Flow of Control: Loops (Savitch, Chapter 4)

TOPICS

• while Loops
• do while Loops
• for Loops
• break Statement
• continue Statement

An Example while Loop

```c
int count = 1;
int sum = 0;
while (count < 5)
{
    sum += count;
    count++;
}
```

What exactly does this code do?

Step-by-step

```c
int count = 1;
int sum = 0;
while (count < 5)
{
    sum += count;
    count++;
}
```

Code begins

Count = 1, sum = 0
Try again loop
Re-enter loop: 10

Bottom of loop:
Count = 5, sum = 10

More formally:  *while* Loops

```
while (condition)
  body
```

- Repeatedly executes as long as the *condition* evaluates to *true*
- *body* of the loop is a single statement or multiple statements within {}
- The condition is tested before the body is executed, so loop may execute zero times
  - This is called a *pre-test* loop

---

Echo Example Program

```java
import java.util.Scanner;

public class Foo {
    public static void main(String[] args) {
        Scanner in_str = new Scanner(System.in);
        String user_string = in_str.next();
        while (!user_string.equals("quit")) {
            System.out.println(user_string);
            user_string = in_str.next();
        }
    }
}
```

---

Echo Example: Notes

- “*import java.util.Scanner;*” is necessary to use a Scanner.
  - Problem: Without it, Eclipse will tell you it cannot resolve the Scanner class.
  - Solution: ctrl-shift-o will import needed classes.
- Remember that “!” means “not” in Java.
- Note the indentation: the body of the while loop is indented relative to the surrounding code.

---

Echo Example: Questions

- How many times will the loop body execute?
  - Undetermined: it will keep executing until the user types “quit”
- What is the fewest number of times the loop body could execute?
  - Zero
Warning!

• An infinite loop will occur if the condition never becomes false.
• Example:

```java
int count = 1;
int sum = 0;
while (count <= 5)
{
    sum += count;
}
```

What if my program gets caught in an infinite loop?

• You will need to kill your program
  – This is operating system specific
• Make your life easier: run your program in the debugger!
  – In Eclipse, select “debug” instead of “run”.
  – It will offer to take you to the debug view.
  – Use the red button to kill the program.
  – Benefit: Run program step-by-step (F5).

Another example: find divisors

```java
public class foo {
    public static void main(String[] args) {
        int number = Integer.parseInt(args[0]);
        int divisor = 2;
        while (divisor < number ) {
            if ((number % divisor) == 0) {
                System.out.print(divisor + " ");
            }
            divisor = divisor + 1;
        }
    }
}
```

Notes on divisor example (1)

• The main method takes an array of strings (called arguments or args).
  – args[0] is the first string passed to the method
  – args[1] would be the second string
  – args.length tells you how many strings there are
  – More about arrays later...
Notes on divisor example (2)

• Integer is an object class in Java. It has a method that reads a string and returns the integer it contains. Hence
  `Integer.parseInt(args[0]);`
• We append a space to the number when printing, so that the numbers are separated in the output.

Divisor example questions

• If the argument is ’32’, how many times will the loop body be executed?
  – 30
• If the argument is ’2’, how many times will the loop body be executed?
  – 0
• If the argument is ’-5’, what will happen?
  – The loop body will run 0 times

Example Program to Remove Vowels

```java
public class Foo {
    public static void main(String[] args) {
        String str = args[0];
        int ctr = 0;
        while (ctr < str.length()) {
            switch(str.charAt(ctr)) {
                case 'a':
                case 'e':
                case 'i':
                case 'o':
                case 'u':
                    break;
                default: System.out.print(str.charAt(ctr));
            }
            ctr = ctr + 1;
        }
    }
    ...
}
```

Remove Vowels: Notes

• The `charAt(i)` method of String returns the i-th character.
  – Zero-based: 0, 1, 2, ...
• The `length()` method of String returns the number of characters in the string.
Remove Vowels: Questions

- If the input is “Programming”:
  - How many times will the loop body execute?
    - 11
  - What will the output be?
    - Prgrmmng
- If the input is “Java”:
  - How many times will the loop body execute?
    - Exercise
  - What will the output be?
    - Exercise

for Loop

- It is common to iterate counter number of times.
  - counter might be a numeric bound
    - As in the divisor example
  - counter might be the length of a string or array
    - As in the remove vowels example
- A for loop gives you a mechanism to specify this explicitly

for Loop: Syntax

\[
\text{for (initialization; condition; update)} \quad \text{body}
\]

- A pre-test loop that:
  - Initializes a loop variable
  - Executes body of loop zero or more times
  - Repeatedly:
    - Tests the condition
    - Executes the body if condition is true, else exits loop
    - Updates the loop variable

for Loop: Order

```java
for( initialization ; booleanExpression ; incremernter )
{
 statements;
}
```
Example

```java
int sum = 0;
for (int count = 1; count <= 5; count++)
    sum += count;
```

Mapping between `for` and `while`

- **while loop version**
  - initialization;
  - while (condition)
    - statement;
    - update;
  - }
- **for loop version**
  - for (initialization; condition; update )
    - statement;

Temperature Conversion Program

```java
System.out.println("\tDEGREES C\tDEGREES F");

for (int cent = 50; cent <= 100; cent++)
{
    double fahr = (9.0 / 5.0) * cent + 32.0;
    System.out.print("\t" + cent);
    System.out.print("\t" + fahr);
}
```

