Estruturas de Linguagem

Interpretação de Programas (com programação funcional)

Expressões

Francisco Sant'Anna

francisco@ime.uerj.br

http://github.com/fsantanna-uerj/EDL

Programa em C

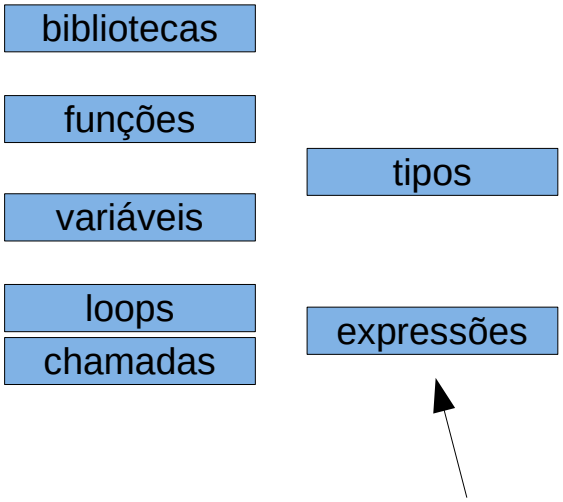
- Como executar?
- Como representar?
- Como interpretar?

```
#include <stdio.h>

int main()
{
   int c = 1;

   while (c <= 10) {
      printf("%d ", c);
      c++;
   }

   return 0;
}
```



Expressões

- Combinação de constantes, variáveis, operadores, etc, que pode ser avaliada (reduzida) a um valor.
  - `1 + 10`
  - `x * y`
  - `(x > 10) && (x < 100)`
- Expressões envolvendo constantes e operações aritméticas:
- Como representá-las usando Haskell?

Expressões Aritméticas

- Expressões envolvendo números e operações aritméticas
- Como representá-las em Haskell?
  - somente números inteiros, subtração, adição
  - sem variáveis
- Como avaliá-las em Haskell?
  - `avalia :: Exp -> Int`

```
data Exp = Num Int
         | Add Exp Exp
         | Sub Exp Exp
  deriving Show

e1 = Num 10
e2 = Add e1 e1
e3 = Sub (Num 100) e2

main = print e3
```

```
...

avalia :: Exp -> Int
avalia (Num v)     = v
avalia (Add e1 e2) = (avalia e1) + (avalia e2)
avalia (Sub e1 e2) = (avalia e1) - (avalia e2)

main = print (avalia e3)
```

Variáveis

```
data Exp = Num Int
         | Add Exp Exp
         | Sub Exp Exp
         | Var String
  deriving Show

-- 5 + i
e1 = Add (Num 5) (Var “i”)

main = print e1
```

```
consulta :: String -> Int
consulta id = <...> -- retorna o valor de ID

avalia :: Exp -> Int
avalia (Num v)     = v
avalia (Add e1 e2) = (avalia e1) + (avalia e2)
avalia (Sub e1 e2) = (avalia e1) - (avalia e2)
avalia (Var id)    = consulta id

main = print (avalia e1)
```