National University of Singapore
School of Computing
CS1101S: Programming Methodology
Semester I, 2021/2022

S11-in-class

Problems:

Stream of pairs

1. Given a stream `s` the following function returns a stream of pairs of elements from `s`:

```
function stream_pairs(s) {
    return is_null(s)
        ? null
        : stream_append(
            stream_map(
                sn => pair(head(s), sn),
                stream_tail(s)),
            stream_pairs(stream_tail(s)));
}
```

   (a) Suppose that `ints` is the (finite) stream 1, 2, 3, 4, 5. What is `stream_pairs(ints)`?

   (b) Give the clearest explanation that you can of how `stream_pairs` works.

   (c) Suppose that `integers` is the infinite stream of positive integers. What is the result of evaluating

```
const s2 = stream_pairs(integers);
```

   Hint: Note that the function `stream_append` is defined in Source §3 as follows:

```
function stream_append(xs, ys) {
    return is_null(xs)
        ? ys
        : pair(head(xs),
               () => stream_append(stream_tail(xs),
                                   ys));
}
```

   (d) Consider the following variant of `stream_append`, called `stream_append_pickle` and the function `stream_pairs2` which makes use of it.

```
function stream_append_pickle(xs, ys) {
    return is_null(xs)
        ? ys()
        : pair(head(xs),
               () => stream_append_pickle(stream_tail(xs),
                                          ys));
}

function stream_pairs2(s) {
    return is_null(s)
        ? null
        : stream_append_pickle(
            stream_map(
              (x, sn) => pair(head(s), sn),
              stream_tail(s)),
            () => stream_pairs2(stream_tail(s)));
}

const s2 = stream_pairs2(integers);
```

Why does the function `stream_pair2` solve the problem that arose in the previous question?

(e) What are the first few elements of `stream_pairs2(integers)`? Can you suggest a modification of `stream_pairs2` that would be more appropriate in dealing with infinite streams?

Multiplying series

2. Multiplying two series is a lot like multiplying two multi-digit numbers, but starting with the left-most digit, instead of the right-most.

   For example:

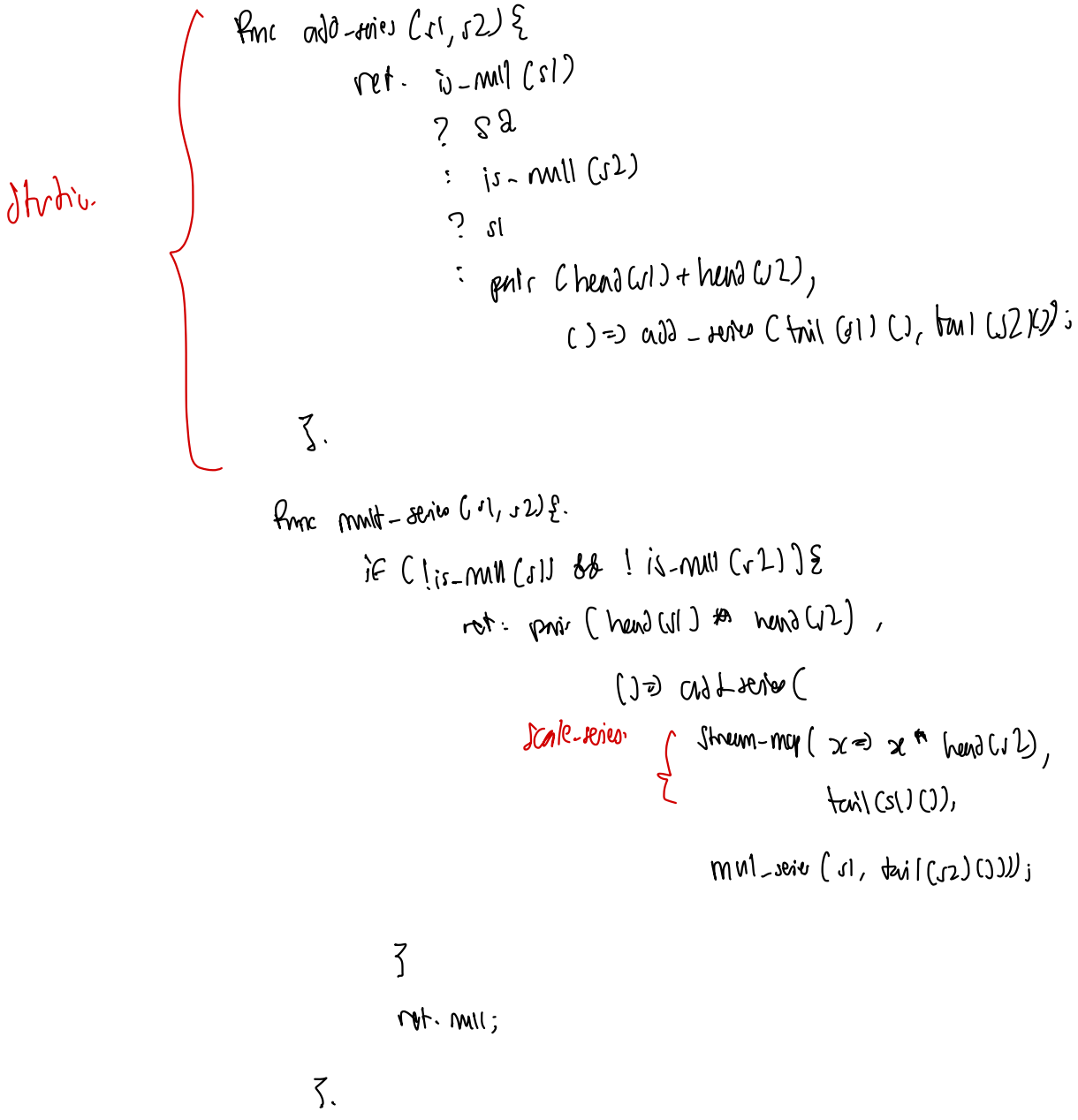
```
    11111
 x  12321
----------

11111
 22222
  33333
   22222
    11111
---------
136898631
```

