Topic 12
Conditional Execution

"We flew down weekly to meet with IBM, but they thought the way to measure software was the amount of code we wrote, when really the better the software, the fewer lines of code."

-Bill Gates

Based on slides for Building Java Programs by Reges/Stepp, found at http://faculty.washington.edu/stepp/book/

Conditional execution with if statements

The if statement

- Programs that read user input often want to take different actions depending on the values of the user's input.
- **if statement**: A Java statement that executes a block of statements only if a certain condition is true.
  - If the condition is not true, the block of statements is skipped.
  - General syntax:

```
if (<condition>) {
    <statement(s)> ;
}
```

  - Example:

```
double gpa = console.nextDouble();
if (gpa <= 2.0) {
    System.out.println("Your application is denied.");
}
```

The if/else statement

- **if/else statement**: A Java statement that executes one block of statements if a certain condition is true, and a second block of statements if the condition is not true.
  - General syntax:

```
if (<condition>) {
    <statement(s)> ;
}
else {
    <statement(s)> ;
}
```

  - Example:

```
double gpa = console.nextDouble();
if (gpa <= 2.0) {
    System.out.println("Your application is denied.");
}
else {
    System.out.println("Welcome to Mars University!");
}
```

Relational expressions

- The ***<condition>*** used in an `if` or `if/else` statement is the same kind of value seen in the middle of a `for` loop.
  - `for (int i = 1; i <= 10; i++)`

- These conditions are called **relational expressions**.
- Relational expressions use one of the following six **relational operators**:

Operator	Meaning	Example	Value
`==`	equals	`1 + 1 == 2`	`true`
`!=`	does not equal	`3.2 != 2.5`	`true`
`<`	less than	`10 < 5`	`false`
`>`	greater than	`10 > 5`	`true`
`<=`	less than or equal to	`126 <= 100`	`false`
`>=`	greater than or equal to	`5.0 >= 5.0`	`true`

Evaluating relational expressions

- The relational operators can be used in a bigger expression with the mathematical operators we learned earlier.
- Relational operators have a lower precedence than mathematical operators, so they are evaluated last.
  - Example:

```
5 * 7 >= 3 + 5 * (7 - 1)
5 * 7 >= 3 + 5 * 6
35    >= 3 + 30
35    >= 33
true
```

- Relational operators cannot be "chained" in the way that you have seen in algebra.
  - Example:

```
2 <= x <= 10
true   <= 10
   error!
```

Nested if/else statements

- **Nested if/else statement**: A chain of `if/else` that can select between many different outcomes based on several conditions.
  - General syntax (shown with three different outcomes, but any number of `else if` statements can be added in the middle):

```
if (<condition>) {
    <statement(s)> ;
} else if (<condition>) {
    <statement(s)> ;
} else {
    <statement(s)> ;
}
```

  - Example:

```
int grade = console.nextInt();
if (grade >= 90) {
    System.out.println("Congratulations!  An A!");
} else if (grade >= 80) {
    System.out.println("Your grade is B.  Not bad.");
} else if (grade >= 70) {
    System.out.println("You got a C.  Work harder!");
}
```

Structures of if/else code

- Choose 1 of many paths:
  (use this when the conditions are mutually exclusive)

```
if (<condition>) {
    <statement(s)>;
} else if (<condition>) {
    <statement(s)>;
} else {
    <statement(s)>;
}
```

- Choose 0 or 1 of many paths:
  (use this when the conditions are mutually exclusive and any action is optional)

```
if (<condition>) {
    <statement(s)>;
} else if (<condition>) {
    <statement(s)>;
} else if (<condition>) {
    <statement(s)>;
}
```

- Choose 0, 1, or many of many paths:
  (use this when the conditions/actions are independent of each other)

```
if (<condition>) {
    <statement(s)>;
}
if (<condition>) {
    <statement(s)>;
}
if (<condition>) {
    <statement(s)>;
}
```

