Tarea 1

Problema 9

Primero definimos una función que genere los primeros n dígitos de la secuencia de Fibonacci modulo 5.

```
genFibonacci <- function(x0,x1,m,n)
{
  aux<-c(x0,x1)
  for (i in 1:n)aux<-c(aux,((aux[i]+aux[i+1]) %% m))
  return(aux)
}
```

De probar con diferentes combinaciones de números, concluímos que hay dos ciclos diferentes:

El primero, observado con semillas como $x_0 = 0$ y $x_1 = 1$, y $x_0 = 6$ y $x_1 = 7$ es la siguiente secuencia de **20** dígitos

```
genFibonacci(0,1,5,18)
```

```
##  [1] 0 1 1 2 3 0 3 3 1 4 0 4 4 3 2 0 2 2 4 1
```

El segundo ciclo, observado para semillas como $x_0 = 3$ y $x_1 = 4$, y $x_0 = 8$ y $x_1 = 9$ es la siguiente secuencia de **4** dígitos

```
genFibonacci(3,4,5,2)
```

```
## [1] 3 4 2 1
```

Problema 10

Definimos una función que genere el siguiente número cuadrado medio de Neumann.

```
cmNeumann<-function(z0)
{
  aux<-toString(z0^2)
  if (nchar(aux)<4)
  {
    aux<-paste(0,0,0,aux, sep="")
    aux<-substr(aux, nchar(aux)-3, nchar(aux))
  }
  return(as.numeric(substring(aux,2,3)))
}
```

Ahora, hacemos un programa que identifique los bucles en la generación de números aleatorios para semillas entre 0 y 99.

```
n <- 99
a <- matrix(, nrow = 65, ncol = 15)
j <- 1;i <- 1;k <- 1
a[i,j] <- 0
```

```r
while(k < n + 1)
{
  aux <- cmNeumann(a[j,i])
  if(aux == a[j,i] || (i>2 && aux == a[j,i-1])) #Bucle de periodo 1 o 2
  {
    j <- j + 1
    i <- 1
    k <- j - 1
    while(k %in% a)
    {
      k <- k + 1
    }
    if(j < nrow(a) + 1)a[j,i] <- k
  }
  else
  {
    i <- i + 1
    a[j,i] <- aux
  }
}
```

Table 1: Secuencias del Generador de Cuadrados Medios de Neumann

0										
1	0									
2	0									
3	0									
4	1	0								
5	2	0								
6	3	0								
7	4	1	0							
8	6	3	0							
9	8	6	3	0						
10										
11	12	14	19	36	29	84	5	2	0	
13	16	25	62	84	5	2	0			
15	22	48	30	90	10					
17	28	78	8	6	3	0				
18	32	2	0							
20	40	60								
21	44	93	64	9	8	6	3	0		
23	52	70	90	10						
24	57	24								
26	67	48	30	90	10					
27	72	18	32	2	0					
31	96	21	44	93	64	9	8	6	3	0
33	8	6	3	0						
34	15	22	48	30	90	10				
35	22	48	30	90	10					
37	36	29	84	5	2	0				
38	44	93	64	9	8	6	3	0		
39	52	70	90	10						
41	68	62	84	5	2	0				

Table 1: Secuencias del Generador de Cuadrados Medios de Neumann

42	76	77	92	46	11	12	14	19	36	29	84	5	2	0
43	84	5	2	0										
45	2	0												
47	20	40	60											
49	40	60												
50														
51	60													
53	80	40	60											
54	91	28	78	8	6	3	0							
55	2	0												
56	13	16	25	62	84	5	2	0						
58	36	29	84	5	2	0								
59	48	30	90	10										
61	72	18	32	2	0									
63	96	21	44	93	64	9	8	6	3	0				
65	22	48	30	90	10									
66	35	22	48	30	90	10								
69	76	77	92	46	11	12	14	19	36	29	84	5	2	0
71	4	1	0											
73	32	2	0											
74	47	20	40	60										
75	62	84	5	2	0									
79	24	57												
81	56	13	16	25	62	84	5	2	0					
82	72	18	32	2	0									
83	88	74	47	20	40	60								
85	22	48	30	90	10									
86	39	52	70	90	10									
87	56	13	16	25	62	84	5	2	0					
89	92	46	11	12	14	19	36	29	84	5	2	0		
94	83	88	74	47	20	40	60							
95	2	0												
97	40	60												
98	60													
99	80	40	60											

