for ( initialization; test ; update ) {
    // do something
}
```

`for` and `while` loop equivalence

```
for ( initialization ;  test; update )  {
        statements to repeat
}
```

is the same as

```
initialization;
while (test) {
        statements to repeat
        update;
}
```



do while version

- To read 7 integers with a do while loop

```
Scanner keys = new Scanner(System.in);
int  sum = 0, count;
count = 0;
do {
  sum = sum + keys.nextInteger();
  count++;
} while(count < 7);
```

while version

- To read 7 integers with a while loop

```
Scanner keys = new Scanner(System.in);
int  sum = 0, count;
count = 0;
while(count < 7){
  sum = sum + keys.nextInteger();
  count++;
}
```

for version

- To read 7 integers with a for loop

```
Scanner keys = new Scanner(System.in);
int  sum = 0, count;
for (count = 0; count < 7; count++){
  sum = sum + keys.nextInteger();
}
```

Print numbers 1 to 5 and square roots

```
double dog;
for (double cat = 1.0; cat < 6.0; cat++) {
        dog = Math.sqrt(cat);
        System.out.println(cat +" "+ dog);
}
```

Counting Down

- The following program segment displays the numbers from 8 to 0 in reverse order

```
for (int goat = 8; goat >= 0; goat--){
    System.out.println( goat );
}
```

Declaring the Loop Counter

- In a for loop, you can declare the variable that is used as the loop counter
- The loop counter can only be used in the loop

```
for (int bull = 0; bull < 47; bull ++) {
        // loop body
        // the loop counter, bull, can be used here
}
// You cannot use bull after the loop
```

What does this print?

```
int dog = 1;
for (int cat = 2; cat < 4; cat++) {
    dog = dog + cat;
}
System.out.print( dog );
```

A. 1

B. 2

C. 3

D. 6

E. 10

Loops in Loops

- Sometimes a program will need one loop in the body of another loop

```
int sum = 0, cat = 5, dog = 3;
for (int i = 0; i < cat; i++) {
    for (int k = 0; k < dog; k++) {
        sum += i * k;
    }
}
```

- Note that the loop counter variables (i and k) must be different

Repeating Inner Loop

```
for (int cat = 0; cat < 3; cat++ ) {
    for (int dog = 0; dog < 2; dog++ ){
        System.out.println("cat " + cat +" dog "+dog);
    }
}
```

```
cat 0  dog 0
cat 0  dog 1
cat 1  dog 0
cat 1  dog 1
cat 2  dog 0
cat 2  dog 1
```

What does this program display?

```
int count = 0;
for (int i = 0; i < 2; i++) {
    for (int k = 0; k < 2; k++) {
        count = count + 1;
    }
}
System.out.println( count );
```

A. 0
B. 2
C. 4
D. 5
E. 6

Selecting from Multiple Choices

- The if - else if statements are a way of selecting from multiple choices

```
if ( dog > big ) {
      // if dog is big
} else if ( cat > big ) {
      // if dog is small and cat is big
} else if ( bird > big ) {
      // if dog and cat are small and bird is big
} else {
      // if dog, cat and bird are small
}
```

Write Java with the students near you

- Set the String variable **`weather`** to "calm" if wind is less than 3.0, "strong" if wind is greater than 24.0 and "breeze" otherwise

```
double wind = something;
String weather;
```

Possible Solution

- Set the String variable weather to "calm" if wind is less than 3.0, "strong" if wind is greater than 25.0 and "breeze" otherwise

```
double wind = something;
if (wind < 3.0) {
        weather = "calm";
} else if (wind > 25.0) {
        weather = "strong";
} else {
        weather = "breeze";
}
```

What is displayed?

```
double wind = 2.0;
if (wind < 3.0) {
        weather = "calm";
}
if (wind > 25.0) {
        weather = "strong";
} else {
        weather = "breeze";
}
System.out.println(weather);
```

A. calm
B. strong
C. breeze
D. 3.0
E. 25.0

With added else

```
double wind = 2.0;
if (wind < 3.0) {
        weather = "calm";
} else
if (wind > 25.0) {
        weather = "strong";
} else {
        weather = "breeze";
}
System.out.println(weather);
```

Output is

calm

With added else and formatting

```
double wind = 2.0;
if (wind < 3.0) {
        weather = "calm";
} else if (wind > 25.0) {
        weather = "strong";
} else {
        weather = "breeze";
}
System.out.println(weather);
```

Output is
calm

Changing the Loop Counter

- You can change the loop counter just like any other variable
- This will usually have an impact on the number of times a loop repeats

```
for (int count = 0; count < 5; count++) {
    // something
    count = 47;
}
```

break statement

- The break statement stops a loop
- The break statement can be used in for loops, while loops, do-while loops and switch statements
- A break statement causes the loop to stop looping and starts executing after the loop
- break statements are almost always inside the body of an if statement

break example

```
int sum = 0;
for (int cow = 0; cow < 5; cow++) {
        sum = sum + cow * cow;
        if ( sum > 10)
                break;
}
System.out.println(sum);    // prints 14
```

cow	sum
0	0
1	1
2	5
3	14
4	30

What is Displayed?

