This question paper consists
of **??** printed pages, each of
which is identified by the
Code Number COMP5250M01

©UNIVERSITY OF LEEDS

School of Computing

January 2006

PRF: Programming Fundamentals

Time allowed: Two hours and 30 minutes

Answer BOTH questions.

Submit your programs by email as attachments to `prf@comp.leeds.ac.uk`. Confirm their arrival before you leave the examination room.

Candidates are reminded that Question 1 is worth 20 marks, and Question 2 is worth 40 marks. You are advised to use the time available in proportion to the marks available.

This is an Open Book examination. Candidates may bring with them whatsoever materials they please. Online resources may also be used. This explicitly includes dictionaries. Calculators may be used. Reproduction or simple rephrasing of notes or from texts or similar sources will not win any credit in any question.

Email and Instant Messaging (and similar) facilities may not be used.

Question 1

Complete the following tasks using Python.

(a) A program is required that converts a sequence of digits into words, as follows. Each digit is simply replaced with its word equivalent, so 123 becomes ”One Two Three” and 087 becomes ”Zero Eight Seven”.

A start has been made on this program. This can be found in `~prf/Exam/numbers.py`. Complete the program. For the moment it will be safe to assume that the input provided will contain only digits, but remember that this input may well start with zero. Solutions that do not make use of the code provided will not score full marks!

[5 marks]

(b) Write a *function* that determines whether or not a string (provided as a parameter) does indeed consist of only digits. Include your function in a program that demonstrates that it works.

[5 marks]

(c) Use your function from part (b) to extend your program from part (a) so that the program rejects any strings that are not made up solely of digits.

[5 marks]

(d) Finally, modify the program so that the number to be translated is provided as a *command-line argument*. Remember to make sure that your program behaves sensibly if the expected argument is not provided.

[5 marks]

[20 marks total]

Question 2

Take a copy of the file in `~prf/Exam/Person.java`. This is a simple class definition, intended to store basic details of a person.

(a) As it stands the class compiles, and would work. However, it contains two examples of bad programming practice. Identify these, and correct them. You might consider, for example, the use of access modifiers and the possibilities for invalid values being given to the attributes. Include a clear comment that shows what you have changed and why.

[8 marks]

(b) The class is currently devoid of comments. Add a clear comment to the start of the class and also to each method, outlining its purpose. Be sure to use appropriate `javadoc` tags.

[6 marks]

(c) Suppose that a program using this class contained the lines:

```
Person bill = new Person ("Bill", 14);
System.out.println (bill);
```

Add a new method to the class that would ensure that the output from this code was:

```
Bill is 14 years old.
```

Include the two lines given above in a main method so as to show that your new method works.

[6 marks]

(d) Bus Drivers are a special sort of person. Using the `Person` class implement a *derived* class called `BusDriver` that could be used to store details of Bus Drivers.

In addition to the usual attributes of a `Person` Bus Drivers have a “Home Garage” and a “Route Number”. The garage is one of the strings “Hunslet”, “Roseville Road”, “Bramley” or “Kirstall”. The route number is made up of at most three characters, which can include only “X”, “A”, “C” and digits.

Include proper `javadoc` comments in your class and pay proper attention to good style. Remember to extend the method you wrote for part (c) above.

Include a `main` method that shows clearly how all methods in your class work.

[20 marks]

[40 marks total]