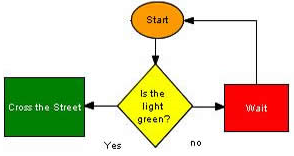
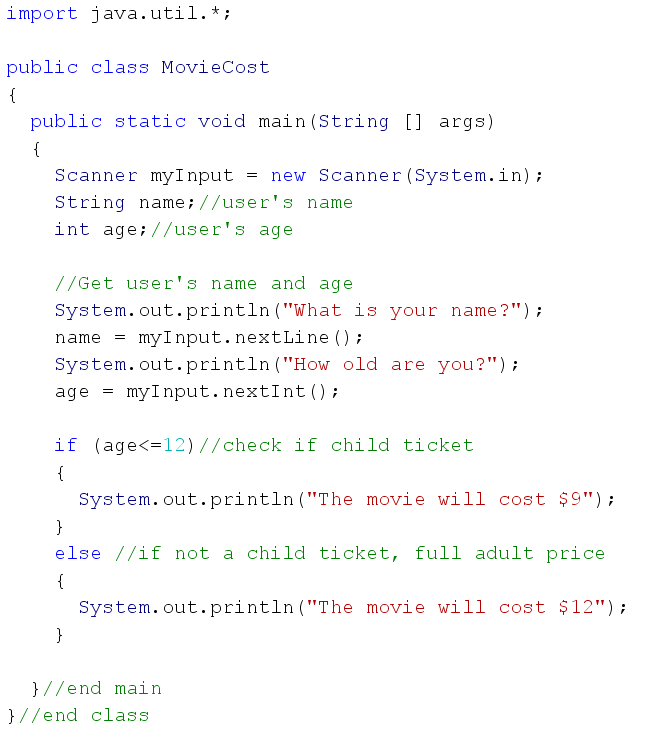
We make simple decisions all the time. Like what to do at a traffic light. Depending on the status of the light, we decide what instructions to follow. The flowchart to the right illustrates our decision-making pattern.  
  


**Conditional Statements**

The two common condition constructs in Java are the **if** statement and **switch**. A properly constructed condition statement allows the program to appear to make decisions by allowing data to be "sifted" through a series of comparison statements. If a comparison is ***true*** (the comparison is called a *boolean* because the response is either true or false), the sequence of events is executed. If the comparison is ***false***, the data continues to flow through the statement. The simplest example is known as a two-way" decision structure; however, this statement can easily be edited to perform more complex, "multi-layered" decisions.

**If Statements**

The traffic light example is a two-way decision sometimes called a "binary" decision. In programs complicated decisions are made of many small decisions. Here is a program that includes a binary decision. The program will output the cost of a movie ticket given the age of person attending.

****

*Example Notes:*

* To use the **if** statement, the condition must be inside () brackets. In this program, if the person inputs an age that is less than, or equal to, 12 the true branch will be executed; otherwise, the false branch is executed.
* The prompt statement uses, System.out.print() rather than System.out.println()**.** This reserves space at the end of the line rather than a carriage return, so that user input can occur on the same line as the prompt.
* Whatever path the decision follows, the program resumes immediately after the **else** statement.
* Also, note that there are multiple statements to be executed within the body of the **if** statement. You must have a set of braces { } enclosing them; otherwise, only the first statement will be executed. Note the changes in the programming fragment below:

**TASK #1**

**Type in the program, compile and check for errors, and run it to see that it works correctly. What values did you try entering for the age to test your code? What happened? What other cases might you want to try to make sure that it always works?**

**TASK #2**

**The theatre also offers a discount for seniors. Anyone age 65 or over pays $10. Make this change and test it. Show me your new program.**

**TASK #3**

**Add a line to each case where the person’s age is repeated back to them. For instance if the user inputs 16, the output would be:  
 > If you are 16 years old it is a regular price ticket.**

**>The movie will cost $12.00**

**>Enjoy the show!**

**Make this change and test it. Show me your new program.**

*Example Notes:*

* The **else if** statement has the same basic syntax as the **if** statement. It contains a comparison statement to allow the program to determine if the condition is either ***true*** or ***false***.
* The statements are executed in sequence. If the first condition is false, the next condition is tested, and so on.
* *You can include as many* ***else if*** *statements as necessary*; however when 3 or more conditional statements are needed, a ***switch*** statement is more efficient.
* Arrange your condition statements so they logically cascade from one to another.

**Error Traps**

Multi-way selection structures also allow the programmer to incorporate input entry "*error traps*". These traps can be inserted to catch invalid input from the user. Such statements normally appear at the beginning or end of a selection structure. If programming efficiency is an issue (and a programming goal), then the statement **should** appear at the beginning of the structure. If it is **true**, the condition statement(s) is executed and the program can resume after the **else** statement, thus avoiding the testing of other conditions in the structure.

**TASK #4**

**Add an “error trap” to the program to check if the user entered an age less than zero (negative) or over 120. If the user enters an age in either of these ranges print out “Error. Invalid age entered.” instead of the ticket price. Make this change, check your program and show me your new program.**

**TASK #5**

**On Tuesdays, tickets are $6 for a child, $7 for senior and $9 for adult. Modify your program to also ask the user for the day (you can use initials, full word of the day or have them enter an integer 1-7 for the days of the week but your instructions should be clear in your user interface) Have your program check the day of the week AND the user’s age, and output the correct price accordingly. Include all previous modifications.**