COM2001 MOCK ASSIGNMENT 2

JAMES MARSHALL
APRIL 13, 2015

Submission deadline: Part 1: 12:30 30th April 2015 (Part 2 will be self-marked)
Submission hand-in: To James Marshall, or at DCS Reception

1. Part 1

1.1. **Specification Completeness.** For the stack implementation from lecture 7, prove that the specification's axioms also imply that

```
isEmpty (pop (push x emptyStack)) == True
```

1.2. **Implementation Correctness.** For the set implementation from lecture 8, prove that the implementation satisfies the specification axiom

```
union emptySet s == s
```

1.3. **Subproblem Graph.** Draw a subproblem graph for the calculation of the first 8 numbers in a number sequence that is defined as

$$f(n) = \begin{cases} 3 & \text{if } n = 0 \\ 0 & \text{if } n = 1 \\ 2 & \text{if } n = 2 \\ f(n-2) + f(n-3) & \text{if } n > 2 \end{cases}$$

2. Part 2

2.1. **Number Sequence.** Implement a recursive function `number n` for calculating the n-th number in the number sequence defined in question 1.3.

N.B. If a negative number (n) is input the `number` function should return the message "`number cannot be negative`" (excluding quotes) using the `error` function defined in the Haskell Prelude.

2.2. **Number Triangle.** Implement a recursive function `node x y` for calculating the (x, y)-th number in a number triangle defined as

$$g(x, y) = \begin{cases} 0 & \text{if } |x| > y \\ 1 & \text{if } (x, y) = (0, 0) \\ \text{sum of all incoming paths} & \text{otherwise} \end{cases}$$

The sum of all incoming paths to a node is defined as the sum of the values of all possible paths from the root node $(x, y) = (0, 0)$ to the node under consideration, where at each node (x, y) a path can either continue diagonally down and left $(x - 1, y + 1)$, straight down $(x, y + 1)$, or diagonally down and right $(x + 1, y + 1)$. The value of a path to a node is defined as the sum of all the nodes along that path up to, but not including, the node under consideration.

The first few entries in the number triangle are given in table 1.

N.B. If a negative row (y) value is input the `node` function should return the message "`row index cannot be negative`" (excluding quotes) using the `error` function defined in the Haskell Prelude.

$y\backslash x$	-4	-3	-2	-1	0	1	2	3	4
0	0	0	0	0	1	0	0	0	0
1	0	0	0	1	1	1	0	0	0
2	0	0	2	4	6	4	2	0	0
3	0	4	16	40	48	40	16	4	0
4	8	72	352	728	944	728	352	72	8

TABLE 1. First Elements of Number Triangle

3. Self-Marking

For part 2 you can do preliminary testing of your solutions using automarking software developed at the University of Bristol. Sample tests are provided for you to check that your program meets the formatting requirements of the automarker. These materials can be downloaded from MOLE.

To run the provided tests on your program SSH into `stulogin`, change to the directory containing your file `dynprog.hs`, and type

```
java -jar mark.jar test
```

After a few seconds you will see a mark out of 4 on the command line for the number of tests your code passed, with more details in the newly-created file `feedback.txt`

At the end of the course an automarker script will be provided that will enable you to more fully test your solutions to part 2.

N.B. Ensure that both your functions from questions 2.1 and 2.2 can efficiently and accurately compute numbers that are a long way into/down the number sequence/triangle.