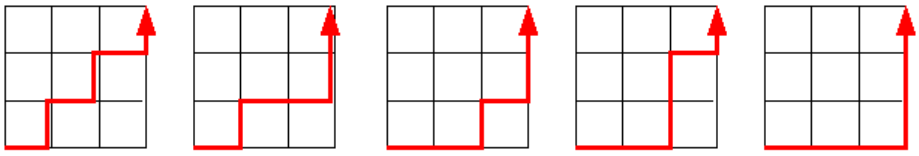
Mid-semester Examination

CS 152 – Abstractions and Paradigms in Programming

Date - 23nd Feb 2013 Max. Marks – 55

1. Write a function `(paths n)` which will find all paths from the leftmost lowermost corner (0,0) of a $n \times n$ grid to the rightmost uppermost corner (n,n) of the same grid. However, (a) you can only move up or right, and (b) you cannot go to a point (x, y) such that $y > x$. For example `(paths 3)` returns 5 as shown in the figure. (7 Marks)



2. One of the ways of estimating the area of a region r is to place it within a larger background region R whose area is known, generate a large number of random points within R, and count the number of points that also fall within r. Then we have:

$$\frac{\text{area of } r}{\text{area of } R} = \frac{\text{number of points that fall within } r}{\text{total number of points}}$$

As discussed in the class, we shall represent a region by a function taking a point as an argument and returning a boolean value as a result. And a point is a consed pair of two floating point numbers.

   (a) Write a function `(random-number n)` that will generate a random floating point number between `-n` and `n`. Refer to the function `random` in the documentation. (3 Marks)

   (b) Using this function, write a function `(random-point r)` to generate a random point within the region `r`. You can assume that if `(x,y)` is a point within the `r`, then both `x` and `y` lie in the range [`-N`, `N`], where `N` is a global variable. (2 Marks)

   (c) Now define a function `(calc-area r)` which will calculate the area of `r`. You can assume the same property of `r` as in the previous question. Your answer should be accurate upto a single decimal place. (5 Marks)

   You may test your function by calculating the area of a region formed by intersecting a circle of radius 3 around the origin and a region formed by translating a square of side 3 by $(-0.75, 1.25)$. The definitions of `N`, `square-maker`, `circle-maker`, `intersection` and `translate` will be supplied to you.

3. Suppose you had to find the value of:

$$\frac{a_1 * a_2 * \ldots a_n}{b_1 * b_2 \ldots b_m}$$

Suppose you were also told that

   (a) All the a's and the b's are unsigned numbers that can be stored in 8-bits.

(b) The final result is an unsigned number that can be stored in 8-bits.

(c) The product of any subset of the as, even two of them, could result in an overflow, i.e. an unsigned number not storable in 8-bits. Similarly for the bs.

Write a function `(evaluate la lb)` which will take two lists containg the as and the bs and calculate $(a_1 * a_2 * \ldots a_n)/(b_1 * b_2 \ldots b_m)$ without a overflow being produced. You should not use the operators `*` or `/`; instead use the functions `mult-8` and `div-8` that will be provided to you. You can also use the function `gcd` provided by Scheme. (8 Marks)

4. Write a function `(number-paren-exps n)` that calculates the number of ways in which an n-operand expression can be parenthesized. For example, `(number-paren-exps 4)` evaluates to 5 because there are 5 ways of parenthesizing a 4-operand expression, namely $a-(b-(c-d))$, $a-((b-c)-d)$, $(a-b)-(c-d)$, $(a-(b-c))-d$ and $((a-b)-c)-d$. (8 Marks)

5. Consider a binary tree represented as:

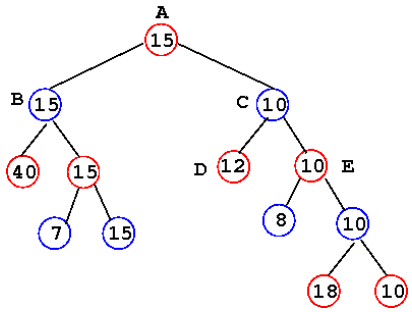
```
(struct gnode (val ltree rtree) #:transparent)
(struct leaf (val) #:transparent)
```

The nodes at odd levels have been colored red, and the nodes at even levels have been colored blue.

To start with, assume that the tree already has values at the leaf level. The values at the interior nodes are calculated as follows: If an interior node is red, then its value is the maximum of the values of its children. Else, if the interior node is blue, its value is the minimum of the values of its children.

(a) Define a function `(value tree)` which returns the value at the root given a tree with the values at the leaves. (2 Marks)

(b) Now let us think of a more efficient way to define `value`. Consider the tree shown in the figure below. Assume that the values at any level are computed from left to right and the values for the nodes labeled B and D have been already computed to be 15 and 12. Then, **without evaluating the tree rooted at E and C**, we can say that the value at the root is also 15. The reason is that the value at C being the minimum of D and E must be less than or equal to 12. And the value at A being the maximum of B and C must be 15.



Use this idea to define a more efficient version of the function `value` called `fast-value`. (10 Marks)