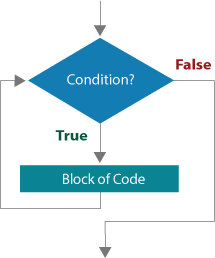
**Loopy Loop**



Like condition statements, loops alter the flow of control within a program; however, unlike **if** statements, iterative structures repeat a block of code. The characteristic repetitive structure of a **while** loop is shown in the accompanying flowchart.

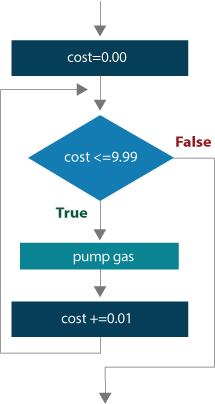
While the **boolean** condition is **true**, the block of code is executed, and then the program tests the condition again. But once the condition becomes **false**, the **while** loop exits. There is a potential problem with a loop: it may be endless!

To avoid an [**infinite loop**](javascript:void(0);), the state of the condition must eventually change from **true** to **false**. What causes a change in the **boolean** condition? Examining the list of activities and events you have begun recording gives a clue. How does counting backward to 0 or doing 25 crunches differ from blowing out candles, sleeping, or washing dishes?

The first two conditions terminate based on a counter reaching a specific terminal value (e.g., 0 and 25). The remaining conditions do not terminate based on a counter, but instead when some action, event, or process changes. For example, until the alarm goes off (e.g., an event change), you don't need to wake up.

So **while** loops continue to execute as long as a condition remains **true**. Be careful because soon you will begin recognizing loops everywhere as you develop a programmer's perspective. For example, while there is more information to learn in this lesson, keep reading!

Part 1



Imagine that you are driving around in your car and notice that it is time to fill up; however, you only have enough money for $10 worth of gas. (You can't get far with that these days!)

The computer operating the pump's shut off switch is broken. Consequently, you have to watch the meter very carefully to avoid going over the $10 amount.

The meter is initially set at \$0.00 and the cost increments by \$0.01 while you pump the gas.

The numbers whiz by, so you decide to stop when the cost reaches $9.99 in the display.

The accompanying flowchart segment illustrates this situation from a programmer's perspective.

Moving from the flowchart to code, the following program is an example of a **while** loop that stops based on comparing a counter to a terminal value (e.g., $9.99). Enter the program and observe the output.

public class GasPump  
{  
public static void main(String [] args)  
{  
double costOfGas = 0.0;  
  
while(costOfGas <=9.99)  
{  
costOfGas += 0.01;  
System.out.println("Cost: " + ((int)(costOfGas \* 100)) / 100.0);  
}  
}  
}

Writing a valid **boolean** condition for a **while** statement is sometimes tricky, but there is a very simple strategy that works every time. Decide what will be true when the loop is over, and then "negate" it. In the gas pump example, if the value pumped was **greater than** \$9.99, then you would run the risk of not stopping right on \$10.00, so to negate the condition, you decide to stop pumping when the amount is **less than or equal** to \$9.99.

Negating a simple **boolean** expression is easy using the following operator relationships. The operator in the left column negates to the operator in the right column, and vice versa. You might want to think of them as opposites.

Negation is important when you write the condition to terminate a **while** loop. For example, what do you want to be true after you finish pumping gas in the earlier example? Re-examine the conditions for each of the following activities and determine what would be true when each loop is finished.

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

| **Activities and Conditions** | |
| --- | --- |
| **Activity** | **Condition** |
| Counting backward to 0 | **while** the number is greater than 0, |
| Doing 25 crunches | **while** the number of crunches is less than or equal to 25, |
| Blowing out candles | **while** there are still candles burning, |
| Sleeping | **while** the alarm has not gone off, |
| Washing dishes | **while** there are still dirty dishes in the sink, |