R\_biolmol\_A01254805

Daniel Barreras, Yair Beltrán, Victor Symonds

2023-04-26

## PARTE 1

Crea una función que genere una secuencia aleatoria de nucleótidos de ADN (A, T, G y C) de tamaño “n”. Ejecútala solicitando una secuencia aleatoria de ADN de 30 nucleótidos y muestra el resultado impreso en consola.

crear\_secuencia\_adn <- function(n = 30) {  
   
 nucleotidos <- c("A", "T", "G", "C")  
 secuencia <- sample(nucleotidos, n, replace=TRUE)  
   
   
 return(paste(secuencia, collapse=""))  
}  
  
  
secuencia\_aleatoria <- crear\_secuencia\_adn(30);  
cat(secuencia\_aleatoria)

## GAAAGGGGGTACAACCCAGCTTTAACCGCT

Crea una función que calcule el tamaño de una secuencia de ADN. Utilízala para calcular el tamaño de la secuencia que generaste en el punto 1 y muestra el resultado impreso en consola.

calcular\_tamaño\_secuencia <- function(secuencia) {  
 return(nchar(secuencia))  
}  
  
tamaño\_secuencia <- calcular\_tamaño\_secuencia(secuencia\_aleatoria)  
  
cat("El tamaño de la secuencia de ADN es:", tamaño\_secuencia)

## El tamaño de la secuencia de ADN es: 30

Crea una función que recibe una secuencia de DNA e imprime el porcentaje de cada base (A, C, G y T) en la secuencia. Ejecútala sobre la secuencia que generaste en el punto 1 y muestra el resultado impreso en consola.

calcular\_porcentaje\_bases <- function(secuencia){  
   
 n <- nchar(secuencia)  
 a <- 0  
 c <- 0  
 g <- 0  
 t <- 0  
   
 for (i in 1:n) {  
 base <- substr(secuencia, i, i)  
 if (base == "A") {  
 a <- a + 1  
 } else if (base == "C") {  
 c <- c + 1  
 } else if (base == "G") {  
 g <- g + 1  
 } else if (base == "T") {  
 t <- t + 1  
 }  
 }  
  
 por\_a <- a/n \* 100  
 por\_c <- c/n \* 100  
 por\_g <- g/n \* 100  
 por\_t <- t/n \* 100  
   
 cat("Porcentaje de bases A:", round(por\_a, digits = 2), "%\n")  
 cat("Porcentaje de bases C:", round(por\_c, digits = 2), "%\n")  
 cat("Porcentaje de bases G:", round(por\_g, digits = 2), "%\n")  
 cat("Porcentaje de bases T:", round(por\_t, digits = 2), "%\n")  
}  
  
calcular\_porcentaje\_bases(secuencia\_aleatoria)

## Porcentaje de bases A: 30 %  
## Porcentaje de bases C: 26.67 %  
## Porcentaje de bases G: 26.67 %  
## Porcentaje de bases T: 16.67 %

Crea una función que recibe una hebra directa y regresa la hebra inversa. Ejecútala sobre la secuencia que generaste en el punto 1 y muestra el resultado impreso en consola.

calcular\_hebra\_inversa <- function(secuencia){  
 inversa <- ""  
 for (i in nchar(secuencia):1) {  
 inversa <- paste0(inversa, substr(secuencia, i, i))  
 }  
 return(inversa)  
}  
  
hebra\_inversa <- calcular\_hebra\_inversa(secuencia\_aleatoria)  
cat("Secuencia aleatoria: ", secuencia\_aleatoria, "\n")

## Secuencia aleatoria: GAAAGGGGGTACAACCCAGCTTTAACCGCT

cat("Hebra inversa: ", hebra\_inversa)

## Hebra inversa: TCGCCAATTTCGACCCAACATGGGGGAAAG

Crea una función qué recibe una hebra directa y obtiene la hebra complementaria. Ejecútala sobre la secuencia que generaste en el punto 1 y muestra el resultado impreso en consola

calcular\_hebra\_complementaria <- function(secuencia){  
 complementaria <- ""  
 for (i in 1:nchar(secuencia)) {  
 base <- substr(secuencia, i, i)  
 if (base == "A") {  
 complementaria <- paste0(complementaria, "T")  
 } else if (base == "T") {  
 complementaria <- paste0(complementaria, "A")  
 } else if (base == "G") {  
 complementaria <- paste0(complementaria, "C")  
 } else if (base == "C") {  
 complementaria <- paste0(complementaria, "G")  
 }  
 }  
 return(complementaria)  
}  
  
hebra\_complementaria <- calcular\_hebra\_complementaria(secuencia\_aleatoria)  
  
cat("Secuencia aleatoria: ", secuencia\_aleatoria, "\n")

## Secuencia aleatoria: GAAAGGGGGTACAACCCAGCTTTAACCGCT

cat("Hebra complementaria: ", hebra\_complementaria)

