A previous example, revisited

```java
System.out.print("Enter an interest rate: ");
rate = Double.parseDouble(br.readLine());
while (rate < 0 || rate > 100) {
    System.out.println("You entered an invalid rate. Please enter a rate between 0 and 100:");
    rate = Double.parseDouble(br.readLine());
}
...
```

What's the problem with the previous example?

- 2 lines are repeated:
  ```java
  System.out.print("Enter an interest rate: ");
  rate = Double.parseDouble(br.readLine());
  ```
- This is because we have a pre-test loop
  - need to test <condition> before the loop is executed
  - need to initialize <condition> before we can test it
- Q: is there a way we can prevent having to repeat lines of code?
  - thought: we need a post-test loop here
do-while loop

- "Do this while some condition is true"
- The test follows the loop
- The loop will execute at least once
  - a while loop might never execute, if the initial condition evaluates to false

Rewrite the previous example using do-while

```java
rate = -1.0;
do {
    System.out.println("Enter a rate between 0 and 100:");
    rate = Double.parseDouble(br.readLine());
} while (rate < 0 || rate > 100)
```

Another example

```java
int count = 1, sum = 0;
do {
    sum = sum + count;
    count++;
} while (count < 100);
```

Yet another example

```java
int num = 0, prod = 1;
do {
    System.out.println("Enter a number greater than 0:");
    num = Integer.parseInt(br.readLine());
    if (num < 0) {
        System.out.println("Thanks for playing. Bye!");
    } else if (num == 0) {
        System.out.println("Product is zero. Exiting.");
    } else {
        prod = prod * num;
        if (prod > 5000) {
            System.out.println("Product exceeds 5000. Exiting.");
        }
    }
} while (num > 0 && num != 0 && prod <= 5000);
```
Potential gotcha

- Notice that <condition> is a bit complex
  - we have multiple stop conditions
- Q: How can we clean this up?
- A: Use a flag to indicate when we should exit the loop

The previous example, revised

```java
boolean continue = true; int num = 0, prod = 1;
do {
    System.out.print("Enter a number greater than zero:");
    num = Integer.parseInt(br.readLine());
    if (num < 0) {
        System.out.print("Thanks for playing. Bye!";
        continue = false;
    } else if (num == 0) {
        System.out.print("Product is zero. Exiting.");
        continue = false;
    } else {
        prod = prod * num;
        if (prod > 5000) {
            System.out.print("Product exceeds 5000. Exiting.");
            continue = false;
        }
    }
} while (continue);
```

Half-loops

- Sometimes, we want to exit a loop in the middle:
  - a variable changes partway through that might negatively affect the outcome of the loop, or result in an error
  - we want to do all input handling within the loop, without pre-setting anything (and thus retyping some code)
- while and do-while test at the beginning and end of the loop, respectively
- Q: How can we do this?

Example (pseudocode)

- repeat:
  - prompt user for interest rate
  - if valid rate entered, exit loop
  - else, print an error message
Example (Java code)

```java
double rate;
while (true) {
    System.out.print("Enter interest rate: ");
    rate = double.parseDouble(br.readLine());
    if (rate >= 0 || rate <= 100)
        break;
    else
        System.out.println("You must enter a rate between 0 and 100.");
}
```

Notes

- `<condition>` is always true
  - indicates that we should always run this loop
  - necessary to have a `break` statement in this case
  - watch for the infinite loop!
  - alternately, we could have used another statement that is always true, like
    ```java
    while (1){ ... }
    ```
- Now, we don't need to repeat the statement that asks for input, and we can include an error message too!

Q: How could we improve this code?

```java
System.out.print("Enter a positive number (or any negative number to quit): ");
int num = Integer.parseInt(br.readLine());
int sum = 0, count = 0;
while (num >= 0) {
    count++;
    sum = sum + num;
    System.out.print("Enter a positive number (negative number to quit): ");
    num = Integer.parseInt(br.readLine());
}
double average = ((double)sum)/count;
System.out.println("The average of the numbers is "+average);
```