CSc 372, Fall 2006
Assignment 7
Due: Thursday, November 9 at 23:59:59 MST

Problem 1. (5 points) `bases.pl`

Write a predicate `bases/2` such that `bases(X, Y)` prints the integers from `X` through `Y` in decimal, hex, and binary. Assume that `X` is non-negative and that `Y` is greater than `X`. Examples:

```
?- [bases].
% bases compiled ...

Yes

?- bases(0, 5).
 Decimal      Hex         Binary
     0          0              0
     1          1              1
     2          2             10
     3          3             11
     4          4            100
     5          5            101

Yes

?- bases(1022,1027).
 Decimal      Hex         Binary
   1022       3FE     1111111110
   1023       3FF     1111111111
   1024       400    10000000000
   1025       401    10000000001
   1026       402    10000000010
   1027       403    10000000011

Yes
```

BE SURE that your predicate succeeds, showing `Yes`, not `No`.

Below is a predicate, `fmttest/0`, that shows almost exactly the specifications to use with `format/2`. However, you'll need to do `help(format/2)` and figure out how to output numbers in hex and binary.

```
?- listing(fmttest).

fmttest :-
        format('~tDecimal~t~10|~tHex~t~20|~tBinary~t~35|\n'),
        format('~t~d~6|~t~d~16|~t~d~30|\n', [10, 20, 30]).

Yes

?- fmttest.
 Decimal      Hex         Binary
    10         20             30

Yes
```

Problem 2. (5 points) **`perms.pl`**

Write a predicate `perms/0` that prints all permutations of foods taken three at a time. If less than three foods exist, no output is produced but `perms` always succeeds.

The file `fall06/a7/pfoods.pl` has a set of predicates `foods2/0`, `foods3`, `foods4`, `foods5` that dynamically create collections of two, three, four, or five `food` facts, respectively. Create a symbolic link to avoid having to type that long path more than once:

```
% ln -s /home/cs372/fall06/a7/pfoods.pl
% ls -l pfoods.pl
lrwxrwxrwx 1 whm dept 31 Nov  1 22:08 pfoods.pl ->
/home/cs372/fall06/a7/pfoods.pl
```

Use those `foodsN` predicates like this:

```
% pl
Welcome to SWI-Prolog (Multi-threaded, Version 5.6.20)
?- [pfoods].
% pfoods compiled 0.00 sec, 3,136 bytes
Yes

?- foods3.    % Creates three food facts.

Yes

?- food(X).   % Let's be sure it made three food facts.
X = apple ;
X = broccoli ;
X = carrot ;

No

?- foods2.    % Drop back to two food facts.
Yes

?- food(X).
X = apple ;
X = broccoli ;

No
```

Now we're ready to see `perms` in action. Let's create a set of three foods and show the permutations:

```
?- [perms].
% perms compiled 0.00 sec, 1,416 bytes

Yes
?- foods3.

Yes
?- perms.
apple broccoli carrot
apple carrot broccoli
broccoli apple carrot
broccoli carrot apple
```

```
carrot apple broccoli
carrot broccoli apple

Yes
```

If we drop back to two foods, there is no output:

```
?- foods2.

Yes
?- perms.

Yes
```

It would be better for `perms` to indicate that too few foods exist but that's hard with the little bit of Prolog that we've seen thus far. The behavior of `perms` is undefined if no food facts exist.

`fall06/a7/perms.txt` is a log of a session where I ran each of `foods[2345]` followed by `perms`. Your output should match it.

DO NOT, repeat, **DO NOT** consult a file with `food` facts. Use the `pfoods.pl` predicates instead.

Problem 3. (10 points) `rectangle.pl`

Write a predicate `rectangle/2` that prints a rectangle of asterisks based on a specification of width and height in English. Examples:

```
?- rectangle(five,three).

*****
*****
*****

Yes
?- rectangle(three,one).

***

Yes
```

Widths and heights from `one` through `ten` are recognized. The behavior is undefined if other atoms are used. Note that there is a blank line above and below the rectangle.

If a number is used instead of an atom, the user is reminded to use English:

```
?- rectangle(3, one).
Use English, please!

Yes
?- rectangle(one, 1).
Use English, please!

Yes
?- rectangle(1, 1).
Use English, please!
```

```
Yes
```

Hint: Start by writing a predicate `row(W)` that prints a row of `W` asterisks. Be sure it always succeeds!

Problem 4. (1 point each) `answers.txt`

Create a text file named `answers.txt` with answers to the following questions. **DO NOT submit a Word document, PDF, rich text file, etc.**—I want plain text.

(a) The `showfoods/0` predicate on slide 51 has two clauses. How does the behavior of `showfoods` change if the order of the clauses is reversed?

(b) In the four-port model, what action takes place when the user responds with a semicolon to a query result, like `A = 1`?

(c) On slide 32, what's wrong with the following line?

```
Query1 = food(F), Query2 = color(F,C),
```

Deliverables

The deliverables for this assignment are these files: `bases.pl`, `perms.pl`, `rectangle.pl`, and `answers.txt`. That list can be found on `lectura` in `/home/cs372/fall06/a7/delivs`.

Use the `turnin` tag `372_7` to submit your solutions.

Corrections and FAQs, late submissions, `turnin`, retests, etc.

Refer to the write-up for the first assignment for details on these topics and similar ones.

Miscellaneous

Along with this write-up and the resources mentioned in it, slides 1-56 have all the information you need to do this assignment. My solutions for `bases.pl` and `rectangle.pl` rely on `between/3`.

If time permits `fall06/a7/tester` will appear. If it does, use it! If not, eyeball it!

Solutions will be scored only on correctness.

Keep in mind the point value of each problem; don't invest an inordinate amount of time in a problem before you ask for help with it. Remember that the purpose of the assignments is to build understanding of the course material by applying it to solve problems. Seek the help of Poorna and me as needed to meet your time budget. Poorna's been focusing on Prolog for about the last month. Take advantage of his work!

Feel free to use comments to document your code as you wish but note that no comments, not even your name, are required.

I hate to have to mention it but keep in mind that I don't give cheaters a second chance to waste everybody's time. If you give your code to somebody else and they turn it in, you'll both likely fail the class, and more. (See the syllabus for the details.)