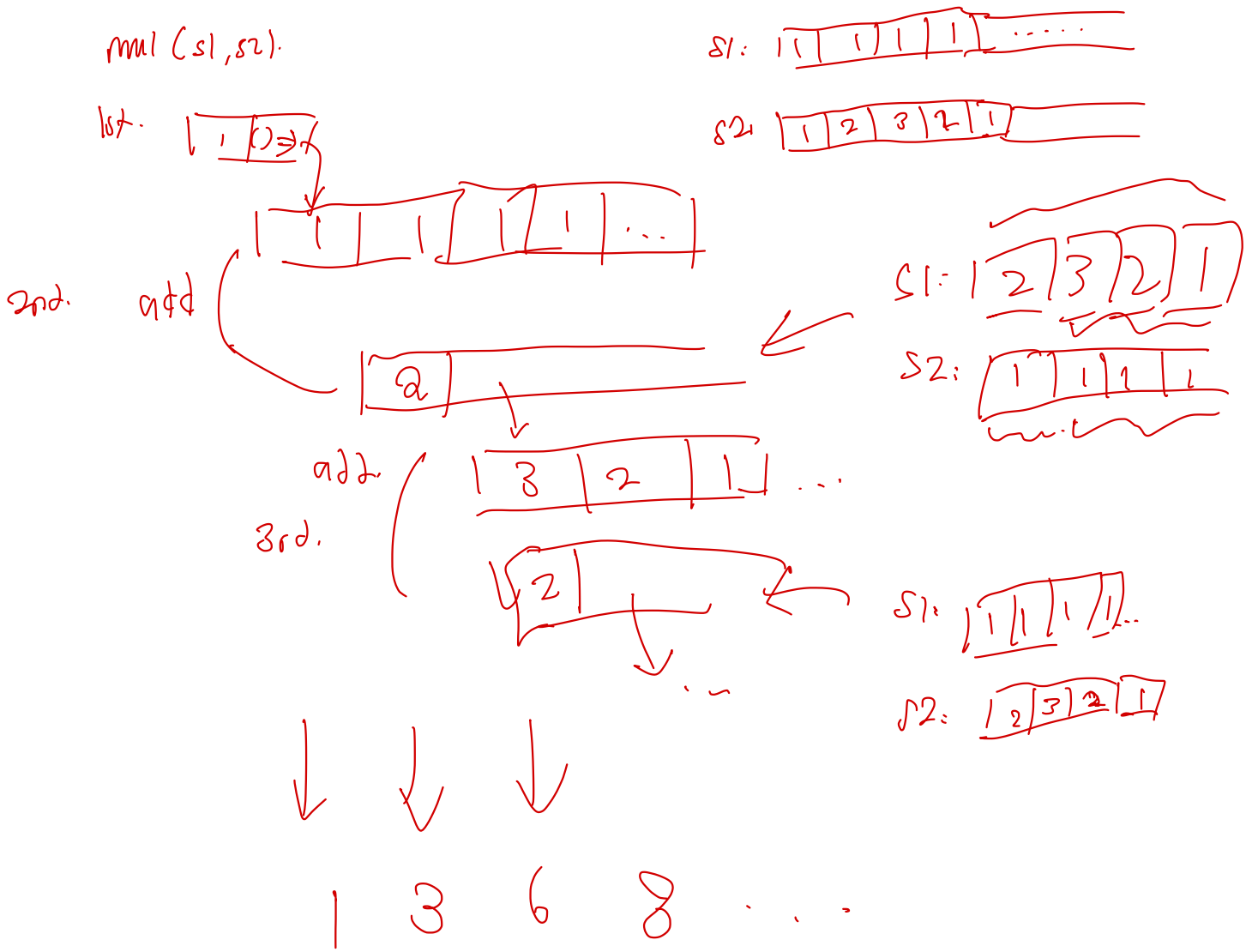
Now imagine that there can be an infinite number of digits, i.e., each of these is a (possibly infinite) series. (Remember that because each "digit" is in fact a term in the series, it can become arbitrarily large, without carrying, as in ordinary multiplication.)

Using this idea, complete the definition of the following function, which multiplies two series:

```
function mul_series(s1, s2) {
   return pair(<E1>,
       () => add_series(<E2>, <E3>));
}
```

To test your function, demonstrate that the product of S_1 (from Problem 3) and S_1 is S_2. What is the coefficient of x^{10} in the product of S_2 and S_2? Turn in your definition of `mul_series`. (Optional: Give a general formula for the coefficient of x^n in the product of S_2 and S_2.)




mul (s1, s2).
s1:
s2:
1st.
2nd. add
s1:
s2:
add
3rd.
s1:
s2:
1 3 6 8 . . .