;;; Constants
;;
(define GINNY
  )

(define ROMILDA
  )

(define VINCENT
  )

(define PANSY
  )

(define DRACO
  )

(define CHO
  )
;;; Data Definitions
;;;

;;; A RumorMill is either
;;; - '()
;;; - (make-gossip Image RumorMill RumorMill)
(define-struct gossip [who next1 next2])
#
;;; RumorMill -> ...
(define (fun-for-rumor-mill rm)
  (cond [(empty? rm) ...]
        [(gossip? rm)
         (... (gossip-who rm)
               (fun-for-rumor-mill (gossip-next1 rm))
               (fun-for-rumor-mill (gossip-next2 rm))))])

;;; An example rumor mill for Ginny to hear about Harry's supposed hypogriff
;;; tattoo. This could be defined using the images directly, as in the
;;; slides
;;; and the starter file. We switch to using constants to make it easier to
type.

(define GINNY-MILL
  (make-gossip GINNY '() '()))

(define ROMILDA-MILL
  (make-gossip ROMILDA '() GINNY-MILL))
(define VINCENT-MILL
  (make-gossip VINCENT '() '()))

(define DRACO-MILL
  (make-gossip DRACO ROMILDA-MILL VINCENT-MILL))

(define CHO-MILL
  (make-gossip CHO '() '()))

(define PANSY-MILL
  (make-gossip PANSY CHO-MILL DRACO-MILL))

;; Functions

;; RumorMill Image -> Boolean
;; Determine whether `who` is informed about a rumor that starts in `rm`.
(define (informed? rm who)
  (cond [[(empty? rm) #false]
         [(gossip? rm)
          (or (image=? who (gossip-who rm))
              (informed? (gossip-next1 rm) who)
              (informed? (gossip-next2 rm) who))]]))

;; Alternative version:
(define (informed? rm who)
  (and (not (empty? rm))
       (or (image=? who (gossip-who rm))
           (informed? (gossip-next1 rm) who)
           (informed? (gossip-next2 rm) who))))

(check-expect (informed? '() GINNY) #false)
(check-expect (informed? GINNY-MILL PANSY) #false)
(check-expect (informed? GINNY-MILL GINNY) #true)
(check-expect (informed? PANSY-MILL GINNY) #true)
(check-expect (informed? PANSY-MILL DOBBY) #false)
;; No one tells Dobby anything.

;; RumorMill -> NonNegativeNumber
;; Determine the maximum number of days for a rumor to propagate at a rate of
;; 1 day to pass it on


(define (rumor-delay rm)
  (cond [(empty? rm) 0]
        [(gossip? rm)
         (+ 1 (max (rumor-delay (gossip-next1 rm))
                   (rumor-delay (gossip-next2 rm))))]))

(check-expect (rumor-delay '()) 0)
(check-expect (rumor-delay CHO-MILL) 1)
(check-expect (rumor-delay ROMILDA-MILL) 2)
(check-expect (rumor-delay DRACO-MILL) 3)
(check-expect (rumor-delay PANSY-MILL) 4)

;; RumorMill Image Image -> RumorMill
;; Adds newper to rm, being told by oldper
(define (add-gossip rm newper oldper)
  (cond [(empty? rm)
         '()]
        [(image=? oldper (gossip-who rm))
         '()]
        [(else)
         (cond [(empty? (gossip-next1 rm))
                (make-gossip (gossip-who rm)
                             (add-gossip (gossip-next1 rm) newper oldper)
                             (add-gossip (gossip-next2 rm) newper oldper))]
               [(else)
                (make-gossip (gossip-who rm)
                             (gossip-next1 rm)
                             (add-gossip newper '() '()))]
               )]
        )]
)

(check-expect (add-gossip '() DOBBY GINNY) '())
(check-expect (add-gossip GINNY-MILL DOBBY GINNY)
              (make-gossip GINNY
(make-gossip DOBBY '() '(())
(check-expect (add-gossip (make-gossip PANSY
  GINNY-MILL
  '()))
  DOBBY
  PANSY)
(make-gossip PANSY
  GINNY-MILL
  (make-gossip DOBBY '() '(()))
(check-expect (add-gossip ROMILDA-MILL
  DOBBY
  ROMILDA)
(make-gossip ROMILDA
  (make-gossip DOBBY '() '())
  GINNY-MILL))