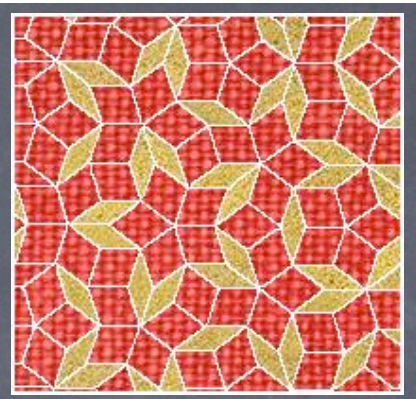
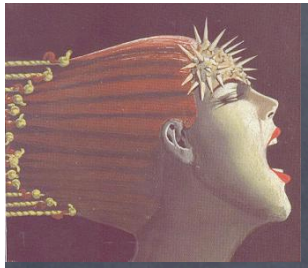


http://goldennumber.net/penrose.htm

Bildverarbeitung in Haskell

Ralf Lämmel
The man behind @lambdacat

Thanks to Dietrich Paulus for the invitation.

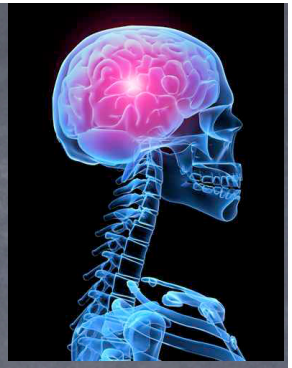


What's Haskell?

Haskell (pronounced ['hæskəl]) is a standardized, general-purpose purely functional programming language, with non-strict semantics and strong static typing. It is named after logician Haskell Curry.

http://en.wikipedia.org/wiki/Haskell_(programming_language)

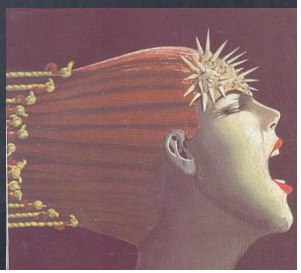
What's Haskell?

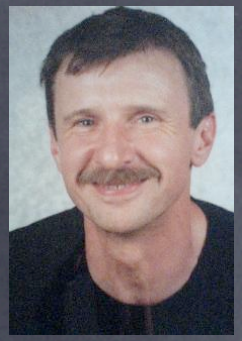


Haskell (pronounced ['hæskəl]) is a standardized, general-purpose purely functional programming language, with non-strict semantics and strong static typing. It is named after logician Haskell Curry.

http://en.wikipedia.org/wiki/Haskell_(programming_language)

It's a programming language!





: Is "Bildverarbeitung" possible in Haskell?

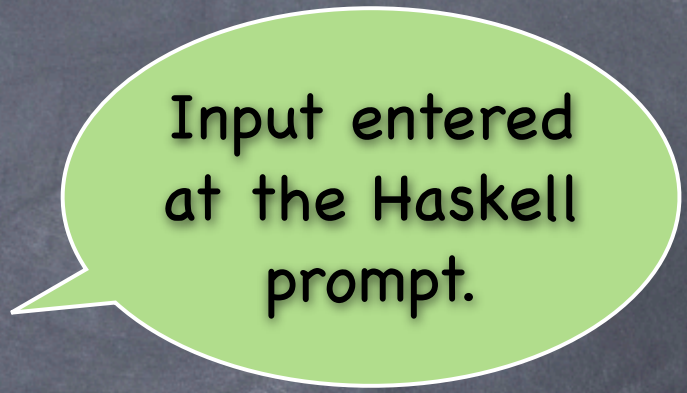
: Sure! Haskell is Turing complete.

But will we like the encoding?

How to do "Bildverarbeitung" in Haskell?

- Encoding 1:

```
> "Bildverarbeitung"

"Bildverarbeitung"
```



What's the type of "Bildverarbeitung"?

- Type 1:

```
> :t "Bildverarbeitung"
[Char]
```

Strong typing: do we care?

```
> :t "Bildverworking"
[Char]
> :t "C++"
[Char]
> :t "Java"
[Char]
```

Tons of subtle type errors are conceivable!

It appears that "Bildverarbeitung" is of the same type as "Java" which sounds pretty much like a contradiction.

"Bildverarbeitung" as first-class citizen?

