Strings:

Strings are objects.
.equals()
.equalsIgnoreCase()
.compareTo() [returns a integer value] {String Object is less than, equal to, greater than String argument passed}

compares the value of two strings in lexicographic order.
String Object is less than String argument - Negative Integer
String Object equal to String argument - 0
String Object is greater than String argument - Positive Integer

Flow of Control, Looping:

In computing, we often perform tasks that follow this same pattern:

1. Initialize Values
2. Process Items one at a time
3. Report Results

The while loop is designed for repeating a set of instructions for each input value when we don't know at the beginning ho many input values there will be. We simply process each input value, one at a time, until a signal-an event-tells us that there is no more input. This is called Event Controlled Looping.

The condition is a while loop is a boolean expression. The scope of any variable defined within the while loop body extends from its declaration to the end of the while loop. Thus, any variable declared within a while loop body cannot be referenced after the while loop ends.

It is also possible to construct a while loop whose condition never evaluates to false. That results in an endless loop/infinite loop.The program doesn't terminate. it appears to hang. We need to abort the program.

The way to ensure that the condition will eventually evaluate to false is to include code, called loop update statement, within the body of the loop that changes the variable that is being tested by the loop condition.

ONE COMMON LOGICAL ERROR THAT CAUSES AN ENDLESS LOOP IS PUTTING SEMI-COLON AFTER THE CONDITION.
while(condition);
This indicates an empty loop body.

- Primary Read
- Update Read

Accumulation operation: Each time we input a new value, we add that value to the total. When we reach the end of input, the current value of total is the total for all the input.

```
set total to 0
read a number //priming read
while the number is not the sential value
{
add the number to total
read the next number //update read
}
ouput the total
```

```
for(initialization;loop condition; loop update)
{
//loop body  //loop control variable used for counting
}
```

while loops can be used for event driven or count controlled loops. a for loop is especially useful for count controlled loops. Each execution of loop body is an iteration of the loop.