U N I V E R S I T Y

CS 180: Problem Solving and Object-Oriented Programming

Lecture 5: Selection and Formatting

Dr. Jeff Turkstra

Lecture 05

- Integer division
- Control flow
- If-then
- Short circuiting
- If-then-else
- Ternary operator
- Format specifiers
- Nesting

Integer division

- Operations involving only integers always result in an integer
  - Accomplished through truncation
  - Any remainder or decimal portion is discarded

Control flow statements

- Statements are generally executed from top to bottom
  - Unless there is an error (“exception”)
- Control flow statements alter the flow of execution
- Conditional execution

Control flow statements

- Decision-making
  - if-then, if-then-else, switch
- Looping
  - for, while, do-while
- Branching
  - break, continue, return

if-then

- Execute a certain block only if an expression evaluates to true
- Remember booleans?

```
if (<boolean expression> == true)
  if-statement;
else
  else-statement;
```

- == true is implicit

boolean type

- Result of any comparison
- Operators are:
  - == and !=
  - Logical complement !
  - Logical &, ^, and |
  - Conditional-and/or && and ||
  - Conditional operator ?
  - String concatenation +
    - "true" or "false"
- Only expression usable for control flow
  - if, while, do, for, conditional ?

Operator results

p	q	p && q	p || q	!p	p ^ q
false	false	false	false	true	false
false	true	false	true	true	true
true	false	false	true	false	true
true	true	true	true	false	false

Reference types

- Remember: comparing reference types (e.g., Strings) only compares the references to the objects, not the objects themselves


```
int i, j;
byte b, c;
float f, g;
double d, e;

i < j     f >= i       d > 9.3      2 == c       j != i   g <= (b*c + d)

(i > j) && (f >= i)    (d > 9.3) || (2 != d)     !(c <= j) ^ (j != i)

((i > j) && (f >= i)) || ((d > 9.3) || (2 != d)) ^ (!(c <= j) ^ (j != i))
```


Short-circuit

- Conditional operators (&& and ||) exhibit "short-circuiting" behavior
- Second operand is evaluated only if needed

Abstracting conditions

- If it is a weekday and I am not on vacation, then I will get up early. Otherwise I will get up whenever.

```
if (isWeekday && !onVacation)
  getUpEarly();
else
  getUpWhenever();
```

if-then-else

- Secondary path of execution when if evaluates to false
- Once a condition is satisfied, remaining conditions are not evaluated

- If there is a basketball game on and Purdue is playing, I'll cheer for Purdue; otherwise, if IU is playing, I'll cheer for their opponent

```
if (gameOn(basketball) && playing(purdue))
  cheerFor(purdue);
else if (gameOn(basketball) && playing(iu))
  cheerFor(opponent(iu));
```

Problem: SecretWord

- Write a program that reads a word from the user and prints a message if it matches a "secret word" in the program.

Solution

```
import java.util.Scanner;

public class SecretWord {
  final static String SECRET = "awesome";

  public static void main(String[] args) {
    Scanner in = new Scanner(System.in);
    String word = in.next();

    if (word.equals(SECRET))
      System.out.printf(
        "You have said the secret word: '%s'\n",
        SECRET);
  }
}
```


Problem: Absolute Value

- Write a program that illustrates how to convert the value in a variable x to the absolute value using an if statement


```
import java.util.Scanner;

public class AbsVal {
  public static void main(String[] args) {
    Scanner in = new Scanner(System.in);
    int x = in.nextInt();

    System.out.printf("BEFORE: x = %d\n", x);

    if (x < 0)
      x = -x;

    System.out.printf("AFTER: x = %d\n", x);
  }
}
```

Ternary operator

```
int value;
if (condition)
  value = a;
else
  value = b;
```

- Can be written as:

```
int value = (a < b) ? a : b;
```

- Only use it if it makes your code more readable
  - When expressions are compact
  - No side effects

Blocks

- Conditionals apply to a single statement or block
  - Remember {…}?
- Using braces for singleton statements is optional
  - Not doing it can make code more brittle