Example: Reversing a String

```java
String s = "nice string";
for (int i = s.length() - 1; i >= 0; i--)
{
    System.out.print(s.charAt(i));
}
```

What happens if we use `println` instead of `print`?
Variants on the for Loop

- Multiple variables in for loop

```java
int x = 1;
for( int lo = 0, hi = 10; lo < hi; lo++, hi-- )
    System.out.println(x++);
```

- Not all parts to the for loop

```java
String s = "Javarules";
int i = s.length() - 1;
for ( ; i>=0; )
    System.out.print( s.charAt(i--) );
```

do while Statement

```java
do
{
    body
} while (condition);
```

- post-test loop: always executes the loop body at least once
- Executes again as long as its condition is true
- {} are required
- ; required after while

Example

```java
int count = 1;
int sum = 0;
do
{
    sum += count;
    count++;
} while (count <= 5);
```

How does this differ from the previous?

Mapping between do while and while

- do while version

```java
do
{
    statement;
} while (condition);
```

- while version

```java
statement;
while (condition);
```
Which loop to use?

if (you know the # of iterations)
  – use a for loop
else if (statements should be done at least once)
  – use a do...while loop
else
  – use a while loop

Problem Solving and Formulating Loops

• Stepwise Development:
  – Break problem into subparts
  – Identify repeating pattern in problem formulation
  – Put pattern into body of loop
  – Identify a continuing condition (or termination condition) that concerns what is being updated in the body of the loop
  – Make separate loops when multiple patterns are found; make nested loops when one pattern fits within another.

Example: Reading Input from User

• Strategy:
  – Ask user for input
  – Do something with the input
  – Ask user for input
  – Do something with the input
  – ...
  – Until user no longer has input to enter

• Questions:
  – How does user indicate no more input?
  – What is the pattern?
  – What is terminating condition?

Reading Input Using a Loop

```java
Scanner in = new Scanner(System.in);
int score = 0, sumOfScores = 0;
while (true) { // sentinel
    System.out.println("Enter score [or -1 for end of input]: ");
    score = in.nextInt();
    if (score == -1) break; // sentinel
    sumOfScores += score;
} // sentinel
System.out.println("Sum of scores was "+ sumOfScores);
```

When the user is entering a set of data, you need some way for them to say “no more” -- called a sentinel.
In other words ...

- **Stepwise refinement**
  - don’t do everything at once
  - identify sub-tasks and work on one at the time

- **Identify loop patterns**
  - the repeated behavior
  - what is to be done before the loop
    - e.g., initialization
  - how is loop termination decided
  - what needs to be done after the loop
    - e.g., store or print results

---

### Nested Loops

- Write the code to print out the following:

  ```
  *
  **
  ***
  ****
  *****
  ******
  *******
  ********
  *********
  **********
  ```

  **ALGORITHM**
  - OUTER LOOP: 10 times (10 rows)
  - INNER LOOP: 1 to outer loop counter

```java
public class Stars {
    public static void main(String[] args) {
        for (int c = 1; c <= 10; c++) {
            for (int i = 0; i < c; i++) {
                System.out.print("* ");
            }
            System.out.println();
        }
    }
}
```  

---

### Cautions about Loops

- Ensure that the required precision of your condition matches that of the type (recall that two doubles may be *mathematically* equal but not in the computer!)
- Use `{ }` for multiple statements
- Check for off-by-1 errors (make sure that it is ending at the right time)
- Do NOT put a ; at the end of a for() or while() !!!
- In a while loop, the condition must be testable prior to executing the body
- In any loop, ensure that the update will eventually cause variable(s) in the condition to cause the condition to become false.
Cautions about Loops

• Check for off-by-1 errors (make sure that it is ending at the right time)

```java
for ( int i=1; i<100; i++ )
{
    System.out.print( "*" );
}
```

Prints 99 stars. Why?

• **Infinite Loops**: loop with a conditional that never becomes false:

```java
while( true )
    computeSquares();
```

```java
for ( int i=1; i>0; i++ )
    processOutput();
```

```java
x = 1;
while( x < 10 )
    x = x + 5;
```

```java
y = 1;
while( y < 10 )
    System.out.print( y );
y++;
```

Cautions about Loops

• Do NOT put a ; at the end of a for() or while() !!!
Declares an empty body for the loop. Therefore the statements you think are in the body of the loop actually aren’t

```java
for ( int i=0; i<100; i++ );
```

```java
for ( int i=0; i<100; i++ )
{
    System.out.print( "*" );
}
```

Prints ONE star! Why?

Infinite Loop

Programming Practice

• Run your loops by hand (pencil and paper)
  – Write out expectations, check them if needed
• Don’t use **break** and **continue** in loops
  – They get very confusing very fast
• **Echo values of variables**
  ```java
  System.out.println("Str: " + Str);
  ```
• **Useful identifiers**
  – no one-letter identifiers, except for loop indices
• **Declare variables in the right scope**
  – Often at top of scope is good
• **Give yourself a chance to succeed**
  – Don’t start your project on day before the deadline
Loop Practice Problems

- Find the minimum integer in a sequence of integers
- Find the maximum in a sequence of integers
- Find the longest word in a sequence of words
- Determine if a word is a palindrome