- Encoding 2:

```
data Bildverarbeitung = Bildverarbeitung
```

```
> Bildverarbeitung

Bildverarbeitung
```

- Type 2:

```
> :t Bildverarbeitung

Bildverarbeitung
```

Theorem:

This is **not** a fixed point! (attributed to René Magritte)

Wow! This is strongly typed!

"Bildverarbeitung" & **Compositio**Phobia

- Encoding 3:

```
data A = A
data B = B
data C = C
...
> let bildverarbeitung =
(B,(I,(L,(D,(V,(E,(R,(A,(R,(B,(E,(I,(T,(U,(N,(G,())))))))))))))))
```

- Type 3:

```
> :t bildverarbeitung
bildverarbeitung ::
    (B,(I,(L,(D,(V,(E,(R,(A,(R,(B,(E,(I,(T,(U,(N,(G,())))))))))))))))
```

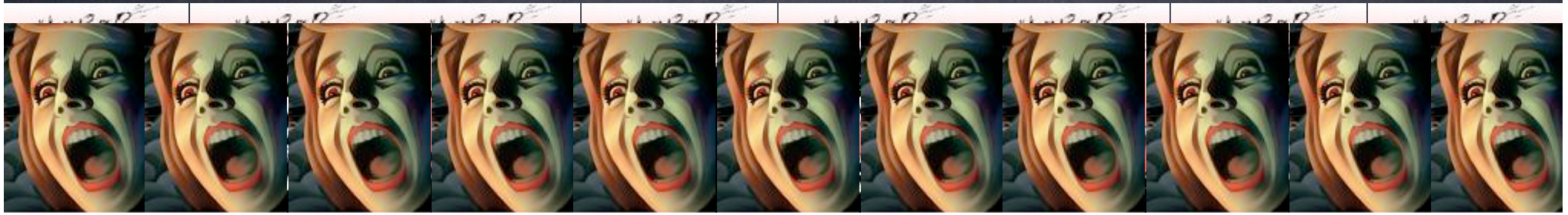
A domain-specific language for "Bildverarbeitung"

```
infixr 2 +

(+) x y = (x,y)

bildverarbeitung = B+I+L+D+V+E+R+A+R+B+E+I+T+U+N+G+()
```

Now we can compose "Bildverarbeitung" really easily!

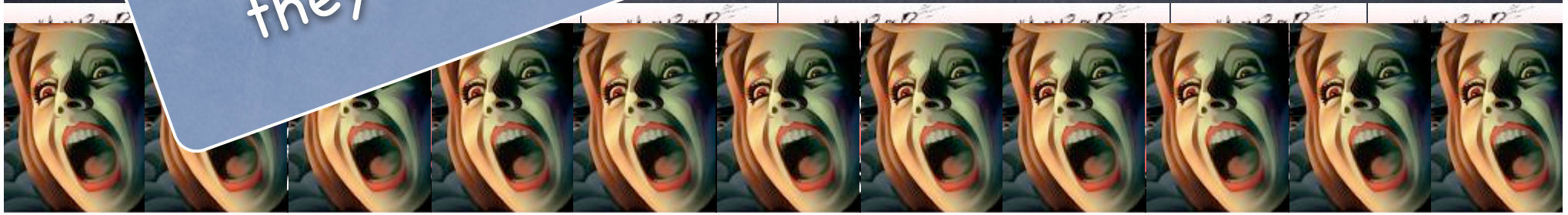
Sigh! This is again weakly typed!

B+L+Ö+D+V+E+R+T+Y+P+U+N+G+()



A domain-specific language for "Bildverarbeitung"

```
infixr 2 +

(+) x y = (x,y)

bildverarb
```



© Professor Fish: "Type errors typically arise from the lack of higher-order functions. Hence, they are atypical in typical Haskell code."