Nota: El último número en cada secuencia es el inicio de un bucle

Comentarios sobre el código:

Variables:

- n: Número máximo evaluado en la función de cuadrados medios de Neumann
- a: Matriz que almacena las secuencias y bucles
- j: Contador de renglones
- i: Contador de columnas
- k: Número con el que iniciará la siguiente secuencia
- El programa funciona gracias a un *while* que corre mientras todavia hayan secuencias posibles a generar.
- Calculamos el `cmNeumann` de la entrada correspondiente al (i,j) en cuestión:

  – Si el `cmNeumann` es igual a la semilla, significa que hay un bucle de periodo 1.
  – Si el `cmNeumann` es igual a la entrada anterior a la semilla, significa que hay un bucle 2.

- En cada renglón de la matriz hay una secuencia.

Problema 11

Empezamos por definir una función que a partir de un *string*, almacene por separado los caracteres que lo componen.

```
digits <- function(digitos)
{
  aux<-numeric(0)
  for (i in 1:nchar(digitos))aux<-c(aux,as.numeric(substring(digitos, i, i)))
  return(aux)
}
```

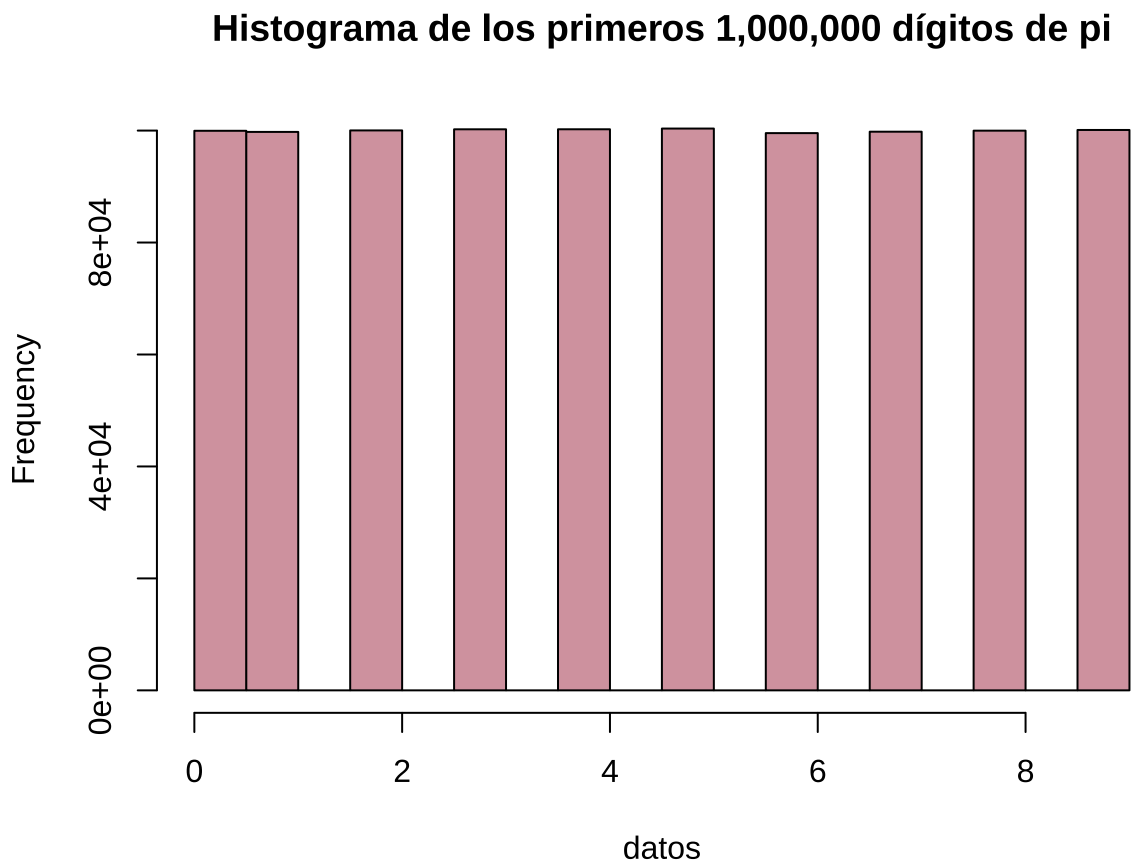
Ahora leemos el archivo *pi.txt* con el primer millón de dígitos de π; con ayuda de la funcion de definimos procesamos los datos para poder usarlos en pruebas de uniformidad e independencia. La razón por la cual trabajamos con *strings* (en lugar de *int*) es para evitar un `intOverflow`.

```
fileName <- "pi.txt"
w<-readChar(fileName, file.info(fileName)$size)
```

```
#w<-substring(w, 1, 100000)
datos<-digits(w)
```

