Repetition in Java

Do Loops
(check condition at end)

Recall: Simple Selection One Action

The simplest selection statement tests a single condition, and has a single action if the condition is true.

```
int age = In.getInt();

if (age >= 18)
{
   System.out.println("You can vote!");
}
```

Recall: Simple While Loop
(version 1 – does NOT work)

A while loop is like the if statement. It tests a condition, and if that condition is true, it performs the action. It also checks the condition before entering the loop.

```
int age = In.getInt();

while (age >= 18)
{
   System.out.println("You can vote!");
}
```

Simple Repetition: Do Loop
(version 1 – does NOT work)

A do loop is another type of loop, also designed to repeat actions while the condition is true. The difference is that it checks the condition at the end of the loop.

```
int age = In.getInt();

do
{
   System.out.println("You can vote!");
} while (age >= 18);
```

Do Loop vs While Loop
(check at end vs check at beginning)

```
int age = In.getInt();

do
{
   println("You can vote!");
} while (age >= 18);
```

```
int age = In.getInt();

while (age >= 18)
{
   println("You can vote!");
}
```

Consider age is less than 18 (e.g., 16)

do loop

- does not check condition at start
- will always run code inside loop at least once

while loop

- check condition at start
- false, so it never runs code inside loop

Simple Repetition: Do Loop
(version 1 – does NOT work)

This do loop has the same problem we encountered with a while loop – there is no way to end the loop if age is 18 or over. We end up with an infinite loop.

```
int age = In.getInt();

do
{
   System.out.println("You can vote!");
} while (age >= 18);
```

Simple Repetition: Do Loop
(version 1 – does NOT work)

Because it will always enter the loop at least once, we need to be more careful in how we use the do loop. This example, in fact, is not suitable for a do loop, and only serves to compare it to the while loop.

```
int age = In.getInt();

do
{
   System.out.println("You can vote!");
} while (age >= 18);
```

Simple Repetition: Do Loop
(ask for a positive number)

A common application of a do loop is ensuring that the user follows instructions for data entry. Suppose we only want positive numbers.

```
int number = -1;  // initialize to a bad value

do
{
   System.out.println("Enter a positive number");
   number = In.getInt();
} while (number < 0);
```

Simple Repetition: While Loop
(ask for a positive number)

This can also be done with a while loop. In fact, any looping task can be done with a while. It is the most general, although not always the most efficient.

```
System.out.println("Enter a positive number");
int number = In.getInt();

while (number < 0)
{
   System.out.println("Enter a positive number");
   number = In.getInt();
}
```

Do Loop - Summary

* always declare your variable(s) before the loop
* the code inside the loop will always execute at least once
* there must be a way to change variable(s) inside the loop (or you will never get out!)

```
set variable(s) to initial value(s)

do
    perform action(s)
    change variable(s)
loop while <condition 1>
```