ROTEIRO APRESENTAÇÃO FIBONACCI EM HASKELL:

LAZY EVALUATION:
.Atrasa a avaliação da expressão até que o uso do seu valor seja necessário, e quando for, restringe a avaliação até a posição solicitada.
.Isso permite criar listas infinitas em um espaço finito, já que nenhum dos dados são computados.

.Vou mostrar um exemplo do conceito em python por ser uma linguagem que conhecemos melhor.

- Em python2, como pode ver, o valor não era atrasado, a função retornava toda a estrutura de valores.
- Já em python3 é retornado a função e quando solicitamos um valor é impresso aquela posição

TABELA AUXILIAR:
.Os dois próximos slides não sei se isso ocorre especificamente com haskell, mas é algo comum nas linguagens funcionais. Talvez tais tratamentos sejam feitos ao compilar Haskell com o parâmetro -O, onde ele otimiza o código.

.Quando f(2) é calculado pela primeira podemos armazenar seu valor para reutilizar nas próximas chamadas.

RECURSÃO EM CAUDA PARA LOOP:
.Uma recursão em cauda pode ser escrita na forma de laço de repetição, alguns compiladores cuidam dessa conversão.