Homework 01

15-150 Fall 2024

Due: Mon 2nd Sept, 2024 at 8:00pm

LaTeX

The written part of your assignments must be written in LaTeX. To make your job easier, a LaTeX template is provided for each assignment where you simply need to fill in your solutions.

If you want to know what is the LaTeX name for a symbol, you can use Detexify. Also, Mathcha can help you type math. For constructs particular to this course (inductive definitions, proofs, etc), we provide a LaTeX guide with templates.

Finally, if you are just lost on how to type something or why your file is not compiling, reach out to the course staff.

Your submission will be penalized if it is not written using LaTeX. The applicable penalties are described in the style guide.

Code Structure

Check out the *basic requirements* for coding tasks in the style guide.

Proof Structure

Check out the *basic structure* for proofs in the style guide.

1 Interpreting error messages

In this part of the homework, you will be working with the file `errors.sml` that you downloaded as part of the startup code

You can evaluate the SML declarations in this file using the command

```
use "errors.sml";
```

at the SML prompt. The file contains errors for you to correct. The next seven tasks will guide you through the process of correcting these errors. Remember to fix only one error at a time and evaluate the `errors.sml` file after fixing each error.[1]

Task 1.1 (2 points) What error message do you see when you evaluate the `errors.sml` file without modifying it? What caused this error? How can it be fixed?

Task 1.2 (2 points) What is the next batch of errors? What caused these errors? How can they be fixed?

Task 1.3 (2 points) What error do you see after that? What caused it? How can it be fixed?

Task 1.4 (2 points) What is the next error? What caused it? How can it be fixed?

Task 1.5 (2 points) After this, you should see two errors. What are they? What caused them? How can they be fixed?

Task 1.6 (4 points) Once you have fixed these errors, you will get a bunch more errors. Two simple fixes are sufficient to make them go away. What are these errors? What caused them? What are the fixes?

When you correct this final error and evaluate the file there should be no more error messages. Congratulations! This is your first taste of debugging SML code.

[1] The semicolons at the end of the declarations in `errors.sml` are not necessary. We included them to make this task easier for you.

2 Specs and Functions

Consider the following function:

```
(* octal n ==> r *)
fun octal (n: int): int =
    if n < 8
      then n
      else (n mod 8) + 10 * octal (n div 8)
```

A specification for this function has the form

```
(* octal n ==> r
 * REQUIRES:  . . .
 * ENSURES: . . .
 *)
```

The function *satisfies* this spec if for all values `n` of type `int` that satisfy the assumption from the `REQUIRES` condition, `octal` evaluates to a value `r` that satisfies the `ENSURES` condition.

For each of the following specifications, say whether or not this function satisfies the specification. If not, give a counter-example to illustrate what goes wrong.

Task 2.1 (2 points)

```
(* octal n ==> r
 * REQUIRES: true
 * ENSURES:  r <> 0 (r is a non-zero number)
 *)
```

Task 2.2 (2 points)

```
(* octal n ==> r
 * REQUIRES: n > 0
 * ENSURES:  true
 *)
```

Task 2.3 (2 points)

```
(* octal n ==> r
 * REQUIRES: n >= 0
 * ENSURES:  r <> 0 (r is a non-zero number)
 *)
```

Task 2.4 (2 points)

```
(* octal n ==> r
 * REQUIRES: n >= 0
 * ENSURES:  r >= 0 (r is a non-negative number)
 *)
```

Task 2.5 (2 points)

```
(* octal n ==> r
 * REQUIRES: n >= 0
```

```
 * ENSURES:  r > 0 (r is a positive number)
 *)
```

Task 2.6 (2 points) Which *one* of these specifications gives the *most* information about the applicative behavior of the function `octal`? Say why, briefly.

3 Inductive Definitions

In this section, you will practice working with inductive definitions and proving properties by induction.

