

# Department of Computer Science

## COMP1100-99-14

Notes for Laboratory Session 08 (week 11)

Tutorial 08 (week 11)

### 1 Laboratory Session: Some Short Exercises

The thrust of this week's lab is the same as for Lab 7. It is to help you to develop your programming skills (to a level which is appropriate for going on to COMP1110).

Use the text and all the previous example programs for inspiration.

#### The simplest program

The starting version of the program should be one that reads a two-digit integer (one in the range 10..99) and says whether the digits in the decimal representation are in increasing order or not. The program should contain a routine that takes a 2-digit number represented as a string and computes the required (boolean) answer.

As an example of the output, we might see

`The digits of 84 are not in ascending order.`

Hints: You may find the method *to\_string* useful. Accessing characters in a string is done using the *item* method (as with arrays).

#### An easy modification

Make a change to the program so that the function takes an integer, instead of its string representation.

#### Testing the routine

Now change the program so that we count the number of 2-digit integers in the range 10 to 99 for which the digits are in increasing order. Also count the number that are in decreasing order. For the last task write another function

#### Testing the random number generator

If the random number generator was being used to generate numbers from 0 to 999, we would want there to be as many numbers between 100 and 899 with digits in ascending order as in descending order. Generate 500 random numbers and make this comparison.

#### The final program

This is the final program! Call it lab08.e and submit it to the marker system.

## 2 Tutorial

The discussion at the tutors meeting revealed that the topics from last week were still alive. It seems that most groups just covered one aspect and didn't get into the other. Here are those areas again.

The first: understanding the lecture material on syntax, which is important to being able to work with the Eiffel grammar, in both its forms.

The second recommended activity was to develop your ability to capture the essence of some real world class by a class in the Eiffel sense.

### Grammars

If you know some 2nd year students, you might well have heard them cursing Diana recently. As part of an assignment, they had to deal with formulae which had the following Syntax:

```
Diana_Formula
    variable | integer | function ( Diana_Formula , Diana_Formula )
variable
    letter
function
    letter
```

Start by suggesting some expressions which are instances and some which are similar but actually not legal. Then produce a rail diagram for Diana formulae. You could also do a rail diagram for Eiffel identifier, if there's time.

### Classes for Fun

Explore, perhaps in groups, what sort of classes are appropriate to represent some other classic games, such as checkers, noughts and crosses, boggle, or even monopoly.

The discussion should focus on getting at all of the attributes of each class suggested, including routines. Some of these routines might be simple enough that you could write code.