iteration (loops)
lecture 11 (7/21/2006)

iteration motivation
• iteration means doing something over and over and over again... often using the results from the previous step in the next step.
• why are loops important?
• sometimes you don’t want to type a section of code over and over and over and over and over and over and over and over and over and over and over and over and over and over and over and over and over and over and over and over and over and over and over and over and over and over and over and over and over and over and over...

useless fact of the day
• three jokes by the late mitch hedberg are:
  • this guy asked me if i wanted a frozen banana, and i said, “no... but i would like a regular banana later, so... yeah.”
  • i like escalators because they can never break. they can only become stairs.
  • i haven’t slept for ten days, because that would be too long.

iteration motivation
• loops are also useful because they let you:
  • group repeatedly executed code, for uniformity and readability
  • make the number of repetitions easily changeable
  • repeat events where the number of executions is known only dynamically (when the program’s running)
  • combine with selection statements to make complex algorithms
while Loop

initialization

int \( i = 0; \)
while ( \( i < 10 \) ) condition
{
    System.out.println(i);
    i++;
}
update

while Loop: Strangeness

- Consider:
  - We expect it to output 0, then 0.1, then 0.2... until 0.9, and then stop, since \( i \) should be 1.0...........

not-equal operator

double \( i = 0; \)
while ( \( i \neq 1 \) )
{
    System.out.println(i);
    i = i + 0.1;
}

for Loop

initialization condition update

for \( ( \text{int } i = 0; \ i < 10; \ i++ ) \)
{
    System.out.println(i);
}

• An infinite loop?? What happened????

• The computer doesn’t actually store numbers in decimal (base-10) format; it stores them in binary (base-2) -- even thought it shows you the numbers in base-10 when you print them out

• The computer can’t actually represent “0.1” (1/10) exactly in binary! It is 0.00011 (just like 1/3 can’t be represented exactly in decimal -- it is 0.3).

• The error accumulates to the point where we never exactly hit “1.0” by adding a bunch of “0.1”s
**for Loop**

- Printing out 0.1 through 0.9 without a problem:

  ```java
  for ( int i = 0; i < 10; i++ )
  {
      System.out.println(i*0.1);
  }
  ```

- Why? Our condition checks based on the integer value of `i`, which goes from 0 to 9 without error.

**Equivalence of Loops**

```java
int i = 0;
while ( i < 10 )
{
    System.out.println(i);
    i++;
}
```

**Other Loops**

- **do-while loop**
  - Not used very often...
  - Runs the body of the loop before checking the condition; hence the body always runs at least once.

  ```java
  int i = 0;
  do
  {
      System.out.println(i);
      i++;
  } while(i<=10);
  ```

- Special kind of **for** loop...We'll talk about it when we talk about Collections.
When to Use **for** and **while**

Is it known how many times the loop will execute prior to executing the loop body?

- **no**
- **yes**

**while**

**for**

- But really, just use *whichever one is more convenient.*