3.1 The Hosoya triangle

The Hosoya triangle is a geometric arrangement based on the Fibonacci numbers. Building the Hosoya triangle is fairly easy. This is shown in Figure 1. We start building the triangle from the top to the bottom. The top four values of the triangle are 1. Each other number is calculated as follows: if both numbers immediately above it along its *right* diagonal are defined, it is the sum of these numbers; otherwise it is the sum of the two numbers immediately above it along its *left* diagonal.

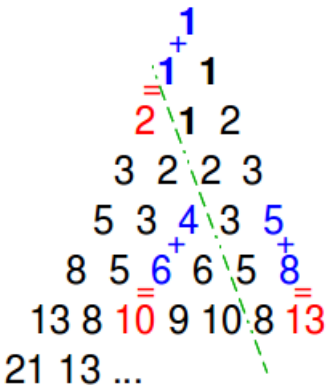

Figure 1: Building the Hosoya triangle

Each number in the Hosoya triangle is called a *Hosoya coefficient*, denoted $H(n, k)$. As shown in Figure 2, the Hosoya coefficient $H(4, 2)$ is 4 and it can be found at the intersection of the row $n = 4$ and the diagonal $k = 2$ in the Hosoya triangle. We start counting rows and diagonals at 0.

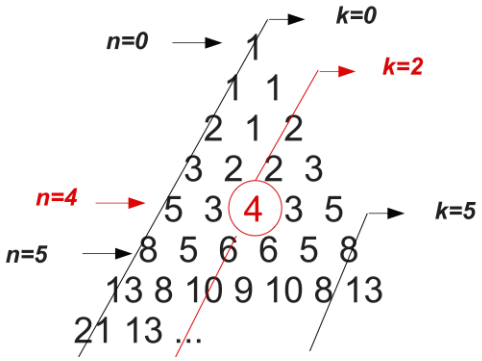

Figure 2: Navigating in the Hosoya triangle

Task 3.1 (8 points) Give an inductive definition of the *Hosoya coefficient* $H(n, k)$ **based on the way the Hosoya triangle is built**. Consider carefully the number of base and inductive cases, and the conditions under which each applies.

Both the leftmost and the rightmost edges in the Hosoya triangle look like the Fibonacci numbers. The variant of the Fibonacci numbers we use in this exercises is mathematically defined by the following

inductive definition:

$$\begin{cases} Fib(0) = 1 \\ Fib(1) = 1 \\ Fib(n) = Fib(n-1) + Fib(n-2) \quad \text{if } n > 1 \end{cases}$$

Task 3.2 (5 points) The mathematical property defining that the left edge of the Hosoya triangle corresponds to the Fibonacci number sequence is as follows:

Property 1. *For all natural number n,*

$$H(n, 0) = Fib(n)$$

Prove this property by induction on n.

Task 3.3 (10 points) By induction on n, prove the following property:

Property 2. *For all natural numbers n and k such that $0 \leq k \leq n$,*

$$H(n, k) = Fib(k) \times Fib(n - k)$$

Hint: your proof should match the structure of your inductive definition for H exactly: same cases, same conditions.

Hint: the proofs for the following two tasks are very short if you use the already proved properties above.

Task 3.4 (3 points) You may have noticed that each row of the Hosoya triangle has the same numbers whether read from left to right or from right to left (the Hosoya triangle has vertical symmetry). This is mathematically expressed by the following property:

Property 3. *For all natural numbers n and k such that $0 \leq k \leq n$,*

$$H(n, k) = H(n, n - k)$$

Prove this property.

Task 3.5 (3 points) Symmetry (or an easy observation) allows us to conclude that the rightmost edge of the Hosoya triangle is again the Fibonacci numbers. Formulate this fact as a mathematical property and prove it.

3.2 Coding it up

Now that we understand how the Hosoya triangle works, it is time to code it! Do not forget to add specs, tests, and follow the style guidelines.

Coding Task 3.6 (8 points) Implement your inductive definition for $H(n, k)$ as an SML function called `hosoya` with type `hosoya: int * int -> int`. For example,

```
hosoya(0,0) ↪ 1
hosoya(2,0) ↪ 2
hosoya(6,5) ↪ 8
```