CSC330 Summer 2004 Assignment 5

George Tzanetakis

July 26, 2004

1 Overview

This assignment will be done individually. The careful design and documentation (i.e comments) of your code will be important factors in your grade. If there is a bug it's better to report it than hide it. Be honest, precise and clear and you shall be rewarded. NEVER (at least in this class) sacrifice clarity for efficiency of execution unless explicitely asked to do so. PROVIDE A SET OF WELL-DESIGNED TEST CASES FOR EVERY PART OF THE ASSIGNMENT.

The assignment is worth 10 points (10% of the final grade) and will be graded in using half-point intervals. The overall documentation and packaging of your submission according to the instructions provided in the web page (READ THEM CAREFULLY) will be worth 1 point.

The main goal of the assignment is to familiarize you with C++, contrast object-oriented programming with functional programming and give you a taste of Prolog.

2 Part 1 (4 pt)

Implement the variable-length arithmetic part of assignment 2 in C++. Use the *vector* template from the standard template library. Each non-negative integer will be represented as an object using a vector of "digits" as internal representation. For a more complete description refer to assignment 2. Use operator overloading to implement the arithmetic operations.

Write 2 paragraphs as comments in your code contrasting the two approaches (worth 2pt). I realize that I haven't cover in detail C++ but you can find a lot of information and code examples on the web so part of the assignment is figuring the details out.

3 Part 2 Prolog (3pt)

An astronomer provides you with the following information about celestial objects in English: Sun, sirius and betelgeuse are *stars*. Mercury, venus, earth, and mars *orbit* the sun. Moon *orbits* the earth. Phobos and deimos *orbit* mars. A planet is a celestial object that orbits the sun. A satellite is a celestial object that orbits a planet. A celestial object is part of the solar system if it is the sun, or a planet or a satellite of a planet.

Encode these facts in Prolog building a system that can answer queries about celestial objects. Some examples that you will need to support are:

```
orbits(mars, sun) ?
(yes)
orbits(moon, sun) ?
(no)
orbits(phobos, B) ?
(mars)
orbits(B, mars) ?
(phobos and deimos)
planet(mars) ?
(yes)
planet(P) ?
(mercury, venus, earth)
satellite(phobos ) ?
(yes)
satellite(S) ?
(moon, phobos, deimos)
solar(sun) ?
(yes)
solar(sirius) ?
(no)
solar(B) ?
( sun, mercury, venus, earth, mars,moon, phobos, deimos)
```

To run prolog, type in your rules and facts in a file with extension .pl. Type gprolog and at the prompt you can load the facts/rules by typing ['file.pl']. Then you are ready to issue queries. Remember the distinction between variables (capital letters) and relations (lowercase letters).

4   Part 3 List stuff in Prolog (3pt)

As we showed in class, a definition of the relation member for lists in Prolog is the following:

```
mymember(X, [X|T]).
mymember(X, [Y|T]) :- mymember(X,T)
```

Define a relation *multiple* that is the property of being a list with multiple occurances of some element. (hint: use mymember). For example: multiple([1,2,3]) is false and multiple([1,2,2,3]) is true.

Define a relation *last* such that last(X,L) is true if X is the last element of list L. For example last(1,[1,2,3]) is false and last(3,[1,2,3]) is true.

Define a relation *drop* such that if drop(L1, N, L2) is true L2 is obtained by dropping every Nth element from L1. For example: drop([1,2,3,4,5,6], 2, [1,3,5]) is true.

(Hint (maybe more confusing than helpful): define a relation dropaux(L1, N, L2, K) where L2 is obtained from L1 by first copying K-1 elements and then dropping an element and, from then on, dropping every N'th element. Use dropaux to define drop. (hint to evaluate you have to use the is keyword: for examplenbsp K1 is K -1 if K is 5 then K1 will become 4)

5   Extra Credit

Warning EC this time is harder than usual for the less points.

- (2pt) Implement a full calculator for big numbers with variables in C++ using the code from the assignment (include subtraction and division). Infix notation should be used. For example something like this should be valid input:

```
a = 111111111111111111111111111
b = 99999999999999999999999
c = (a + b) * 22222222222222222
```

- (2pt) Implement any two of the Prolog assignments from the Ramsey-Kamin book.

6 Submission

Please follow the submission guidelines from the course webpage. In summary tar and gzip everything into one file. Include a README explaining how things are structured. Include all the code necessary to compile the assignment not just the parts you extended/modified.

HAVE FUN AND I HOPE YOU ENJOY THIS ASSIGNMENT AS MUCH AS I ENJOYED PREPARING IT