# Pseudocode for estimating pi using the circle method

set.seed(2053) # remember to set the random number seed so that you can reproduce your output if you end up needing to hunt bugs

# Let's assume the square has length 2, and is centered at (0,0)

# Set circle radius

CircleRadius<-1

NumberOfTrials<-10000 # how many random points to generate

NumberOfPointsInsideCircle<-0 # this will count how often the point lands inside the circle

# do the expteriment - here's one way to do it

# Start the clock

ptm <- proc.time()

for (i in 1:NumberOfTrials){

# generate a random point in the square

XCoordinate<-runif(1,0-CircleRadius,CircleRadius)

YCoordinate<-runif(1,-CircleRadius,CircleRadius)

# check whether it falls within the circle (i.e. the distance to the origina is less than Circle Radius). If so, set NumberOfPointsInsideCircle<-NumberOfPointsInsideCircle+1

if (sqrt(XCoordinate^2 + YCoordinate^2) <= CircleRadius)

{

NumberOfPointsInsideCircle<-NumberOfPointsInsideCircle+1

}

}

ProportionOfPointsInCircle<-NumberOfPointsInsideCircle/NumberOfTrials

# and then do the math to produce you estimate of pi - yours to write

MyPi<-4\*ProportionOfPointsInCircle

# Stop the clock

proc.time() - ptm

# Here's another way -is it quicker than the other version?

# Start the clock

ptm <- proc.time()

AllXCoordinate<-runif(NumberOfTrials,-CircleRadius,CircleRadius)

AllYCoordinate<-runif(NumberOfTrials,-CircleRadius,CircleRadius)

# Now check what proportion of these points fall within the circle - yours to write

# and then do the math to produce you estimate of pi - yours to write

# create a vector of TRUE and FALSE variables of whether each (X,Y) coordinate falls within the circle

MyTruthTable<-sqrt(AllXCoordinate^2 + AllYCoordinate^2) <= CircleRadius

# calculate the proportion

ProportionOfPointsInCircle<-length(MyTruthTable[MyTruthTable==TRUE])/NumberOfTrials

MyPi<-4\*ProportionOfPointsInCircle

# Stop the clock

proc.time() - ptm

# if you are feeling fancy, write some code to plot your points and the circle and square, or your estimate of pi as you perofrm the iterations

# Now you will need to write some code to repeat the above for different sizes of circle (i.e. different values of CircleRadius)

# this will be easiest if you turn the above into functions