## Hebra complementaria: CTTTCCCCCATGTTGGGTCGAAATTGGCGA

## PARTE 2

Crea una función que transcribe ADN a ARN. Es decir, que recibe una secuencia molde de nucleótidos de ADN y devuelve la secuencia del transcrito correspondiente que se transcribiría a partir de dicha secuencia. Ejecútala sobre la secuencia que generaste en el punto 1 y muestra el resultado impreso en consola.

transcribir\_adn\_arn <- function(secuencia) {  
 arn <- ""  
 for(i in 1:nchar(secuencia)){  
 base <- substr(secuencia, i, i)  
 if(base == "T"){  
 arn <- paste0(arn, "U")  
 }else{  
 arn <- paste0(arn, base)  
 }  
 }  
 return(arn)  
}  
  
arn\_transcrito <- transcribir\_adn\_arn(secuencia\_aleatoria)  
cat("Secuencia aleatoria: ", secuencia\_aleatoria, "\n")

## Secuencia aleatoria: GAAAGGGGGTACAACCCAGCTTTAACCGCT

cat("Secuencia transcrita a ARN: ", arn\_transcrito)

## Secuencia transcrita a ARN: GAAAGGGGGUACAACCCAGCUUUAACCGCU

Crea una función que te sirva para encontrar codones de inicio y de terminación en una secuencia dada de ARN. A la secuencia presente entre un codón de inicio y uno de terminación en un gen se le conoce como marco de lectura. Ejecútala sobre la secuencia que generaste en el punto anterior (6) y muestra el resultado impreso en consola.

encontrar\_codones <- function(secuencia){  
 codon\_inicio = "AUG"  
 codon\_final = c("UAA", "UAG", "UGA")  
   
 secuencia <- paste0(codon\_inicio, secuencia, "UAAUAGUGA")  
  
 codones <- c();  
   
   
 for(i in seq(1, nchar(secuencia), by=3)){  
   
 j <- (i+2)/3;  
   
 codones[j] <- paste0(substr(secuencia, i, i), substr(secuencia, i+1, i+1), substr(secuencia, i+2, i+2));  
 }  
 cat("Codones de la secuencia: ", codones, "\n")  
   
 for(i in 1:length(codones)){  
 if(codones[i] == codon\_inicio){  
 cat("Marco de lectura: START, ")  
 }else if(codones[i] %in% codon\_final){  
 cat("STOP")  
 break  
 } else {  
 cat(codones[i], ", ")  
 }  
 }  
   
 return(codones)  
}  
  
codones <- encontrar\_codones(arn\_transcrito)

## Codones de la secuencia: AUG GAA AGG GGG UAC AAC CCA GCU UUA ACC GCU UAA UAG UGA   
## Marco de lectura: START, GAA , AGG , GGG , UAC , AAC , CCA , GCU , UUA , ACC , GCU , STOP

Crea una función que traduce una secuencia de ARN a una secuencia de aminoácidos (proteína), siguiendo para ello las reglas del código genético y la representación en una letra de los aminoácidos.

traducir\_arn <- function(codones){  
 tabla\_aminoacidos <- data.frame(  
 codon = c("UUU", "UUC", "UUA", "UUG", "CUU", "CUC", "CUA", "CUG", "AUU", "AUC", "AUA", "AUG",  
 "GUU", "GUC", "GUA", "GUG", "UCU", "UCC", "UCA", "UCG", "CCU", "CCC", "CCA", "CCG",  
 "ACU", "ACC", "ACA", "ACG", "GCU", "GCC", "GCA", "GCG", "UAU", "UAC", "UAA", "UAG",  
 "CAU", "CAC", "CAA", "CAG", "AAU", "AAC", "AAA", "AAG", "GAU", "GAC", "GAA", "GAG",  
 "UGU", "UGC", "UGA", "UGG", "CGU", "CGC", "CGA", "CGG", "AGU", "AGC", "AGA", "AGG",  
 "GGU", "GGC", "GGA", "GGG"),  
 amino\_acido = c("F", "F", "L", "L", "L", "L", "L", "L", "I", "I", "I", "M",  
 "V", "V", "V", "V", "S", "S", "S", "S", "P", "P", "P", "P",  
 "T", "T", "T", "T", "A", "A", "A", "A", "Y", "Y", "STOP", "STOP",  
 "H", "H", "Q", "Q", "N", "N", "K", "K", "D", "D", "E", "E",  
 "C", "C", "STOP", "W", "R", "R", "R", "R", "S", "S", "R", "R",  
 "G", "G", "G", "G")  
 )  
   
 cat("START ")  
 for(i in seq\_along(codones)){  
   
 codon\_encontrado <- codones[i] == tabla\_aminoacidos$codon  
 posicion <- which(codon\_encontrado == TRUE)  
   
 if(tabla\_aminoacidos$amino\_acido[posicion] == "STOP"){  
 break  
 }  
 cat(tabla\_aminoacidos$amino\_acido[posicion], " ")  
 }   
 cat("STOP")  
}  
  
traducir\_arn(codones)

## START M E R G Y N P A L T A STOP