Quadratic

- Write a method…

```
void printRoots(double a, double b, double c)
```

- …that finds and prints the roots of a quadratic equation
  - Include imaginary roots


```
// Reference: http://www.1728.org/quadratc.htm

public class Quadratic {
  void printRoots(double a, double b, double c) {
    double d = b * b - 4 * a * c;

    if (d < 0) {
      double x = -b/(2*a);
      double x1 = Math.sqrt(-d)/(2*a);

      System.out.printf("%.2f+%.2fi and %.2f-%.2fi are imaginary " +
                        "roots of %.2fx^2 + %.2fx + %.2f\n",
                        x, x1, x, x1, a, b, c);
    }
    else {
      double x1 = (-b + Math.sqrt(d))/ (2 * a);
      double x2 = (-b - Math.sqrt(d))/ (2 * a);

      System.out.printf("%.2f and %.2f are real roots of %.2fx^2 + " +
                        "%.2fx + %.2f\n",
                        x1, x2, a, b, c);
    }
  }

  public static void main(String[] args) {
    Quadratic q = new Quadratic();
    q.printRoots(3, 4, 5);
    q.printRoots(2, 4, -30);
    q.printRoots(12, 5, 3);
  }
}
```


Purdue trivia

- "While students, future author George Ade and cartoonist John McCutcheon faced the wrath of President James H. Smart for attending a ladies' literary society meeting without faculty permission, and McCutcheon got fired as editor of the student newspaper."

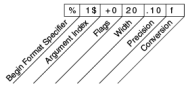
  - A Century and Beyond, by Robert W. Topping

Formatting numeric print output

- System.out.printf() or System.out.format()
  - Format string + optional arguments

```
System.out.printf("a = %2d", a);
```

Format specifiers



Conversions

- General – any argument type: b, B, h, H, s, S
- Character: c, C
- Numeric
  - Integral: d, o, x, X
  - Floating point: e, E, f, g, G, a, A
- Date/Time: t, T
- Percent (%%)
- Line separator (%n)

Precision

- General: maximum number of characters to be written
- Floating-point (e, E, f): number of digits after the decimal separator
  - g, G: total number of digits in the resulting magnitude after rounding
  - Not valid with a and A
- Not applicable for character, integral, date/time

Width

- Minimum number of characters to be displayed (padding)

Flags

- Left-justified: -
- Always include sign: +
- Leading space for positive values: ‘ ’ (space)
- Zero-padded: 0
- Enclose negatives in parentheses: (
- Conversion-dependent alternate format: #
- Locale-specific grouping separators: ,

Argument index

- Decimal integer that indicates position of the argument in the argument list
  - 1$, 2$, etc
  - Can also use < re-use argument from the previous format specifier

Problem: DaisyDriveIn

- If you work more than 20 hours at the Daisy Drive-In, you are paid $8/hr for the first 20 hours plus $10/hr for all hours beyond 20. Otherwise, you are paid $7/hr
- Write a method that returns the correct pay…

```
double computePay(double hours)
```



```
public class DaisyDriveIn {
  double computePay(double hours) {
    if (hours > 20)
      return 8.00 * 20 + (hours - 20) * 10.00;
    else
      return hours * 7.00;
  }

  public static void main(String[] args) {
    DaisyDriveIn d = new DaisyDriveIn();
    double pay;

    pay = d.computePay(20); // pay should be 140
    pay = d.computePay(21); // pay should be 170
    pay = d.computePay(9.5); // pay should be 66.5
    pay = d.computePay(9.1); //pay should be 63.7
  }
}
```


Problem: Swapper

- Write a program that, given two values in variables x and y, ensures that y is not less than x


```
import java.util.Scanner;

public class Swapper {
  public static void main(String[] args) {
    Scanner in = new Scanner(System.in);

    int x = in.nextInt();
    int y = in.nextInt();

    System.out.printf("BEFORE: x = %d, y = %d\n", x, y);

    if (y < x) {
      int t = x;
      x = y;
      y = t;
    }

    System.out.printf("AFTER: x = %d, y = %d\n", x, y);
  }
}
```

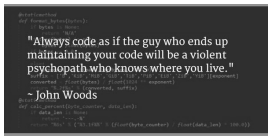

Nested if statement

- Then and else blocks in an if statement can contain any valid statement(s)
  - Including if statements

```
if (chLevel > 239)
  System.out.println("… is too high.");
else
  if (chLevel > 199)
    System.out.println("… is mildly high.");
  else
    System.out.println("… is normal.");
```


Dangling else

```
if (chLevel > 199)
  if (chLevel > 239)
    System.out.println("Too high");
else
  System.out.println("Normal");
```



"Always code as if the guy who ends up maintaining your code will be a violent psychopath who knows where you live."

~ John Woods

Boiler Up!