Homework 1

CMSC 471 – Fall 2012 – Kevin Winner

out 8/30/12 due 9/13/12

http://www.csee.umbc.edu/courses/undergraduate/471/fall12/hw1.html

1 What is AI? (20 pts)

Read "What is AI?" (http://www-formal.stanford.edu/jmc/whatisai.pdf) and answer the following questions. Provide thoughtful, legible and concise responses. This is not a quiz to see if you read the paper; these questions are asking you to give your own input about what you are thinking. Do not simply repeat what McCarthy wrote.

1. In what ways does AI, as described in this paper, differ from your preconceptions of AI? Where did you get these preconceptions (specifically)?
2. Why is intelligence often defined in the context of human intelligence? Do you think human-level AI will ever be achieved? Do you think it is possible? Why?
3. Which branches/applications of AI do you find the most interesting? Which do you suspect are the most active/most practical?

2 Reading Lisp (10 pts)

What do these functions do?

(a) 5 pts.
```
(defun enigma (x)
   (and (not (null x))
      (or (null (car x))
         (enigma (cdr x)))))
```

(b) 5 pts.
```
(defun mystery (x y)
   (if (null y)
      nil
      (if (eql (car y) x)
         0
         (let ((z (mystery x (cdr y))))
            (and z (+ z 1))))))
```

3 Writing simple functions (10 pts.)

(a) 5 pts. Write a function `(lesstwo n)` to return the number that is two less than its integer argument `n`. For example, `(lesstwo 2)` should return `0`; `(lesstwo 5)` should return `3`.

(b) 5 pts. Write a recursive function `(fact n)` to return the factorial of the argument `n`. (The factorial of an integer is the product of all integers from 1 to that integer.) For example, `(fact 3)` should return `6`; `(fact 10)` should return `3628800`.

4 Operating on lists (50 pts.)

(a) 5 pts. Write the function `(my-third l)` which returns the third element of the list `l`. Use only `car` and `cdr` (i.e., you may not use built-in functions like `third` or `caddr`).

(b) 10 pts. each There are often many different ways to solve the same problem in Lisp. In this problem, you will need to use your creativity and knowledge of Lisp functions to write the same function in several different ways. The function `(posint 1)` should take a list `l` and returns the list containing only the positive integers in the list. For example, `(posint '(a 2.3 -1 5 hello 3 (1 2)))` should return `(5 3 1 2)`. You can use the built-in function `integerp` in your solutions. All three implementations should be recursive.

1. Implement `posint1`, a version of the `posint` function using `mapcar`.
2. Implement `posint2`, a version of the `posint` function using the loop macro.
3. Implement `posint3`, a version of the `posint` function that operates recursively but *does not* use `mapcar` or `loop`.

(c) 15 pts. Write a function `(flatten-list l)` that takes an arbitrarily deeply nested list of atoms (possibly including dotted pairs), and returns a flattened list of these atoms (in the same order they appear in the original list). For example, `(flatten-tree '((((1) 2) ((3 . 4) 5)) 6))` should return `(1 2 3 4 5 6)`.

5 User input (10 pts.)

Write a program that takes a line of input from standard-in and echoes it to standard-out in the format "Hello [input from stdin]".