How to comment: if/else

- Comments on an if statement don't need to describe exactly what the if statement is testing.
  - Instead, they should describe why you are performing that test, and/or what you intend to do based on its result.
  - Poor style:

```
// Test whether student 1's GPA is better than student 2's
if (gpa1 > gpa2) {
    // print that student 1 had the greater GPA
    System.out.println("The first student had the greater GPA.");
} else if (gpa2 > gpa1) {
    // print that student 2 had the greater GPA
    System.out.println("The second student's GPA was higher.");
} else {
    // there was a tie
    System.out.println("There has been a tie!");
}
```

  - Good style:

```
// Print a message about which student had the higher grade point average.
if (gpa1 > gpa2) {
    System.out.println("The first student had the greater GPA.");
} else if (gpa2 > gpa1) {
    System.out.println("The second student's GPA was higher.");
} else {  // gpa1 == gpa2 (a tie)
    System.out.println("There has been a tie!");
}
```

How to comment: if/else 2

- If an if statement's test is straightforward, and if the actions to be taken in the bodies of the if/else statement are very different, sometimes putting comments on the bodies themselves is more helpful.
  - Example:

```
if (guessAgain.equals("y")) {
    // user wants to guess again; reset game state and
    // play another game
    System.out.println("Playing another game.");
    score = 0;
    resetGame();
    play();
} else {
    // user is finished playing; print their best score
    System.out.println("Thank you for playing.");
    System.out.println("Your score was " + score);
}
```

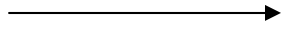
Math.max/min vs. if/else

- Many if/else statements that choose the larger or smaller of 2 numbers can be replaced by a call to `Math.max` or `Math.min`.
  -
```
int z;          // z should be larger of x, y
if (x > y) {
    z = x;
} else {
    z = y;
}
```
  - **`int z = Math.max(x, y);`**
  -
```
double d = a;  // d should be smallest of a, b, c
if (b < d) {
    d = b;
}
if (c < d) {
    d = c;
}
```
  - **`double d = Math.min(a, Math.min(b, c));`**

Factoring if/else code

- **factoring**: extracting a common part of code to reduce redundancy
  - factoring if/else code reduces the size of the if and else statements and can sometimes eliminate the need for if/else altogether.
  - example:

```
int x;
if (a == 1) {
    x = 3;
} else if (a == 2) {
    x = 5;
} else {  // a == 3
    x = 7;
}
```



```
int x = 2 * a + 1;
```

Code in need of factoring

- The following example has a lot of redundant code in the if/else:

```
if (money < 500) {
    System.out.println("You have, $" + money + " left.");
    System.out.print("Caution!  Bet carefully.");
    System.out.print("How much do you want to bet? ");
    bet = console.nextInt();
} else if (money < 1000) {
    System.out.println("You have, $" + money + " left.");
    System.out.print("Consider betting moderately.");
    System.out.print("How much do you want to bet? ");
    bet = console.nextInt();
} else {
    System.out.println("You have, $" + money + " left.");
    System.out.print("You may bet liberally.");
    System.out.print("How much do you want to bet? ");
    bet = console.nextInt();
}
```

Code after factoring

- If the beginning of each if/else branch is essentially the same, try to move it out *before* the if/else. If the end of each if/else branch is the same, try to move it out *after* the if/else.

```
System.out.println("You have, $" + money + " left.");

if (money < 500) {
    System.out.print("Caution!  Bet carefully.");
} else if (money < 1000) {
    System.out.print("Consider betting moderately.");
} else {
    System.out.print("You may bet liberally.");
}

System.out.print("How much do you want to bet? ");
bet = console.nextInt();
```

Practice methods

- Write a method to determine the max value, given three integers
- Write a method determines if 2 points form a line and if so if it is a vertical line, horizontal line, or neither
- Write a method to determine the number of times the character 'm' occurs in a String
- Generalize the previous method to determine the number of times any given character occurs in a String
- Write a method that determines the last index of a character in a given String