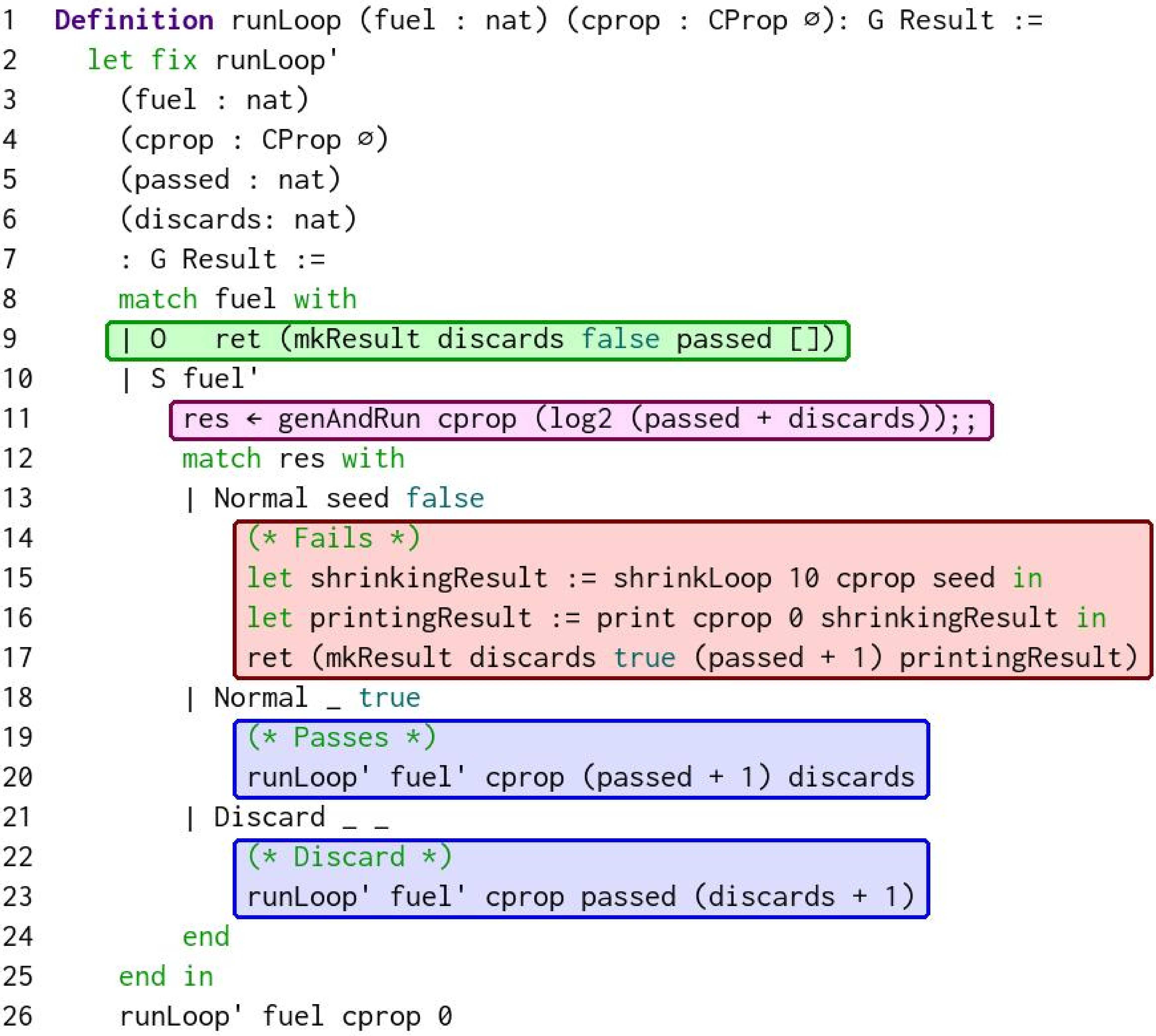

```
Definition runLoop (fuel : nat) (cprop : CProp ∅): G Result :=
  let fix runLoop'
    (fuel : nat)
    (cprop : CProp ∅)
    (passed : nat)
    (discards: nat)
    : G Result :=
    match fuel with
    | O   ret (mkResult discards false passed [])
    | S fuel'
        res ← genAndRun cprop (log2 (passed + discards));;
        match res with
        | Normal seed false
            (* Fails *)
            let shrinkingResult := shrinkLoop 10 cprop seed in
            let printingResult := print cprop 0 shrinkingResult in
            ret (mkResult discards true (passed + 1) printingResult)
        | Normal _ true
            (* Passes *)
            runLoop' fuel' cprop (passed + 1) discards
        | Discard _ _
            (* Discard *)
            runLoop' fuel' cprop passed (discards + 1)
        end
    end in
    runLoop' fuel cprop 0
```