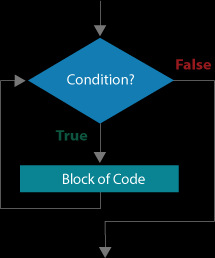
**05.00: LOOPS**

* There are three basic programming control structures:
  + **Sequential** (segments) – execute in order from start to finish
  + **Conditional** – alter the flow of control based on boolean expression
  + **Iterative** – alter the flow of control by repeating statements
* **Loop** – program statements that cause a segment of code to be repeated until a terminating condition is met
  + while, do-while, and for
  + do-while is not covered in the AP Computer Science exam

|  |  |  |
| --- | --- | --- |
| **Repetitive Activities** | | |
| **Activity** | **Condition** | **Action** |
| Counting backwards to 0 (counting loop) | **while** the number is greater than 0, | keep subtracting by 1. |
| Blowing out candles | **while** there are still candles burning, | keep blowing out any lit candles. |
| Sleeping | **while** the alarm has not gone off, | continue sleeping. |
| Doing 25 crunches (counting loop) | **while** the number of crunches is less than or equal to 25, | keep doing crunches. |
| Washing dishes | **while** there are still dirty dishes in the sink, | continue washing dishes. |

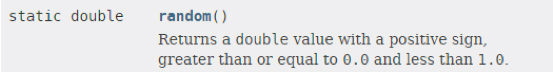
**05.01: WHILE LOOPS (PART 1)**



* Executes if condition is true
* Terminates when condition is false

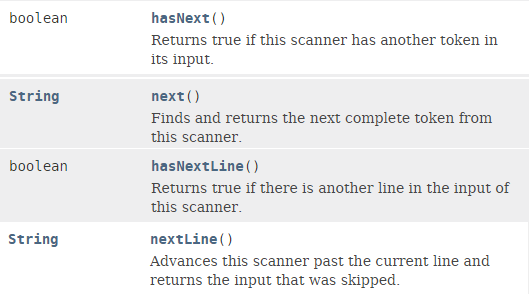
| **Negating an Expression** | | |
| --- | --- | --- |
|  | **negates to** |  |
| = |  | != |
| > |  | <= |
| >= |  | < |
| >= |  | < |
| < |  | >= |
| <= |  | > |

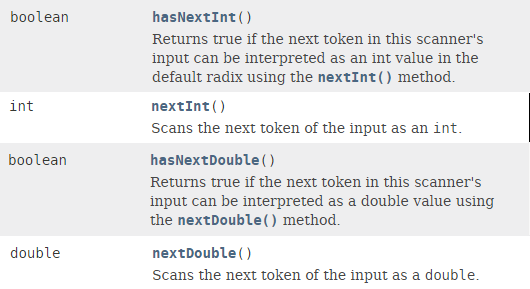
* To write a boolean condition for a while statement, decide what will be true when the loop is over, and then "negate" it.



**5.03: READING TEXT FILES**

* Java handles the content of a file as **tokens** in a stream of information
  + **Token** – single word, number, or symbol
  + “The several 5 beans”; 5; ‘a’





**5.04: FOR LOOPS**

* For loops are good for ***iteration***
* Because counting in ascending and descending order is such a common programming task, many languages define a loop specifically for this purpose: the **for** loop.
  + All it does is count
  + Boolean condition is always numeric

int counter = 0;   
  
while(counter <= 99)  
{   
   System.out.println(counter);  
   counter++;   
}

* + Bulky

for(int counter = 0; counter <= 99; counter++)  
{  
   System.out.println(counter);  
}

* + Concise
* If you find yourself writing the same block of code repeatedly to accomplish a task, it is probably an indication that a for loop is needed.
* Syntax:

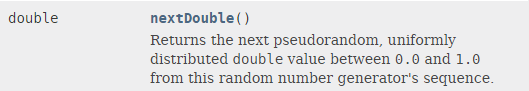
**for**(initialization; booleanExpression; update) {

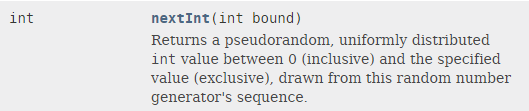
*//statements*

* + } *//end for loop*

**5.05: NESTED LOOPS**

* The Random class can pick random decimals and integers with the nextDouble() and nextInt() methods, respectively.





* The nextInt() method takes a parameter that indicates the maximum value of the integers in the list, minus one since the upper end of the range is not inclusive.
  + Consequently, nextInt(53) would return random numbers in the range of 0–52.
* The nextDouble() method returns decimals values from 0.0 inclusive to 1.0 exclusive.
  + Like the random() method of the Math class, decimal numbers returned will never equal 1.0.
* Random randNumList = new Random();
  + Constructs a new object of the Random class called randNumList.
  + This object holds the list of random numbers generated.

**05.06: WRITING TEXT FILES**

* The PrintWriter class contains methods with names similar to print() and println() and is the easiest way to write to text files with Java.
* Loops like “for” and “while” can be used to write tokens to a file
* Using print statements can help to debug programs that have to deal with writing to a text file so that live output can be observer simultaneously









