F28PL Coursework PL1 (Prolog questions). Deadline 30 Nov 2018

Important note: In all cases code should be clearly-written and should include a brief explanation in English explaining the design of your code.

Your answer must take the form of a prolog text file including the program and a nontrivial collection of tests (see template provided).

You may not use a library function if it renders the question trivial.

1. **Complex number arithmetic**

The **complex numbers** are explained here (and elsewhere):
http://www.mathsisfun.com/algebra/complex-number-multiply.html
Represent a complex integer as a two-element list of integers, so `[4,5]` represents 4+5i.

Write Prolog predicates

```
cadd/3
cmult/3
```

representing complex integer addition and multiplication. Thus for instance,

```
cadd([X1,X2],[Y1,Y2],[Z1,Z2])
```

succeeds if and only if `Z1=X1+Y1` and Z2=X2+Y2.

Note that complex number multiplication is not just like complex number addition. Check the link and read the definition.

2. **Sequence arithmetic**

An **integer sequence** is a list of integers. Write a Prolog predicate

```
seqadd/3
```

such that `seqadd(X,Y,Z)` succeeds when X and Y are lists of integers of the same length and Z is their sequence sum.

3. **Matrices (unmarked)**

Explain how you would implement matrix addition and multiplication, starting from the Prolog prompt.

(Hint: the answer starts with "^D, rlwrap python3")

4. **Essay-style question**

4a. Explain what **backtracking** has to do with Prolog. You might find this webpage helpful: https://www.doc.gold.ac.uk/~mas02gw/prolog_tutorial/prologpages/search.html

4b. Explain to what extent Prolog can be viewed as a logic programming language, and to what extent it cannot be so viewed. Include example code fragments as appropriate.

5. Cool question

Write a database for a predicate `cycleoflife/1` such that the query

```
cycleoflife(X)
```

returns the instantiations

```
X = eat
X = sleep
X = code
X = eat
X = sleep
X = code
...
```

in an endless cycle.

(This question has a beautiful and simple answer. If you find yourself writing lines and lines of complex code, there's probably something amiss.)