DIT948 Programming H16
Lecture 3

Instructor: Musard Balliu, `musard@chalmers.se`

September 3, 2016

QUESTIONS?

Today's plan

Basics of programming:

- Last lecture
  1. primitive types, variables, assignment, scope, blocks, strings
  2. Quick recap
- Today's Plan:
  1. conditionals (`if` and `switch` statements)
  2. loops: `for`, `while`, `do-while`

Conditionals

The general form of a conditional is:

```
if (booleanExpresion) {
    statement
}
```

The semantics of the conditional is: the statement is executed only if `booleanExpresion` evaluates to `true`.

Remember that a block is treated as a statement!

Conditionals

We can have an optional `clause`:

```
if (booleanExpresion) {
    statementIfTrue
}
else {
    statementIfFalse
}
```

The semantics is: if `booleanExpresion` evaluates to `true`, then `statementIfTrue` is executed and `statementIfFalse` is not. If `booleanExpresion` is `false`, then `statementIfFalse` is executed and `statementIfTrue` is not.

Exercises using `if` statements

1. Write a program to compute the minimum of two integers.

   Initial code

   Solution

2. Write a program to compute the minimum of three integers.

   Initial code

   Solution

3. Write a program to order three integers in ascending order.

   Initial code

   Solution

Chaining `if` statements

Exercise: An exam for 100 points can lead to one of five grades *A*, *B*, *C*, *D*, *E* according to the following scheme:

90–100	A
80–89	B
70–79	C
50–69	D
0–49	E

Write a program to output the correct grade given the number of points.

Initial code
Solution

Conditionals and boolean operators

Once we have conditionals, we can “emulate” the boolean operators.

Exercise: given booleans `b1` and `b2`, print the values of the expressions

`b1` *and* `b2`

`b1` *or* `b2`

not`b1`

without using `!, &&, ||`, but only `if-else` statements.

Initial code
Solution

The `switch` statement

The general form of a `switch` statement is:

```
switch (expression) {
case constant1:
      statement1
          break;
case constant2:
      statement2
          break;
      // (more cases)
default:  // optional default case
      defaultStatement
} // end of switch statemen
```

The semantics is: if the expression evaluates to one of the case constants, the corresponding statement is evaluated, **as are all statements until the first break or the end of the `switch` is reached**. Otherwise, if there exists a `default` case, then it is activated, if not, then nothing is done.

The `switch` statement

The expression in the `switch` statement must evaluate to one of the primitive types `byte, short, int, char` or to a `String`. It cannot, for example, be used for the grading program above.

Exercise: Write a short message of congratulations or encouragement, as the case may be, for a student who has received one of the grades above.

Initial code
Solution

The `for` loop

The general form of the `for` loop is somewhat complicated:

```
for (initialize; booleanExpression; expression) {
    statement
}
```

but usually it is used like this:

```
for(counter = start; counter  < end; counter++) {
    statement
}
```

If `counter` has not been declared, it can be declared where it is initialized, and is visible only in the `for` loop.

The semantics is: first, `initialize` is executed (for example, `count` is set to `start`). Then, the `booleanExpression` is evaluated. If `false`, the loop is over. If `true`, `statement` is executed and then the loop is executed again, **but with** `expression` **instead of** `initialize` (so, for example, `counter` is incremented instead of being set, again, to `start`).

Exercise using `for`

Write a program to add two integers using only incrementing and decrementing.

Initial code
Solution

The `while` loop

The general form of the `while` loop is:

```
while(booleanExpression) {
    statement
}
```

The semantics is: first, `booleanExpression` is evaluated. If `false`, the loop is finished. If `true`, then `statement` is executed, and the loop is executed again.

Exercise using `while`

Write a program to multiply two integers using only addition and subtraction.

Initial code

Solution

The `do-while` loop

The general form of the `do-while` loop is:

```
do {
  statement
} while(booleanExpression);
```

The semantics is: first, `statement` is executed. Then

```
while(booleanExpression) {
    statement
}
```

is executed.

`do-while` loop example

Write a program to count the number of times you need to roll a dice to get a six.

Initial code
Solution

Example: simple input

The line

```
import static dit948.SimpleIO.*;
```

brings into scope several expressions related to input-output.

We've already seen the output functions `print` and `println`.

For each primitive type `type`, there is also a corresponding expression `readType()` which, when evaluated, waits for input from the keyboard and then converts that input in a value of type `type` and returns that value as the final result.

The problem is that this conversion is not always possible (e.g., the user has not entered a number when a number was expected).

Example: simple input

To solve this problem, we need to use another element brought in scope by the `import` statement: the `boolean` variable `IOError`. The input functions in `SimpleIO.java` will set this variable to `true` if they cannot make sense of the input (or if another error has occurred).

Exercise: read an integer number from the user and double it.

Initial code
Solution

Conditionals and loops

Given the semantics of loops, we can easily see that any one of them can be used instead of the others, i.e. (any) one loop construct would have been enough.

Conditionals can also be translated to loops. This is perhaps easiest to see using a `while` loop.

Exercise: use a `while` loop to implement `Min2` above.

Initial code
Solution

Exercises

- Make sure you understand and run all the examples and exercises from this lecture.
- Read carefully sections 3.1 - 3.6, 3.8 - 3.10, 3.13 - 3.16 and Chapter 5 (except 5.9) of Liang's book (Version 10/E).

Preparation for Lecture 4: Read Chapter 6 and 7 of of Liang's book (Version 10/E).