CS 302: Introduction to Programming

Lectures 7&8

Hopefully the Programming Assignment #1 released by tomorrow

Review

- The switch statement is an alternative way of writing what?
- How do you end a case in a switch statement?
- What is a better way of writing:

```
if (b == false) { //assume b is a declared and initialized boolean
  //do something
}
```

- if (1 + 2 < 3 || !(5 > 2*2)) evaluates to?

BRACES = { }

- Used to group blocks of code
- So far we use them:
  - Group code in a class
  - Group code in method
  - Group code for an if/else if/else block
  - Group code for a switch statement
- Code inside braces is indented with a tab (eclipse usually does this automatically for you)
- To auto format:
  - Windows: control-a control-i
  - Mac: apple-a apple-i

Loops

- Sometimes we need to run the same code multiple times, but we don't want to copy and paste our code
- Solution: loops
  - 3 Types:
    - For
    - While
    - Do while
- <u>The for loop is used when you know from the start how many times you want to loop to run</u>

For Loop

- Simple example: print numbers 1 – 5
- Could just have 5 System.out.println() calls
- Or:
- for (int i = 1; i < 6; i++)
- {

  System.out.println(i);
- }

For Loop

- For loops are based around a loop control variable
- Parts of a for loop
  - for reserved word
  - Initialization of loop control variable
  - Condition checking to see if the loop should continue to run
  - Updating the loop control variable

```
for (int i = 1; i < 6; i++)
{
    System.out.println(i);
}
```

THE UNIVERSITY of WISCONSIN MADISON

Execution of a for loop

- 1. Initialize i = 1
- 2. Check condition – is i < 6?
- 3. Execute Code within braces
- 4. Update i – i now equals 2
- Repeat from step 2 – check condition – is i < 6?

```
for (int i = 1; i < 6; i++)
{
    System.out.println(i);
}
```

FOR LOOP

- Can also count down instead of up

- Don't have to increment by 1

```
for (int i = 6; i > 1; i--)
{
    System.out.println(i);
}
for (int i = 1; i < 6; i *= 2)
{
    System.out.println(i);
}
```

Scope

- Variables declared within the loop are available only within the loop and are re-initialized every time the loop runs

```
for (int i = 1; i < 6; i++)
{
 int x = 1;
 x++;

   System.out.println(x);
}
System.out.println(x);
System.out.println(i);
```

THE UNIVERSITY of WISCONSIN MADISON

SCOPE

- If you want to use variables in and out of the loop, declare them before the loop

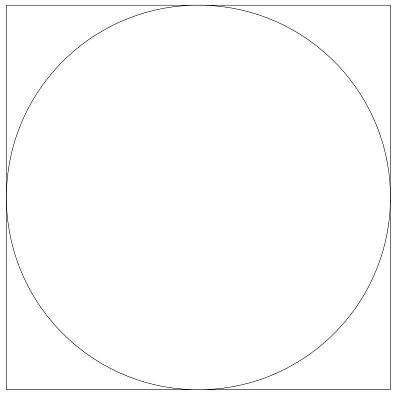
```
int x = 0;
int i;
for (i = 1; i < 6; i++)
{
 x = 1;
 x++;

   System.out.println(x);
}
System.out.println(x);
System.out.println(i);
```

How many iterations for the following loops?

- for (int i = 0; i <= 10; i++)
- for (int i = -2; i > -10; i--)
- for (int i = 0; i < 100; i += 2)
- for (int i = -2; i < 10; i *= -1)

Practice 1



- Generate 100,000 random dots in a square inside which there is a tangential circle, and output the ratio of the number of dots appearing in the circle to the total number of random dots. Origin is set at the center of the circle. Diameter of the circle is 1 (side length of the square is also 1)
- Hint: Pythagorean equation

Break and Continue

-
  ```
  for (int i = 0; i < 10; i++)  {
      if (i == 5) {
          break;
      }

      S.O.P(i);
  }
  ```

- Use “break” when you need to exit a loop immediately

-
  ```
  for (int i = 0; i < 10; i++)  {
      if (i == 5) {
          continue;
      }

      S.O.P(i);
  }
  ```

- Use “continue” to proceed with next iteration immediately

Practice 2

- Palindrome
- Use Scanner object to read in a string
- Test whether the string is a palindrome.

