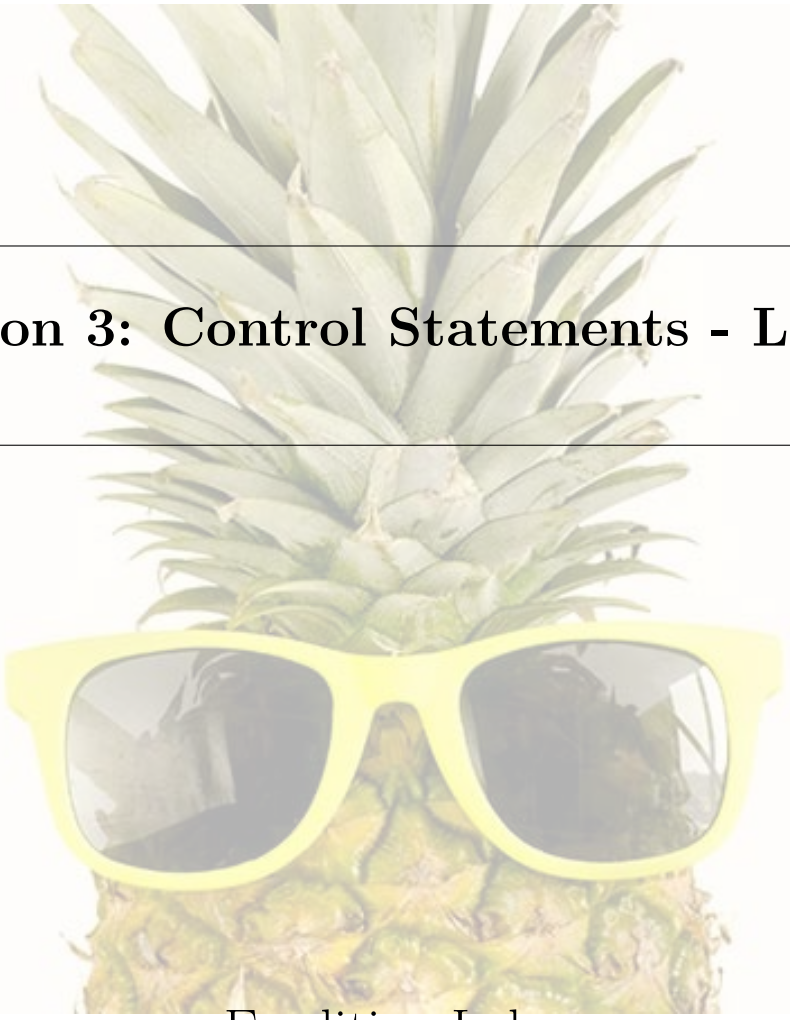

Section 3: Control Statements - Loops

Erudition Labs

Computer Science 101: Introduction to Java and Algorithms

May 13, 2019

Contents

1 Terminology

- ***Break out of a loop*** – We often use this to describe that we are done with the loop and need to stop iterating, aka exit the loop. This often occurs when the condition evaluates to false or we manually use the "break" statement to exit the loop.
- ***Iterate*** – When we talk about iteration or iterate, it is often in reference to loops. One iteration is one time through a loop.

2 Pre-Chapter

2.1 ++ Incrementor and −− Decrementor (Might as well do += and -=)

When programming it is often a common task to increment or decrement things, such as a counter, by 1. In fact it is so common that most languages, including java, have a short hand notation to do it. We also often need to perform some operation on a variable and then save it in that same variable, so there is shorthand notation for that as well. For example,

```
int counter = 0;

counter = counter + 1; //long way to increment by one
counter += 1; //shorter way
counter++; //shortest way

counter = counter - 1;
counter -= 1;
counter--;
```

The first three are equivalent and the last three are equivalent.

3 Loops

Loops are used for iterating over collections and counting. At the moment you don't have anything to iterate over (loop over or loop through). In this section we are introducing the syntax and some basic things you can do with what we have gone over so far.

4 While and Do-While Loop (Video Series Lecture 16 and 17)

While loops are most useful when you don‘t need to keep count of where you are at in a list. Although you certainly can.

4.1 While

As seen in the video series, a while loop looks like this

```
while(CONDITION_IS_TRUE) {
   //do this stuff
}
```

For example, if we wanted to print “Hello“ five times, we could use a counter and our relational operators.

```
public class Hello {

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

       while(counter < 5) {
           System.out.println("Hello");
           counter++;
       }
   }
}
```

4.2 Do-While

The do-while loop is honestly kind of useless. I don‘t think I have ever used it in any production code. But it does exist and you might find it helpful depending on how you think of things. The do-while loop always iterates at least once. It will go through your block of code first and then check your condition. Whereas with the while loop, it will check the condition first and then execute your block of code.

As seen in the video series, it looks like this

```
do {
   // execute code in block
}
while (conditionIsTrue);
```

5 For Loop (Video Series Lecture 18 and 19)

The for loop is the most commonly used loop, at least in my experience. The loop maintains a counter, a condition and the incrementation of the counter. As seen in the video series,

```
for (statement 1; statement 2; statement 3) {
  // code block to be executed
}
```

For example, to count to 5,

```
for (int i = 0; i < 5; i++) {
  System.out.println(i);
}
```

Let's break it down. The "for" is the keyword for a "for loop". The Parenthesis enclose the parameters of the for loop.

```
int i = 0; //initializes the counter i to 0
i < 5 // The condition of the loop, as long as this evaluates to true, your block of code will be
    executed
i++ // increments the counter each time the code block is executed.
```

Note that the counter variable does not need to be i. You can name it anything you like. The best way to get familiar with loops is to just use them.

6 Example

Let's write a program that prints out all of the integers from 1 to 100 (inclusive) that are divisible by 5. First let's talk about what we need. First we want to use a loop to iterate from 1 to 100. For each iteration, we need to look at the counter and check if it is divisible by 5. If it is, then print it out, otherwise, continue the loop. So inside out loop, we will need an if-statement. How do check if the counter is divisible by 5? Well, we can use a little bit of math. We can divide the number by 5 and see if there is a remainder. If there is no remainder, then it is obviously divisible by 5. Luckily, Java gives us the modulus operator to get the

remainder.

6.1 Using A For Loop

```
public class Fives {
    public static void main(String []args) {
      for(int i=1; i<=100; i++) {
          if(i % 5 == 0) {
              System.out.println(i);
          }
      }
   }
}
```

Note that we start from 1 and we use <= as out relational operator. This is because we want to include 100 in our list.

6.2 Using A While Loop

```
public class Fives {
    public static void main(String []args) {
       int i = 1;
       while(i <= 100) {
          if(i % 5 == 0) {
              System.out.println(i);
          }
          i++;
       }
   }
}
```

Note that we have to create the incrementor and increment it ourselves in the right spot.

7 Gotchas With Loops

7.1 Infinite Loops

The most common problems that you will have when using loops is the infinite loop. This means that the condition is never met. This usually happens when you either wrote the condition wrong or you forgot to

increment the incrementor. The program will just run until your computer runs out of memory (or at least the memory that the java virtual machine allows your program to have).