```
int apple = 1, pear;
for(pear = 1; pear<4; pear++) {
   if (apple > 3 ) {
      break;
   }
   apple += pear;
}
System.out.println(apple+" "+pear);
```

A. 1 1

B. 4 2

C. 4 3

D. 6 4

E. none of the above

What is Displayed?

```
int apple = 1, pear;
for(pear = 1; pear<4; pear++) {
   apple++;
   if (apple > 3 ) {
      break;
   }
   apple += pear;
}
System.out.println(apple+" "+pear);
```

A. 1 1

B. 4 2

C. 4 3

D. 6 4

E. none of the above

Break in Searching

- Consider a program that is looking for an ID and a name

```
int id, goal=123;
String name;
java.io.File fileObj = new java.io.File("data.txt");
java.util.Scanner inFile = new java.util.Scanner(fileObj);
while (inFile.hasNextInt()) {
        id = inFile.nextInt();
        name = inFIle.nextLine();
        if (id == goal) break;
}
System.out.println(id +" "+name);
```

Loops in Loops

- If you have loops inside other loops, a break statement will stop the looping of the inner most loop where the break is executed
- Outer loops will not be effected by a break in an inner loop

Loop in a Loop

```
int sum = 0;
 for (int i = 1; i < 3; i++ ) {
  for (int k = 1; k < 4; k++ ) {
    System.out.println("i:"+i+"  k:"+ k + "  sum: "+sum );



    sum = sum + k;
  }
 }
System.out.println( "final sum: "+sum );
```

```
i:1  k:1  sum: 0
i:1  k:2  sum: 1
i:1  k:3  sum: 3
i:2  k:1  sum: 6
i:2  k:2  sum: 7
i:2  k:3  sum: 9
final sum: 12
```

Break in a Loop in a Loop

```
int sum = 0;
 for (int i = 1; i < 3; i++ ) {
  for (int k = 1; k < 4; k++ ) {
    System.out.println("i:"+i+"  k:"+ k + "  sum: "+sum );
    if (sum > 2 ) {
      System.out.println("  break i: "+i+" k: "+k+" sum:"+sum);
      break;
    }
    sum = sum + k;
  }
 }
System.out.println( "final sum: "+sum );
```

```
i:1  k:1  sum: 0
i:1  k:2  sum: 1
i:1  k:3  sum: 3
  break i: 1 k: 3 sum:3
i:2  k:1  sum: 3
  break i: 2 k: 1 sum:3
final sum: 3
```

Labeled Statements

- You can put a label before any statement
- Labels start the Java statement and are separated from the statement by a colon
- Label names follow the usual rules

```
fred : for (int cat = 0; cat < 100; cat++) {
```

```
mary : while ( dog < 13 ) {
```

break to a label

- To avoid problems with loops in loops, you can put a label on the loop and then specify the label on the break

```
fred : for (int cat = 0; cat < 13; cat ++ ) {
                    if (cat * 2 > 20 )
                                break fred;
          }
```

- When you have loops in loops, this specifies exactly which loop you are terminating

break Labels

- The label on a break statement must indicate a loop statement (for, do or while) that encompasses the break

```
tom: for (int i = 0; i < 47; i++ ) {
        sum += i * i;
        if (sum > 100)  break george; // ERROR
}
sum++;
george: System.out.println("after loop");
```

Nested Loop Break

```
int cow, bull, sum = 0;
daisy: for (cow = 1; cow <= 3; cow++ ) {
    ferdinand: for (bull = 1; bull <= 5; bull++ ) {
        sum += cow * bull;
        if (sum > 25) break daisy; // ends both loops
    }
}
```

How useful are copies of the slides?

A. Very useful
B. Useful
C. Somewhat useful
D. No value at all
E. Better to have exact slides on the web

How useful is recitation?

A. Very useful
B. Useful
C. Somewhat useful
D. No value at all
E. I don't go to the recitation class

How useful is the textbook?

A. Very useful
B. Useful
C. Somewhat useful
D. No value at all
E. I do not have a textbook

How useful is the lab?

A. Very useful
B. Useful
C. Somewhat useful
D. No value at all
E. I don't go to the labs

How helpful are the lab Teaching Assistants?

A. Very helpful
B. Helpful
C. Somewhat helpful
D. No help at all
E. I don't go to the labs

continue Statement

- The **`continue`** statement makes the loop go the beginning of the loop and start the next iteration

