```
library("markovchain")

# Retorna TRUE/FALSE dependiendo de si el vector es un
# vector de probabilidad válido de 4 elementos.
validarFila <- function(v) {
  if(length(v) != 4) return(FALSE)
  for (i in 1:length(v)) {
    if(v[i] < 0 || v[i] > 1) return(FALSE)
  }
  if(sum(v) != 1) return(FALSE)
  return(TRUE)
}

# Realiza la simulación con los parámetros especificados,
# y retorna la cantidad de veces que visitó cada estado como
# una tabla.
simulacionMarkov <- function(tMatrix, initial, rep) {
  # Check initial (ejercicio a)
  if(!validarFila(initial)) return(FALSE)

  # Check transitionMatrix (ejericio b)
  filas <- nrow(tMatrix)
  if(filas != 4) return(FALSE)
  for (i in 1:filas) {
    if(!validarFila(tMatrix[i,])) return(FALSE)
  }

  # Hacer la simulación (ejercicio c)
  estados <- c("A", "B", "C", "D")
  MC <- new("markovchain", states=estados, transitionMatrix=tMatrix)
  init <- sample(estados, 1, prob=initial)

  if(length(absorbingStates(MC)) > 0) return(FALSE) # No debe tener estados abso
rbentes
  lista <- rmarkovchain(n=rep, object=MC, t0=init)

  listaPlot <- replicate(length(lista), 0)
  for (i in 1:length(lista)) {
    listaPlot[i] <- match(lista[i], estados)
  }

  # Gráfico (enunciado)
  pdf("markovchain.pdf")
  plot(listaPlot, yaxt="n", type='b')
  axis(2, at=1:4, labels=c("A", "B", "C", "D"))
  dev.off();

  return(table(lista))
}

simulacionMarkov(matrix(c( 0 , .25, .5 , .25,
                           .75,  0 , .12, .13,
                           .1 , .1 ,  0 , .8,
                           .2 , .2 , .6 ,  0), byrow=TRUE, nrow=4), c(1/8, 2/8, 1
/8, 4/8), 20)
```