

### Revision

- The while statement, which is used to build loops in Java, has a wide variety of usage.
- Since a while statement are just one Java statement, while statement can appear where, say, there is a while statement.
- A break statement can be used to stop a loop during the execution of the loop body.
- Loops are usually used to print tables, but printing tables usually requires tight control of output formatting.
- You can use the package provided by the book to do the formatting. You just need to use something like `Fmt.printf("%xx.yyc", something)` to print it instead of `System.out.print`.

0911B L12-1

### Implementing a for-each construct

Usually we want to do something for each number in a range. We can of course write it as a while statement.

For example, if we want to print out  $i^2$  for each  $i$  in the range from 1 to 20, we can do

```
int i = 1;
while (i <= 20) {
    System.out.println(i*i);
    ++i;
}
```

Problem: To understand the while statement, it requires to read the whole loop body and find out where  $i$  is changed.

When loop body is larger, it becomes very difficult to understand the program.

0911B L12-2

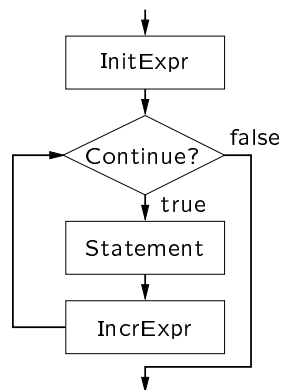
### The for statements

For statements moves everything to the beginning. It has the general form:

```
for (initExpr; condition; incrExpr)
    statement
```

which is very similar to

```
{
    initExpr;
    while (condition) {
        statement
        incrExpr;
    }
}
```



0911B L12-3

### Example

The multiplication table:

```
int i, j;
for (i=1; i<=10; i++)
    for (j=1; j<=10; j++) {
        Fmt.printf("%3d", i*j);
        if (j<10)
            System.out.print(" ");
        else
            System.out.println();
    }
```

This is completely equivalent to the program we have seen some time earlier, although this time we need not deal with invariants.

The other for loop says "for each  $i$  between 0 and 9", and the inner for loop says "for each  $j$  between 0 and 9". Much more direct.

0911B L12-4

### continue in for statements

- We say that the for statement is "very similar" to the while statement instead of they are equivalent, because `continue` has different effect in the two statements.
- In a while statement, a `continue` statement causes the program to go directly to check the condition.
- In a for statement, a `continue` statement causes the program to go executing the increment statement.
- Therefore, if the increment statement restore the loop invariant, it usually makes sense to use `continue`—unlike while.

0911B L12-5

### Example

Our example to print out the tan function values:

```
int i;
for (i = 10; i <= 180; i+=10) {
    if (i==90)
        continue;
    Fmt.printf("tan(%3d) = ", i);
    Fmt.printf("%8.2f\n", Math.tan(i*Math.PI/180));
}
```

Unlike our previous code, this do the natural thing: it starts with  $i=10$ , and the loop invariant says that  $i$  is the value to be printed with its tangent value.

When we `continue`, we still have one more chance to restore our invariant at the increment expression.

0911B L12-6

### Variations of for statement

- It might need more than one expression for initialization or for restoring invariants. In this case, you can write more than one expressions and separate them by commas.
- The init expression may be a variable declaration statement. In this case, the variable can be used only within the for loop.
- If *initExpr* or *incrExpr* is omitted, the part is simply skipped.
- If *condition* is omitted, this means *true*, i.e. the loop will only be stopped by a *break* somewhere inside the loop body. So you can write an infinite loop like "*for (;;) statement*".

0911B L12-7

### Example 1: Asking for yes/no

Ask for "Yes or No?" and read a character. If it is not either 'Y' or 'N', print an error message and repeat.

It is not convenient to check for the condition at the beginning, so let's use *break* to break out of the loop.

```
char ch;
for (;;) {
    System.out.print("Yes or No? ");
    ch = in.readChar();
    if (ch == 'Y' || ch == 'N')
        break;
    System.out.println("Not Y or N. Please try again.");
}
```

0911B L12-8

### Example: Printing a tree shape figure

```
*
***
*****
*****
*****
*
```

```
/* Print a tree */
import chapman.io.*;
public class PrintTree {
    public static void main(String[] args) {
        StdIn in = new StdIn();
        System.out.print("Tree width = ");
        int size = in.readInt();
    }
}
```

```
}
}
```

0911B L12-9