Loop

Cheng, Wei  COMP110-001  May 21, 2014
Review of Previous Lecture & Lab

• Q1 What output is produced by the following statements?
  ```java
  int number = 7;
  boolean isPositive = (number > 0);
  if (number > 0);
      number = -100;
  if (isPositive)
      System.out.println("Positive.");
  else
      System.out.println("Not positive.");
  System.out.println(number);
  ```

• Q2 Write a program that assigns grade based on score:
  - A: 90 ~ 100
  - B: 80 ~ 89
  - C: 70 ~ 79
  - D: 60 ~ 69
  - F: 0 ~ 59
Review of Previous Lecture & Lab

• Use parenthesis for each branch. But simplify it when you can.
• Remember the if-else-if form for multiple parallel tests

if (year==1) {
    System.out.println("Freshman");
} else {
    if (year==2) {
        System.out.println("Sophomore");
    } else {
        if (year==3) {
            System.out.println("Junior");
        } else {
            System.out.println("Senior");
        }
    }
}

if (year==1) {
    System.out.println("Freshman");
} else if (year==2) {
    System.out.println("Sophomore");
} else if (year==3) {
    System.out.println("Junior");
} else {
    System.out.println("Senior");
}
When working on nested if-else

• Write out the complete outer level first!
• Then proceed to each branch and write out inner level.

```java
if (input.equals("mouth")) {
    Mouth branch
} else if (input.equals("nose")) {
    Nose branch
} else if (input.equals("eyes")) {
    Eyes branch
}
```

```java
if (input.equals("mouth")) {
    s.setMouthStartAngle(0);
} else if (input.equals("nose")) {
    Nose branch
} else if (input.equals("eyes")) {
    Eyes branch
}
```
Today’s Contents

• Loop

Warm-up so far in this course.
Loop is where it starts to get harder.
I encourage you to spend more time on the examples given in this & future lectures.
Flow of control

• There are multiple types of control:

✓ – continuation (unconditional)
✓ – executing a set of statements only if some condition is met (choice)

→ – executing a set of statements zero or more times, until some condition is met (loop)

– ...
Elements of loop

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

• **while loop**
  – Simplest loop form

• **do-while loop**
  – Not covered in this course

• **for loop**
  – Most expressive loop form
  – You will be using this most of the time
while loop

```java
while (Boolean_Expression) {
    Loop body statements...
}
```

- 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 loop

```plaintext
while (Boolean_Expression){
    Loop body statements...
}
```

Diagram:
- **Evaluate** Boolean Expression
  - true → **Execute Body**
  - false → **End loop**
Execution Flow

**Iteration 1:** (boolean expression is true)

```java
while (Boolean_Expression) {
    Loop body statement 1
    Loop body statement 2
    Loop body statement 3
}

...
Execution Flow

Iteration 2: (boolean expression is true)

```java
while (Boolean_Expression) {
    Loop body statement 1
    Loop body statement 2
    Loop body statement 3
}
...
Execution Flow

Iteration 3: (boolean expression is false)

while (Boolean_Expression)
{
  Loop body statement 1
  Loop body statement 2
  Loop body statement 3
}
...

Some Examples

```java
while(true) {
    System.out.println(“I am still running”);
}

while(false) {
    System.out.println(“I am still running”);
}
```
More Examples

```java
int counter = 0;
while (counter < 5) {
    System.out.println(counter);
    counter = counter + 1;
}
```

• What is the output?
More Examples

```java
int counter = 0;
while (counter < 5) {
    System.out.println(counter);
    counter = counter + 1;
}
```

- At the while statement (where counter value is tested)
  - Iteration 1: counter(0) < 5 is true
  - Iteration 2: counter(1) < 5 is true
    ...
  - Iteration 5: counter(4) < 5 is true
  - Iteration 6: counter(5) < 5 is false
More Examples

```java
int counter = 0;
while (counter < 5) {
    counter = counter + 1;
    System.out.println(counter);
}
```

• What is the output?
More Examples

```java
int counter = 0;
while(counter < 5) {
    counter = counter + 2;
    System.out.println(counter);
}
```

- What is the output?
Calculate the sum of 1...100

```java
int counter = 1;  // this is the initial state
int sum = 0;

while(counter <= 100) {
    sum = sum + counter;
    counter = counter + 1;
}

System.out.println("The sum is " + sum);
```
Simple input checking

Scanner s = new Scanner(System.in);
int input = -1;

while (input < 0 || input > 100) {
    System.out.println("Please input a score between 0 and 100");
    input = s.nextInt();
}

The few lines will keep reading user input until we have a number between 0 and 100.
Early Exit

break;

Exit from the loop and continue to what’s after.
Calculate factorial with break

```java
int counter = 1; // this is the initial state
int factorial = 1;
int n = 100; // computer the factorial of 1...n

while (counter <= n) {
    factorial = factorial * counter;
    counter = counter + 1;
    if (Integer.MAX_VALUE / factorial < counter) {
        System.out.println("We have to stop before it explodes!");
        break;
    }
}
```

.................
Continuation with next iteration

continue;

to skip the remainder of the loop body and continue with the next iteration of the loop.
Calculate the sum of multiples of 3

```c
int counter = 0;
int sum = 0;

while (counter <= 100) {
    counter = counter + 1;
    if (counter % 3 != 0) {
        continue;
    }
    sum = sum + counter;
}
```

..............................
Calculate the sum of multiples of 3

What’s wrong with this implementation?

```c
int counter = 1;
int sum = 0;

while(counter <= 100) {
    if ( counter % 3 != 0) {
        continue;
    }
    counter = counter + 1;
    sum = sum + counter;
}
```

..............................
Next Class

- More on loop
- For loop
- Nested loop
- More complicated examples