COMP 110
Introduction to Programming

Fall 2015
Time: TR 9:30 – 10:45
Room: AR 121 (Hanes Art Center)

Jay Aikat
FB 314, aikat@cs.unc.edu

Previous Class

• What did we discuss?
Today

• Announcements
  • Quiz today – boolean logic and if-else
  • Assignment 1 : Due Tuesday, Sep 22 @ 11:55 PM
    http://cs.unc.edu/~aikat/courses/comp110/assignments/Assignment1

  • HACKER110: goto ➔
    http://comp110.com/hacker

• More If-else

---

Multibranche If-Else Statement

• Example
  – Write a program that takes as input your year in college (as an integer) and outputs your year as freshman, sophomore, junior, senior, or super senior
Multibranch If-Else Statement

- Flow chart:

```java
if (year == 1)
    System.out.println("freshman");
else {
    if (year == 2)
        System.out.println("sophomore");
    else {
        if (year == 3)
            System.out.println("junior");
        else {
            if (year == 4)
                System.out.println("senior");
            else {
                if (year == 5)
                    System.out.println("super senior");
                else
                    System.out.println("huh?");
            }
        }
    }
}
```

Multibranch If-Else Statement

- We can write a program like this
Multibranche If-Else Statement

- Because the previous version is too tedious, we use the **multibranch** statement instead
  - It is not a new syntax rule. We only ignore the brackets so that the logical structure is clear.

```java
if (year == 1)
    System.out.println("freshman");
else if (year == 2)
    System.out.println("sophomore");
else if (year == 3)
    System.out.println("junior");
else if (year == 4)
    System.out.println("senior");
else if (year == 5)
    System.out.println("super senior");
else
    System.out.println("huh?");
```

Multibranche If-Else Statement

- Though all the branches look equal, there is a precedence order among them
  - Only the **first** satisfied branch will be executed
Multibranche If-Else Statement

• What’s wrong with this piece of code?

```java
if (num < 50)
    System.out.println("Number is less than 50");
else if (num < 25)
    System.out.println("Number is less than 25");
else
    System.out.println("Number is greater than 50");
```

Multibranche If-Else Statement

• What’s wrong with this piece of code?

```java
if (num < 50)
    System.out.println("Number is less than 50");
else if (time < 25)
    System.out.println("Number is less than 25");
else
    System.out.println("Number is greater than 50");
```

Will this branch get executed?
Nested If and Else

```java
if (time < 7){
    if (time < 6){
        go to the gym;
    }
    else{ // note the corresponding “if”
        have brkfst and leave;
    }
}
else{
    go to school;
}
```

• What’s the logic flow?
  – If the time is less than 6, we go to the gym;
  – If the time is between 6 and 7, we eat breakfast and leave
  – If the time is greater than 7, we simply sprint to school!

Nested If and Else

```java
if (time < 6){
    go to the gym;
}
else{
    if (time < 7){
        have brkfst and leave;
    }
    else{
        go to school;
    }
}
```

• What’s the logic flow?
  – If the time is less than 6, we cook breakfast;
  – If the time is between 6 and 7, we get something cold
  – If the time is greater than 7, we go to school
Same Logic, Different Code

```java
if (time < 6)
    go to the gym;
else{
    if (time < 7)
        have brkfst and leave;
    else{
        go to school;
    }
}
if (time < 7)
    if (time < 6)
        go to the gym;
    else{
        have brkfst and leave;
    }
else{
    go to school;
}
```

Nested If-Else Statement

• Without brackets, every `else` will automatically match the nearest `if`

```java
if (num < 50)
    if (num < 25)
        System.out.println("Number is less than 25");
    else
        System.out.println("Number is greater than 50");
```

• Is this piece of code correct?
Nested If-Else Statement

• Without brackets, every else will automatically match the nearest if

```java
if ( num < 50 ) {
    if ( num < 25 )
        System.out.println("Number is less than 25");
    else
        System.out.println("Number is between 25 and 50");
}
```

• Use brackets and indentation to avoid such errors

Nested If-Else Statement

• Without brackets, every else will automatically match the nearest if

```java
if ( num < 50 ) {
    if ( num < 25 )
        System.out.println("Number is less than 25");
} else
    System.out.println("Number is greater than 50");
```

• Use brackets and indentation to avoid such errors
Notes…

• Note: To exit your program before end of main method
  
  System.exit (0);

---

LOOPS

• Loops are designed to repeat instructions
  
  – Think about the requirement: Print number 1 to 10
    
    • It’s easy
      – System.out.println("1");
      – System.out.println("2");
      – ......
    
    – Think about the requirement: Print number 1 to 100
      • We can still do this
    
    – Let the user input a value n, then print 1 to n
      • We are in trouble......
Loop Statement

- What is the pseudo code to fulfill the requirement?
  - Count to 1, if 1<=n, write it down, otherwise stop
  - Count to 2, if 2<=n, write it down, otherwise stop
  - Count to 3, if 3<=n, write it down, otherwise stop
  - ....
  - Count to i, if i<=n, write it down, otherwise stop
  - Count to i+1, if i+1<=n, write it down, otherwise stop
  - ....
  - While a counter<=n, write it down, increase the counter. Otherwise stop

While Statement

- Flow of while statement
  - Start from expression evaluation
  - As long as it’s true, repeat instructions in brackets

```java
while (count <= number) {
    System.out.println(count);
    count++;
}
```
While Statement

• You have to do some initialization before the statement

• The loop body typically contains an action that ultimately causes the controlling boolean expression to become false.

```java
number = keyboard.nextInt();
count = 1;
while (count <= number) {
    System.out.println(count);
    count++;
}
```

While Statement

• Usually there is a counter variable in the statement
  — You can use it in different ways

• Requirement: print the odd numbers from 1 to 10000

```java
int count = 1;
while (count < 10000) {
    System.out.println(count);
    count += 2;
}
```

```java
int count = 1;
while (count * 2 - 1 < 10000) {
    System.out.println(count * 2 - 1);
    count++;
}
```
Infinite Loops

• Always make sure that your loop will end
  – Never forget to change the counter

```java
while (count <= number) {
    System.out.println(count);
}
```

• Always make sure that your loop will end
  – Never forget to change the counter
  – Use comparison rather than “==” or “!=“in the control expression
  – Know whether your counter is increasing or decreasing

```java
while (count != number) {
    System.out.println(count);
    count+=2;
}
```
```java
while (count < number) {
    System.out.println(count);
    count--;
}
```
Infinite Loops

- If you wrote an infinite loop and executed it
- Use the **terminate** button of eclipse
  - If it is red, the program is **running**

Infinite loop is not a syntax error. It’s a logical error
- eclipse will not help you in this case
- Write pseudo code, think, and rethink before coding
Next class

- More loops

→ Reading Assignment: Chapter 4