```
# a function that takes in a base dataset and a desired treatment effect and
# creates the structure for simulated datasets

# returns a data frame that contains 3 variables for each simulated dataset:
  # IDx = ID of patients included in simulated dataset x - should be used to link
      # back to covariate and exposure data for each simulated dataset
  # TIMEx = time until event or censoring for patients in dataset x
  # EVENTx = event status (1 = event, 0 = censored) for patients in dataset x

hdSimSetup <- function(x, idVar, outcomeVar, timeVar, treatVar,
                       form, effectRR = 1, MM = 1, nsim = 500,
                       size = nrow(x), eventRate = NULL)
{
# x = datset on which sims are based
  # idVar = name of id variable
# outcomeVar = name of outcome variable
# timeVar = name of the follow-up time variable
# treatVar = name of treatment variable
# form = RHS of formula used for outcome simulation (excluding treatment of interest)
# effectRR = the desired treatment effect relative risk
# MM = multiplier of confounder effects on outcome on the log-scale
  # nsim = number of desired datasets
  # size = desired size of simulated datasets (i.e., # of individuals)
  # eventRate = desired average event rate -- default is the event
        # rate observed in the base dataset

n <- nrow(x)

sidx <- sapply(c(idVar, outcomeVar, timeVar, treatVar),
                       function(v) which(names(x) == v))
names(x)[sidx] <- c("ID", "OUTCOME", "TIME", "TREAT")
y1 <- Surv(x$TIME, x$OUTCOME)
  y2 <- Surv(x$TIME, !x$OUTCOME)
  form1 <- as.formula(paste("y1 ~ TREAT +", form))
form2 <- as.formula(paste("y2 ~ TREAT +", form))

  # estimate survival and censoring models
smod <- coxph(form1, x = TRUE, data = x)
fit <- survfit(smod)
  s0 <- fit$surv      # survival curve for average patient
  ts <- fit$time
  nts <- length(ts)
  cmod <- coxph(form2, data = x)
  fit <- survfit(cmod)
  c0 <- fit$surv      # censoring curve for average patient

  # find event rate in base cohort (if everyone was followed to end of study)
  Xb <- as.vector(smod$x %*% coef(smod))
mx <- colMeans(smod$x)
xb0 <- mx %*% coef(smod)
  s0end <- min(s0)
if(is.null(eventRate)) eventRate <- 1-mean(s0end^exp(Xb - xb0))

# find delta value needed to get approximate desired event rate under new parameters
  bnew <- replace(MM*coef(smod), names(coef(smod)) == "TREAT", log(effectRR))
  Xbnew <- as.vector(smod$x %*% bnew)
sXend <- s0end^(exp(Xb - xb0))
fn <- function(d) mean(sXend^d) - (1 - eventRate)
delta <- uniroot(fn, lower = 0, upper = 20)$root

  # setup n X nts matrix of individual survival and censoring curves under new parameters
  Sx <- matrix(unlist(lapply(s0, function(s) s^(delta*exp(Xbnew - xb0)))), nrow = n)
  Xbnew <- as.vector(smod$x %*% coef(cmod))
  xb0 <- mx %*% coef(cmod)
  Cx <- matrix(unlist(lapply(c0, function(s) s^(delta*exp(Xbnew - xb0)))), nrow = n)

  #### sample and simulate
ids <- tnew <- ynew <- data.frame(matrix(nrow = size, ncol = nsim))
  for(sim in 1:nsim) {
```

```
    idxs <- sample(n, size, replace = TRUE)
    ids[,sim] <- x$ID[idxs]

    # event time
    u <- runif(size, 0, 1)
    w <- apply(Sx[idxs,] < u, 1, function(x) which(x)[1]) # the first time survival drops below u
    stime <- ts[w]
    w <- Sx[idxs,nts] > u     # for any individuals with survival that never drops below u,
    stime[w] <- max(ts) + 1  # replace with arbitrary time beyond last observed event/censoring time

    # censoring time
    u <- runif(size, 0, 1)
    w <- apply(Cx[idxs,] < u, 1, function(x) which(x)[1]) # the first time censor-free survival drop
    ctime <- ts[w]
    w <- Cx[idxs,nts] > u     # for any individuals with censor-free survival that never drops below
    ctime[w] <- max(ts)    # replace with hard censor time at last observed event/censoring time

    # put it together
    tnew[,sim] <- pmin(stime, ctime)
    names(tnew) <- paste("TIME", 1:nsim, sep = "")
    ynew[,sim] <- stime == tnew[,sim]
    names(ynew) <- paste("EVENT", 1:nsim, sep = "")
  }

  names(ids) <- paste("ID", 1:nsim, sep = "")
names(tnew) <- paste("TIME", 1:nsim, sep = "")
names(ynew) <- paste("EVENT", 1:nsim, sep = "")

data.frame(ids, ynew, tnew)
}
```