Haskell:
The *Weapon of Math Distraction*

- Wanted: higher-order functions
  - ¬ OK: **(B,(I,(L,(D,(V,(E,(R,...))))))))))**
  - OK: **b.i.l.d.v.e.r.a.r.b.e.i.t.u.n.g**
- Wanted: Type checking
  - OK: **b.i.l.d.v.e.r.a.r.b.e.i.t.u.n.g**
  - OK: **d.a.t.e.n.v.e.r.a.r.b.e.i.t.u.n.g**
  - ¬ OK: **d.a.t.a.v.e.r.w.o.r.k.u.n.g**

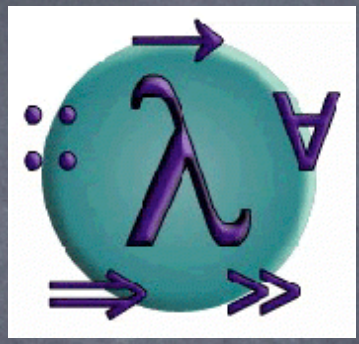
Demo of Encoding 4

```
> :t b.i.l.d.v.e.r.a.r.b.e.i.t.u.n.g $ ()

(B,(I,(L,(D,(V,(E,(R,(A,(R,(B,(E,(I,(T,(U,(N,(G,()))))))))))))))))

> :t b.e.l.d.v.e.r.a.r.b.u.i.t.i.n.g $ ()
```

... A few screens of type errors ...

Encoding 4: overloading the constituents of "Bildverarbeitung"

```
data A
data B
data C
...
```

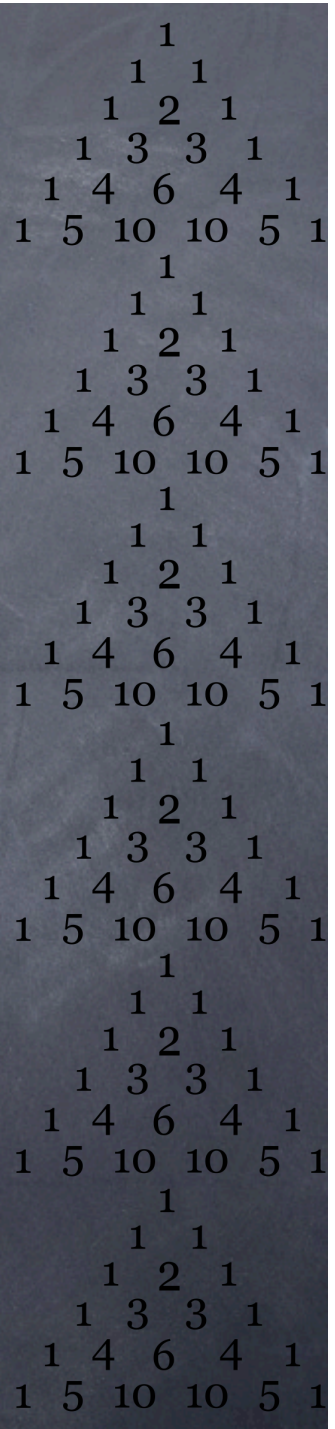


```
class A' x where a :: x -> (A,x); a = undefined
class B' x where b :: x -> (B,x); b = undefined
class C' x where c :: x -> (C,x); c = undefined
...
```

Doing "Bildverarbeitung" with types

```
instance G' ()
instance N' (G,())
instance U' (N,(G,()))
instance T' (U,(N,(G,())))
instance I' (T,(U,(N,(G,()))))
instance E' (I,(T,(U,(N,(G,())))))
instance B' (E,(I,(T,(U,(N,(G,()))))))
instance R' (B,(E,(I,(T,(U,(N,(G,())))))))
instance A' (R,(B,(E,(I,(T,(U,(N,(G,()))))))))
instance R' (A,(R,(B,(E,(I,(T,(U,(N,(G,())))))))))
instance E' (R,(A,(R,(B,(E,(I,(T,(U,(N,(G,()))))))))))
instance V' (E,(R,(A,(R,(B,(E,(I,(T,(U,(N,(G,())))))))))))
instance D' (V,(E,(R,(A,(R,(B,(E,(I,(T,(U,(N,(G,()))))))))))))
instance L' (D,(V,(E,(R,(A,(R,(B,(E,(I,(T,(U,(N,(G,())))))))))))))
instance I' (L,(D,(V,(E,(R,(A,(R,(B,(E,(I,(T,(U,(N,(G,()))))))))))))))
instance B' (I,(L,(D,(V,(E,(R,(A,(R,(B,(E,(I,(T,(U,(N,(G,()))))))))))))))))
```

Also known as Haskell's Triangle



Future work

- File patents on encodings
- Ready to ship
- Scheduled for SP1
  - Shorter error messages
  - Nice Eclipse support
  - Capitalization
    - B.i.l.d.v.e.r.a.r.b.e.i.t.u.n.g