Hacemos un histograma de los datos y vemos que parecen distribuirse de forma uniforme discreta, tal como esperábamos.

```
#Histograma
hist(datos, col='pink3',main = "Histograma de los primeros 1,000,000 dígitos de pi")
```


Histograma de los primeros 1,000,000 dígitos de pi
Frequency
0e+00
4e+04
8e+04
0
2
4
6
8
datos

Problema 12

12. Sea X_1 : cara que cae en dado 1
X_2 : cara que cae en dado 2

Si para el dado 1 el valor 1 aparece el doble de veces

$\Rightarrow$ la probabilidad de que salga 1 es el doble que para cualquier otro número, así tenemos que:

$$\begin{array}{ll} \mathbb{P}(X_1 = 1) = 2x & \mathbb{P}(X_1 = 4) = x \\ \mathbb{P}(X_1 = 2) = x & \mathbb{P}(X_1 = 5) = x \\ \mathbb{P}(X_1 = 3) = x & \mathbb{P}(X_1 = 6) = x \end{array}$$

como $\sum_{i=1}^{6} \mathbb{P}(X_1 = i) = 7x = 1 \Rightarrow x = \frac{1}{7}$

para el dado 2 ocurre lo mismo para el valor de 6, $\therefore$

$$\begin{array}{ll} \mathbb{P}(X_2 = 1) = 1/7 & \mathbb{P}(X_4 = 4) = 1/7 \\ \mathbb{P}(X_2 = 2) = 1/7 & \mathbb{P}(X_5 = 5) = 1/7 \\ \mathbb{P}(X_3 = 3) = 1/7 & \mathbb{P}(X_6 = 6) = 2/7 \end{array}$$

Así,

$$\begin{aligned} p_s &= \mathbb{P}(X_1 + X_2 = s) \\ &= \mathbb{P}(X_2 = s - X_1) \\ &= \mathbb{P}(X_2 = s - X_1 \mid X_1 = 1)\,\mathbb{P}(X_1 = 1) + \dots + \mathbb{P}(X_2 = s - X_1 \mid X_1 = 6)\,\mathbb{P}(X_1 = 6) \\ &= \sum_{i=1}^{6} \mathbb{P}(X_2 = s - i \mid X_1 = i)\,\mathbb{P}(X_1 = i) \\ &= \sum_{2}^{6} \frac{1}{7}\mathbb{P}(X_2 = s - i \mid X_1 = i) + \frac{2}{7}\mathbb{P}(X_2 = s - 1 \mid X_1 = 1) \\ &= \sum_{2}^{6} \frac{1}{7}\mathbb{P}(X_2 = s - i) + \frac{2}{7}\mathbb{P}(X_2 = s - 1) \end{aligned}$$

// por ley de prob. total

Hay indep entre $X_2 = s-i$ y $X_i = i$

con $\mathbb{P}(X_2 = i) = \begin{cases} \frac{1}{7} & i = 1, \dots, 5 \\ \frac{2}{7} & i = 6 \\ 0 & \text{c.o.c.} \end{cases}$