1. **Control Flow**
   1. Statements break up execution conditionally flow with
      1. decision-making: based on readability and expression statement.
         1. if
         2. if-else
         3. if-else-if-else (leg else not required)
         4. switch: **switchblock**
            1. Single integer
            2. Enumerated value
            3. Can use a String object in statement’s expression.

Compared with expressions associated with each case label.

Similar to String.equals method

toLowerCase method, toUpperCase method

All strings associated with caselabels are in lowercase.

* + 1. looping
       1. for: executes *n* times.
          1. Initialization expression: initializes loop; executed once when loop begins.
          2. Termination expression: determines if loops should continue or end.
          3. Increment/decrement expression: invoked after each iteration through loop.

Beware of infinite loop.

* + - 1. Enhanced for: another form designed for iterating through Collections class or array, making loops more compact and easy to read.
         1. 2 parts

Data type of collection must match

Local variable name

Separated by a :

Collection name

* + - * 1. Example

int[] numbers = {1,2,3,4,5,6,7,8,9,10};

for (int item : numbers)

…;

* + - 1. while: continually executes a block while a Boolean expression is true; executes 0 to *n* times.
         1. Beware of infinite loop
      2. do while: executes 1 to *n* times
    1. branching statements
       1. break
          1. Unlabeled: commonly used in switch or to terminate loops to end nested loops.
          2. Labeled: terminate outer statement as well.
       2. continue
          1. Unlabeled: skip current iteration of innermost loop.
          2. Labeled: skip iteration of outer loop.
       3. return: exit from current method; return control flow to where method was invoked.
          1. returns a value
          2. does not return a value

1. **Administrative Flow**