U3SJEDR96 : I think you wnt to move that `)`: `List.map (\x -&gt; List.map (\y -&gt; Domino x y) &lt;| List.range x highest ) &lt;| List.range 0 highest`
U48AEBJQ3 : <https://ellie-app.com/3KQ5Rq7VdHNa1/0> ?
U2SR9DL7Q : That... worked? It says I've created a list of a list of dominoes. But If I can just flatten that, I should be fine.
U3SJEDR96 : `concatMap` to the rescue
U2SR9DL7Q : <@U48AEBJQ3> your solution is probably the more clever, FP way to do it, but I'll have to sit and study it.
U3SJEDR96 : or what <@U48AEBJQ3> did, which is nice :slightly_smiling_face:
U3SJEDR96 : the observation that for every element in the range, you only want to make combinations of the element and everything that follows is clever :slightly_smiling_face:
U2SR9DL7Q : Yes, but it's very imperative thinking. I've just made the elmy equivalent of ```
for i in range(0, num):
    for j in range(i, num):
```

U3SJEDR96 : yeah, and <@U48AEBJQ3> uses the same _idea_ in their implementation. I was actually remarking on his code, even though you'd done the same thing (but I hadn't realized it because I was trying to spot the bad code, rather than understand it)
U48AEBJQ3 : I guess the function-fu of `List.map &lt;&lt; (,)` is probably a bit much for learning. You can read it as:```
\x ys -&gt;
    List.map (\y -&gt; (x, y)) ys
```

U57KYFW67 : ```[1,2,3,4,5] |&gt; andThen (\x -&gt; [1,2,3,4,5] |&gt; andThen (\y -&gt; if x &lt; y then [(x, y)] else []))
[(1,2),(1,3),(1,4),(1,5),(2,3),(2,4),(2,5),(3,4),(3,5),(4,5)]
    : List (number, number)
```

U57KYFW67 : (how do I do code blocks in Slack??)
U48AEBJQ3 : triple backtick on its own line before and after the block
U57KYFW67 : tyty
U2SR9DL7Q : <@U48AEBJQ3> that makes it all much clearer. Unfortunately I never did enough haskell to get comfortable with all the inline functions. I'll remember this one now though.
U57KYFW67 : That code I posted does what the OP wanted. There's only two tricks to know: `andThen` allows you do iterate in a way a bit analogous to a for loop and the `if x &lt; y` condition will either append `(x,y)` or else it will append nothing.
U57KYFW67 : (to be realllly handwavy)
U2SR9DL7Q : <@U57KYFW67> you got it to work! that's what I tried initially, but the exact nature of what `andThen` is (binding operation for Lists that are monad typeclasses) makes me wary of using it too much.
U57KYFW67 : `andThen` is pretty neat, but the name doesn't make much sense.
U57KYFW67 : It will evaluate the lambda for each element in the list, then collects the results together.
U2SR9DL7Q : <@U48AEBJQ3> <@U3SJEDR96> <@U57KYFW67> This is quite a fun thing. Thank you for all the assistance.
U57KYFW67 : Here, it's a bit tricky because we do it twice in a nested way. But if you just think of the inner block, with `x` as a constant, it might make more sense: `[1,2,3,4,5] |&gt; andThen (\y -&gt; if x &lt; y then [(x, y)] else [])` goes through each of 1,2,3,4,5. At each iteration, the current value is called `y`. We compare `y` against the (constant) `x` and then decide if we want to append it or not.
U57KYFW67 : np np
U57KYFW67 : Haskell's do notation is actually really nice, and I miss it. It ends up looking like a sql query:  (this is more honest to what it looks like in Haskell)```
do
    x &lt;- [1,2,3,4,5]
    y &lt;- [1,2,3,4,5]
    guard (x &lt; y)
    return (x, y)
```

U2SR9DL7Q : guards... that's nostalgic.
U5W86UED6 : I have a question about "filtering" keyboard events <https://ellie-app.com/3KRNszj47Lca1/0>

U5W86UED6 : though I suppose it relates to all kinds of subscriptions
U5W86UED6 : essentially, I'm interested in only two keyboard events, but I'd rather not trigger the update function for any other keypress
U3SJEDR96 : That's not possible, at the moment, <@U5W86UED6> - it's a bit of an all-or-nothing situation
U5W86UED6 : <@U3SJEDR96> I realize the performance penalty isn't too bad, but it's more about filtering out information in the debugger
U5W86UED6 : do you have any more context about why it's an all-or-nothing situation? the design decision behind it?
U3SJEDR96 : <@U5W86UED6> - it's ok to have that conversation in the channel rather than in a thread :slightly_smiling_face:
U3SJEDR96 : as for the `why` - basically you'd need the ability to do something like `(a -&gt; Sub b) -&gt; Sub a -&gt; Sub b` so you could return `Sub.none` to filter them out, but that would have some non-obvious effects, such as the ability to suddenly "add" subscriptions from within a subscription