EIDI 2 Cheatsheet

github contributors: `https://github.com/zepatrik/eidi2_cheatsheet`

20. Februar 2018

1 Logik

$$\neg(A \vee B) \equiv \neg A \wedge B$$
$$A \vee (B \wedge A) \equiv A \wedge (B \vee A) \equiv A$$
$$A \implies B \equiv \neg A \vee B$$

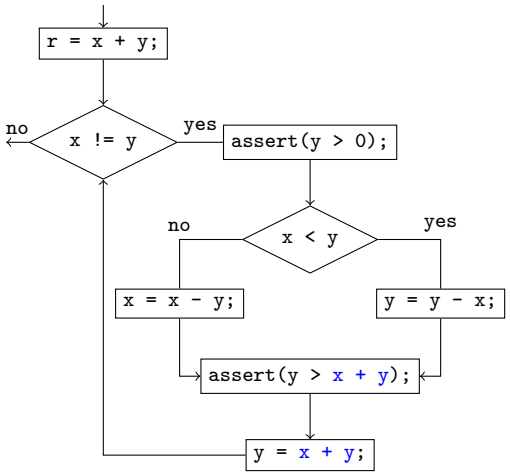
2 Verifikation

2.1 WP

$$\mathsf{WP}[\![\texttt{x = read();}]\!](B) \equiv \forall x.B$$
$$I \Rightarrow \mathsf{WP}[\![\texttt{b}]\!](B_0, B_1) \equiv I \Rightarrow (((\neg b) \Rightarrow B_0) \wedge (b \Rightarrow B_1))$$
$$\mathsf{WP}[\![\texttt{b}]\!](B_0, B_1) \equiv (b \vee B_0) \wedge (\neg b \vee B_1)$$
$$\equiv (\neg b \wedge B_0) \vee (b \wedge B_1) \vee (B_0 \wedge B_1)$$
$$\equiv (\neg b \wedge B_0) \vee (b \wedge B_1)$$

2.2 Terminierung

1. vor jedem Schleifendurchlauf `r > 0`
2. `r` wird bei jedem Durchlauf kleiner



3 ocaml

3.1 Funktoren

```
module type A = sig
  type t
  val f : t -> t
end

module B: A = struct
  type t = int
  let f x = x + 1
end

module Ext(X: A) = struct
  include X
  let g x = f (f x)
end

module C = Ext (B)
```

3.2 Threaded tree Beispiel

```
open Thread open Event

let rec min = function
  | Leaf a -> a
  | Node (a,b) ->
    let c = new_channel () in
    let f t = sync (send c (min t)) in
    let _ = create f a in
    let _ = create f b in
    let x = sync (receive c) in
    let y = sync (receive c) in
    if x < y then x else y
```

3.3 Exceptions

```
type t = exn

try (
  raise Failure "this should fail"
) with Failure s -> print_string s

exeption Custom of int
try (
  raise Custom 0
) with _ -> ()
```