##===========================================================================

## TOXOCARA MODEL [8]

## last update: 21/01/2013

##===========================================================================

##== LIFE FUNCTIONS =========================================================

create <-

function(number){

 individuals <- data.frame(

 age = rep(0, number), # age of inds in months

 ageCat = rep(0, number), # age category

 ab = rep(FALSE, number), # antibody status

 monthsExp = rep(0, number), # months post exposure

 expStatus = rep(0, number) # number of exposures

 )

 return(individuals)

}

subsetAlive <-

function(x){

 x <- subset(x, x$ageCat != 3)

 return(x)

}

births <-

function(x, nBirths){

 x <- rbind(x, create(nBirths))

 return(x)

}

ageing <- # + categorize: [0,9), [9,60) & [60,180) & [180,Inf)

function(x){

 x$age <- x$age + 1

 x$ageCat <- cut(x$age,

 c(0, 9, 5\*12, 15\*12, Inf),

 labels = c(0:3),

 right = FALSE)

 return(x)

}

##== DISEASE FUNCTIONS ======================================================

seroConversion <- # no seroconversion for [0,9)

function(x, envcP, youngP, oldP, monthWiseP){

 youngInds <- x$ageCat == 1 # select individuals [9,60)

 oldInds <- x$ageCat == 2 # select individuals [60,180)

 seroConvYng <- rbinom(sum(youngInds), 1, youngP \* monthWiseP \* envcP)

 seroConvOld <- rbinom(sum(oldInds), 1, oldP \* monthWiseP \* envcP)

 x$ab[youngInds] <- x$ab[youngInds] + seroConvYng

 x$ab[oldInds] <- x$ab[oldInds] + seroConvOld

 x$ab <- x$ab > 0 # re-convert to logical: pos OR neg

 x$expStatus[youngInds] <- x$expStatus[youngInds] + seroConvYng

 x$expStatus[oldInds] <- x$expStatus[oldInds] + seroConvOld

 return(x)

}

seroReversion <-

function(x, startReversion, endReversion){

 x$monthsExp[x$ab] <- x$monthsExp[x$ab] + 1

 test <- runif(nrow(x)) < .5

 x$ab[x$expStatus == 1 &

 x$monthsExp >= rpois(1, startReversion[1]) & test] <- FALSE

 x$ab[x$expStatus == 2 &

 x$monthsExp >= rpois(1, startReversion[2]) & test] <- FALSE

 x$ab[x$expStatus >= 3 &

 x$monthsExp >= rpois(1, startReversion[3]) & test] <- FALSE

 x$ab[x$expStatus == 1 &

 x$monthsExp >= startReversion[1] + endReversion] <- FALSE

 x$ab[x$expStatus == 2 &

 x$monthsExp >= startReversion[2] + endReversion] <- FALSE

 x$ab[x$expStatus >= 3 &

 x$monthsExp >= startReversion[3] + endReversion] <- FALSE

 x$monthsExp[!x$ab] <- 0

 return(x)

}

##== REPORTER FUNCTION ======================================================

census <-

function(x, y, i){

 y$n1[i] <- sum(x$ab[x$ageCat == 1])

 y$n2[i] <- sum(x$ab[x$ageCat == 2])

 y$n[i] <- y$n1[i] + y$n2[i]

 y$pop[i] <- nrow(x)

 y$prev[i] <- y$n[i] / y$pop[i]

 y$prev1[i] <- y$n1[i] / sum(x$ageCat == 1)

 y$prev2[i] <- y$n2[i] / sum(x$ageCat == 2)

 return(y)

}

##== RUN LIFE CYCLES ========================================================

model <-

function(nStart, nCycles,

 envcP, youngP, oldP, monthWiseP,

 startReversion, endReversion){

 ENVC <- rbeta(nCycles, envcP[1], envcP[2]) # random envcP

 MONTH <- matrix(ncol = 12, nrow = nCycles/12) # random monthWiseP

 for (i in 1:12)

 MONTH[,i] <- rbeta(nCycles/12, monthWiseP[1,i], monthWiseP[2,i]) \*

 (monthWiseP[4,i] - monthWiseP[3,i]) + monthWiseP[3,i]

 YOUNG <- runif(nCycles, youngP[1], youngP[2]) # random youngP

 OLD <- runif(nCycles, oldP[1], oldP[2]) # random oldP

 inds <- create(nStart)

 data <- data.frame(

 n = numeric(nCycles),

 n1 = numeric(nCycles),

 n2 = numeric(nCycles),

 pop = numeric(nCycles),

 prev = numeric(nCycles),

 prev1 = numeric(nCycles),

 prev2 = numeric(nCycles)

 )

 for (i in 1:nCycles){

 thisMonth <- 1 + (i-1)%%12

 nBirths <- rpois(1, 5)

 envcP <- ENVC[i]

 monthWiseP <- MONTH[nCycles/12, thisMonth]

 youngP <- YOUNG[i]

 oldP <- OLD[i]

 inds <- seroReversion(inds, startReversion, endReversion)

 inds <- ageing(inds)

 inds <- births(inds, nBirths)

 inds <- subsetAlive(inds)

 inds <- seroConversion(inds, envcP, youngP, oldP, monthWiseP)

 data <- census(inds, data, i)

 }

 out <- list(inds = inds, data = data,

 ENVC = ENVC, OLD = OLD, YOUNG = YOUNG, MONTH = MONTH)

 class(out) <- "run"

 return(out)

}

##== RUN CHAINS =============================================================

Model <-

function(nChains, nStart, nCycles,

 envcP, youngP, oldP, monthWiseP,

 startReversion, endReversion){

 ## Define seed and RNG kind

 set.seed(123, kind = "default", normal.kind = "default")

 ## Run chains

 out <- vector("list", nChains)

 for (i in seq\_len(nChains)){

 out[[i]] <- model(nStart, nCycles,

 envcP, youngP, oldP, monthWiseP,

 startReversion, endReversion)

 }

 class(out) <- "run"

 return(out)

}