1 Combinator backend

1.1 G-machine

- Gofer language, implementation of haskell.
- Originally compiler for ML-language.
- Uses super combinators.
- Close to Miranda.
- S & K, also others but can be expressed in S and K.

1.1.1 Super combinator

Either a constant or a combinator.

```
data NArtiyLambda a where
    End :: b -> a
    Step :: b -> NArtiyLambda a
data SuperCombinator a where
    Const :: a -> SuperCombinator a
    Combinator :: NArtiyLambda a -> SuperCombinator a
```

Any lambda calculus expr can be converted to SuperCombinator with *lambda lifting*. The technique is basically explicitly parameterizing it instead of closure.

2 Bytecode backend

2.0.1 ZINC

- OCaml backend.
- http://caml.inria.fr/pub/papers/xleroy-zinc.pdf
- LLVM official guide for ocaml language implementation, pure ocaml.
- Complete compiler.
- https://llvm.org/docs/tutorial/OCamlLangImpl1.html

3 Emit code

- Emit C/C++ code.
- Implementation easy.
- Dont worry about bytecode level performance.

4 Syntax

```
import std;
import util -> u;

let main =
    let a = 2;

    let b = a + 4;

    std.print a;
    import std.*;
    print b;
```