```
for (int cow = 0; cow < 100; cow++) {
        // something
        if ( cow < 47 || cow > 53 )
                continue;
        // some other thing
        System.out.println( cow );
}
```

This prints 47 to 53

Continue in a Loop in a Loop

```
int sum = 0;
 for (int i = 1; i < 3; i++ ) {
  for (int k = 1; k < 4; k++ ) {
    System.out.println("i:"+i+"  k:"+ k + "  sum: "+sum );
    if (sum > 2 ) {
      System.out.println("  break i: "+i+" k: "+k+
                                          " sum:"+sum);
      break;
    }
    sum = sum + k;
  }
 }
System.out.println( "final sum: "+sum );
```

```
i:1  k:1  sum: 0
i:1  k:2  sum: 1
i:1  k:3  sum: 3
  continue i: 1 k: 3 sum:3
i:2  k:1  sum: 3
  continue i: 2 k: 1 sum:3
i:2  k:2  sum: 3
  continue i: 2 k: 2 sum:3
i:2  k:3  sum: 3
  continue i: 2 k: 3 sum:3
final sum: 3
```

What is Displayed?

```
int apple = 1, pear;
for(pear = 1; pear<4; pear++) {
   apple++;
   if (apple > 3 ) {
      continue;
   }
   apple += pear;
}
System.out.println(apple+" "+pear);
```

A. 1 1

B. 4 3

C. 5 4

D. 6 4

E. none of the above

It would be better to write the loop as

```
int apple = 1, pear;
for(pear = 1; pear<4; pear++) {
   apple++;
   if (apple <= 4 ) {
      apple += pear;
   }
}
System.out.println(apple +" "+pear);
```

Labels on Continue

- Just like the break statement, you can put labels on a continue statement
- The label must be on a surrounding loop, either for, while or do while

Labels are Recommended

- Many Java programs do not use labels on the break or continue statements
- Many Java programs have errors due to misunderstood break and continue statements
- The use of labels makes your intentions clear
- The compiler will tell you if you are doing something wrong

Software Failure

Due to a programmer's misunderstanding of how a break statement works, a NASA probe crashed into the surface of Venus

Modify this to average all the numbers up to a sum of more than 100

```
public class SumFile {
    public static void main(String[] args) throws
                    java.io.FileNotFoundException {
      java.io.File dog = new java.io.File("myData.txt");
      java.util.Scanner cat = new java.util.Scanner(dog);

       double sum = 0.0;                  // sum of numbers
       int    count = 0;                  // count of numbers
       while (cat.hasNextDouble()) {
          sum += cat.nextDouble();
          count++;
       }
       System.out.println("Average is "+sum/count);
    }
}
```

Hint

```
public class SumFile {
    public static void main(String[] args) throws
                    java.io.FileNotFoundException {
      java.io.File dog = new java.io.File("myData.txt");
      java.util.Scanner cat = new java.util.Scanner(dog);

       double sum = 0.0;                  // sum of numbers
       int    count = 0;                  // count of numbers
       while (cat.hasNextDouble() &&                    ) {
          sum += cat.nextDouble();
          count++;
       }
       System.out.println("Average is "+sum/count);
    }
}
```

Possible Solution

```
public class SumFile {
    public static void main(String[] args) throws
                    java.io.FileNotFoundException {
      java.io.File dog = new java.io.File("myData.txt");
      java.util.Scanner cat = new java.util.Scanner(dog);

       double sum = 0.0;                  // sum of numbers
       int    count = 0;                  // count of numbers
       while (cat.hasNextDouble() && sum < 100.0) {
          sum += cat.nextDouble();
          count++;
       }
       System.out.println("Average is "+sum/count);
    }
}
```

Possible Solution

```
public class SumFile {
    public static void main(String[] args) throws
                    java.io.FileNotFoundException {
      java.io.File dog = new java.io.File("myData.txt");
      java.util.Scanner cat = new java.util.Scanner(dog);
       double sum = 0.0;                  // sum of numbers
       int    count = 0;                  // count of numbers
       readLoop:while (cat.hasNextDouble()) {
          sum += cat.nextDouble();
          count++;
          if (sum > 100.0) {
             break readLoop;
          }
       }
       System.out.println("Average is "+sum/count);
    }
```

Second Exam

- The second exam in COMP163 will be on Wednesday, October 17
- The exam will cover topics since the first exam
  - IF statements
  - Loops
  - Reading and writing files