CSCI–4150 Introduction to Artificial Intelligence, Fall 2005
Assignment 1 (62 points), out Thursday September 1, due Thursday September 8

Notes on this assignment

- You should download the `assign1.scm` file from the assignment web page and write your code in this file. It will contain stubs for the procedures that you need to write.
- Your code is to be turned in electronically. You will find a link to the web tester from the assignment web page. See the web page for submission policy and details.
- You may always assume (on all assignments in this class) that your procedures will be given valid inputs.
- If you have any general questions on this assignment, please post to the WebCT discussion boards. The instructor and TA will be periodically monitoring the discussion groups.

Problems

0. (0 points) If you have a new CS account, you must change your password within the next week or so, otherwise your account will be disabled and you won't be able to turn in your code on time. You can log into a CS department machine in person or remotely log in to `freebsd.remote.cs.rpi.edu` (via `ssh`, SecureCRT, or similar). Run the `passwd` command from the prompt.

   If you ever need to get your password reset (e.g., because your account is disabled), contact `labstaff@cs.rpi.edu` or stop by Lally 301 with your ID.

1. (6 points) Write the procedure `(make-change amount)` where `amount` is an integer between 0 and 99, inclusive. It should calculate the minimum number of coins (quarters, dimes, nickels, and pennies) necessary to produce `amount` cents. Your answer should be returned in a list of the form: This procedure should return a list of nonnegative integers of the form: `(quarters dimes nickels pennies)`. For example:

   ```
   > (make-change 89)
   ;Value: (3 1 0 4)
   ```

2. (10 points) Section 6.3, Exercise 3 [`income-tax`]

   Note the restriction in this problem on not using any conditional statement. There will be a hint on the assignment web page for this problem, but try to figure it out yourself first.

3. (8 points) Write the procedure `(swap Lst j k)` that returns a list identical to `Lst` except that elements `j` and `k` are swapped. `j` and `k` are integers between 0 and `(- (length Lst) 1)` inclusive. For example:

   ```
   > (swap '(a b c d e f) 0 4)
   ;Value: (e b c d a f)
   ```

   You may use the MIT/GNU Scheme `sublist` procedure for this problem.

4. (6 points) Write the procedure `(element-number e Lst)` which returns the position of the first occurrence of `e` in the list `Lst` (starting from the front of the list). If `e` does not appear in `Lst`, it should return `-1`. For example:

```
> (element-number 'c '(a b c d b c a))
;Value: 2
> (element-number 'f '(a b c d b c a))
;Value: -1
```

5. (6 points) In preparation for plus/minus grading, I need you to write me a procedure to automatically assign plus/minus grades. Write the procedure `(plus/minus-ize grade)` that takes the `grade` argument (which must be one of the symbols `A`, `B`, `C`, `D`, and `F`) and returns a randomly generated plus- or minus-version of that grade. For example, given `B`, this procedure would randomly pick one of the symbols: `B+`, `B`, `B-`. Recall that there is no A+, D-, F+, or F-.

   Note: this is not an exercise in string manipulation or symbol creation; you will have to hardcode all the grade symbols. Instead, this is an exercise in using conditional expressions and selecting random elements from a list. Hint: you should use a `case` statement in this problem.

6. (6 points) Write the procedure `(meters->distance m)` where `m` is a distance in meters. It should convert this distance into a number of microns, millimeters, centimeters, meters, or kilometers. The unit should be chosen so that the corresponding number is the smallest number greater than or equal to 1, e.g., 1.5 centimeters instead of 15 millimeters and 90 centimeters instead of 0.9 meters.

   The result should be returned in a list where the first element is the number and the second element is either: `microns`, `millimeters`, `centimeters`, `meters`, or `kilometers`. For example:

```
> (meters->distance 0.0001)
;Value: (100. microns)
> (meters->distance 0.0035)
;Value: (3.5 millimeters)
> (meters->distance 1876)
;Value: (1.876 kilometers)
```

   Hint: you should use a `cond` statement for this problem.

7. (10 points) A series that converges reasonably quickly to π is given by:

$$\pi = \frac{3\sqrt{3}}{2} \sum_{i=0}^{\infty} \frac{(i!)^2}{(2i+1)!}$$

   Write a recursive procedure `(approx-pi N)` which sums the first (`N`+1) terms of this series (i.e., terms for $i = 0$ through $i =$`N`). For example:

```
> (approx-pi 0)
;Value: 2.598076211353316
```

8. (10 points) Write a (recursive) procedure `(compute expr)` that evaluates two-argument mathematical expressions of the form: `(op A B)`, where `op` is one of the symbols `add`, `subtract`, `multiply`, or `divide`, and `A` and `B` are either numbers or an expression of this form. For example:

```
> (compute '(add 8 2))
;Value: 10
> (compute '(multiply (add 2 3) (divide 100 (subtract 8.8 5.2))))
;Value: 138.88888888888889
```