Loops

- Sometimes we need to run the same code multiple times, but we don't want to copy and paste our code
- Solution: loops
  - 3 Types:
    - For
    - While
    - Do while
- <u>While loops are used when you have no idea how many times the loop will run</u>

While Loop

- Used when we don't necessarily know from the start how many times the loop will run
- We want to run it 0, 1, or many times

```
while (boolean expression)
{
  //code to execute for each iteration
}
```

While Loop vs For Loop

Simple example: print numbers 1 – 5

```
for (int i = 1; i < 6; i++)
{
   System.out.println(i);
}
```

```
int i = 1;
while (i < 6)
{
   System.out.println(i);
   i++;
}
```

While Loop Examples

Finding the first match:

```
String str = in.nextLine();
boolean found = false;
char goal = '.';
int position = 0;
```

```
while (!found && position < str.length())
{
   char currChar = str.charAt(position);
   if (currChar == goal) { found = true; }
   else { position++; }
}
```

OR

```
while (position < str.length())
{
   char currChar = 
str.charAt(position);
   if (currChar == goal) { 
break; }
   position++;
}
```

Ifs vs While Loops

- If statements
  - Executes 0 or 1 times

```
if (boolean expression)
{
    //code
}
```

- Loops
  - Execute 0, 1 or many times

```
while (boolean
    expression)
{
    //code
}
```

Loops

- Sometimes we need to run the same code multiple times, but we don't want to copy and paste our code
- Solution: loops
  - 3 Types:
    - For
    - While
    - Do while
- <u>Do while loops are used when you know the loop should run at least once.</u>

Do while Loop

- Used when we want to run the loop at least once but maybe more
- We want to run it 1 or many times

```
do
{
   //code to execute for each iteration
} while (boolean expression);
//Remember there is a ; at the end of the while!
```

THE UNIVERSITY of WISCONSIN MADISON

For vs While vs Do while

Simple example: print numbers 1 – 5

For:

```
for (int i = 1; i < 6; i++)
{
  System.out.println(i);
}
```

Do:

```
int i = 1;
do
{
   System.out.println(i);
   i++;
} while (i < 6);
```

While:

```
int i = 1;
while (i < 6)
{
  System.out.println(i);
  i++;
}
```

Do While Loop Examples

Reading input:

```
double weight = -1;
do
{
     System.out.print("Enter Weight (>= 0): ");
     if (in.hasNextDouble())
         weight = in.nextDouble();
    else {
         System.out.println("Please enter a double");
         in.nextLine(); //Need to clear the input buffer!
        }
} while (weight < 0);
```

Nested Loops

- Loops can be nested in the same way if statements can be
- Remember that code will run in sequence

Ex.

```
for (int i = 0; i < 3; i++)
{
    for (int j = 0; j < 4; j++)
    {
        System.out.println("Hello");
    }
}
```

How many times will “Hello“ be printed?

Scanner Annoyances (Demo)

- From the API:
  - next(), nextInt(), nextDouble() skip any input that matches the delimiter pattern and then attempt to return the next token (word/int/double)
  - nextLine() - Advances this scanner past the current line and returns the input that was skipped. This method returns the rest of the current line, excluding any line separator at the end. The position is set to the beginning of the next line.
  - Eclipse Demo for next(), nextInt(), nextDouble() and nextLine()

PRACTICE 3

- Write a program to assist an instructor in calculating grades for a single student
- Input: multiple scores (each out of 100), -1 to quit
  - Example input: 77(then hit enter) 90(then hit enter) 57(…) 88(…) -1(…)
- Output:
  - The number of scores entered
  - The sum of all scores entered
  - Highest single score
  - Average score
- Don't count the -1 as a valid score!
- 1). Validate input type (must be int); 2). Scores must be 0<= scores <= 100 which should be checked by the program!!