Dummy Slide For Poll

XD

some random day

Innermost & Outermost

An expression that can be reduced is called Redex
e.g. $(4 + 1) \rightarrow 5$

- **innermost**: Redex contains no other redex
- **outermost**: Redex is not contained within another redex

Reduction

- **primitive operation**
  can only be reduced if it is applied to fully evaluated arguments
  $1 + (2 + 3)$ as a whole cannot be reduced
- **functions**
  function's reducability depends on its definition
  $\mathit{fst}(1, 2) = 1$

Redexes

$1 + (2 * 3)$

innermost & outermost

Redexes

$1 + (2 * 3)$

neither (it self as a whole is not reducible)

Redexes

$(1 + 2) + (2 + 3)$

innermost & outermost

Redexes

$(1 + 2)$ * $(2 + 3)$

neither

Redexes

fst $(1 + 2, 2 + 3)$

innermost & outermost

Redexes

fst $(1 + 2, 2 + 3)$

innermost

Redexes

$\mathsf{fst}\ (1 + 2, 2 + 3)$

outermost

Redexes

fst (snd (1, 2 + 3), 4)

innermost

Redexes

fst (snd (1, 2 + 3), 4)

neither

Redexes

fst (snd (1, 2 + 3), 4)

outermost

Redexes

$$(\backslash x \rightarrow 1 + x)(2 * 3)$$

Redexes

$(\backslash x \rightarrow 1 + x)(2 * 3)$

innermost

Redexes

$(\backslash x \rightarrow 1 + x)(2 * 3)$

outermost

Redexes

$(\backslash x \rightarrow (1 + 2) + x)$

lambda expressions are different.
In general, one cannot reduce inside it
even if it's quite obvious, like 1+2 in this case

Why do we care about those?

Definition: loop = tail loop

```
Innermost reduction:
fst (1,loop) = fst(1,tail loop)
             = fst(1,tail(tail loop))
             = ...
```

```
Outermost reduction:
fst (1,loop) = 1
```

Outermost reduction terminates as often as possible

(see lecture slides p346)

exkurs: Weak Head Normal Form

https://wiki.haskell.org/Weak_head_normal_form
In order to evaluate an expression, it must be first evaluated to its WHNF.

exkurs: Weak Head Normal Form

consider the following expression:

```
inf :: Eq a => [a]
inf = inf

(==) :: Eq a => [a] -> [a] -> Bool
[] == [] = True
(x : xs) == (y : ys) = x == y && xs == ys
_ == _ = False
```

what happens if we call:

```
inf == inf
```

True? False? not terminating? throw exception?

exkurs: Weak Head Normal Form

'inf' will first be evaluated to its WHNF, but none of the case matches([] or (x:xs)), so the definition of inf is applied:
inf = inf

```
inf == inf
(inf <evaluated by definition> inf) == inf
inf == inf
... (not terminating)
```

and it won't terminate