UNIVERSIDAD AUTÓNOMA METROPOLITANA
DIVISIÓN DE CIENCIAS BÁSICAS E INGENIERÍA
UNIDAD AZCAPOTZALCO

Laboratorio de Análisis y simulación de sistemas

Práctica 3.

Realizado por:

- **GABRIEL FRANCISCO RAMOS 209302867**

Profesor:

ANTONIN SEBASTIEN PONSICH

FECHA DE ENTREGA: 03 de Noviembre de 2011

TRIMESTRE: 11-Otoño

GRUPO: CSI02

INDICE **PAG.**

Parte 1: Método de las medias

Parte 2: Método de éxito – fracaso.

P R E S E N T A C I Ó N

En esta práctica está dedica a la integración de Monte Carlo, generando dos vectores u (i) y v (i), exactamente lo mismo como se viene haciendo desde la práctica 1, en la primera parte de dicha práctica se tiene la intención de calcular la estimación de la integral a través de los dos vectores generados haciendo un cambio de variable dentro del intervalo 0 y 1.

$$\theta = \int_0^{+\infty} \int_0^{x} e^{-x^2-y^2} dydx$$

Mientras que en la segunda parte de la práctica, se obtiene una estimación del valor del número π, usando la función de un círculo inscrito en un cuadrado centrado en el origen (0,0) y de lado con longitud igual a $x^2 + y^2 = 1$, con radio igual a 1.

Parte 1: Método de las medias

1. ***Código fuente.***

```
Option Explicit

Sub aleatorios()

'declaración de variables

Dim a As Double, c As Double, m As Double, x0 As Double, x01 As Double, integral As Double

Dim x As Double, x1 As Double, g() As Double, suma As Double, g2 As Double, ee As Double

Dim i As Integer, N As Integer, u() As Double, v() As Double, desv As Double, desvi As Double

N = InputBox("¿Número aleatorios a generar?") 'pedir valores al usuario de cada uno de las variables

'declarar los parámetros como constantes

a = 214013

m = 2 ^ 32

c = 2531011

x0 = 245

x01 = 2 ^ 18 - 1

ReDim u(N) As Double 'redimensionar el vector u

'hacer el siguiente ciclo a partir de 1 hasta N números de aleatorios a generar

For i = 1 To N

x = Int((a * x0 + c) / m) 'hacer la primera operación

x1 = (a * x0 + c) - x * m 'hacer la segunda operación utilizando el valor de x que salió en la primera operación.

u(i) = x1 / m

x0 = x1                    'x0  toma el valor de x1 que salió de la operación anterior

Cells(i + 1, 1).Value = u(i) 'se imprime todos los números aleatorios en la columna Ai

Next i

ReDim v(N) As Double
```

```
'otro ciclo que permite generar otro vector de N números aleatorios
For i = 1 To N
x = Int((a * x01 + c) / m) 'hacer la primera operación
x1 = (a * x01 + c) - x * m 'hacer la segunda operación utilizando el valor de x que salió en la primera operación.
v(i) = x1 / m
x01 = x1                    'x01  toma el valor de x1 que salió de la operación anterior
Cells(i + 1, 2).Value = v(i) 'se imprime todos los números aleatorios en la columna Bi
Next i
ReDim g (N) As Double
'ciclo que permite calcular el vector g (i) a partir de los dos vectores de números aleatorios
Cells(1, 4).Value = "g(i)"
For i = 1 To N
g2 = Exp(-(1 / u(i) - 1) ^ 2) * Exp(-((1 / u(i) - 1) * v(i)) ^ 2)
g(i) = g2 * (1 / u(i) - 1) / u(i) ^ 2
Cells(i + 1, 4).Value = g(i) 'los valores se imprime en la columna Di
Next i
'ciclo que permite calcular el valor de la integral
suma = 0
For i = 1 To N
suma = suma + g(i)
Next i
integral = suma / N
Cells(1, 6).Value = "valor de la integral"
Cells(2, 6).Value = integral 'el resultado se imprime en la columna F2
desv = 0
```

```
'ciclo que permite calcular el error estándar asociado

For i = 1 To N

desv = desv + (g(i) - integral) ^ 2

Next i

desvi = desv / (N - 1)

ee = 1 / N ^ 0.5 * Sqr(desvi)

Cells(1, 8).Value = "error estándar asociado"

Cells(2, 8).Value = ee ' el resultado se imprime en la columna H2"

MsgBox ("¡La ejecución ha terminado exitosamente!") 'mensaje que le avisa al usuario que la ejecución a terminado

End Sub
```

2. Descripción del código fuente.

El programa anterior tiene como finalidad generar dos vectores de números aleatorios y a través de esos vectores generar otro vector g(i), para calcular el valor de la integral y el valor del error estándar asociado.

Nota. En la parte de generación de números aleatorios es exactamente lo mismo al de la práctica 1, solo que en este caso se genera dos vectores de números aleatorios u (i) y v (i), donde los parámetros (constantes) a, c y m no cambian para los dos vectores, lo único que cambia es valor de la semilla, para el vector u(i) el valor de semilla es x0=245 y el vector v(i) es X01= 2 ^ 18 – 1.

A continuación se hace una lista de todas las variables que se utilizó en la elaboración del código fuente.

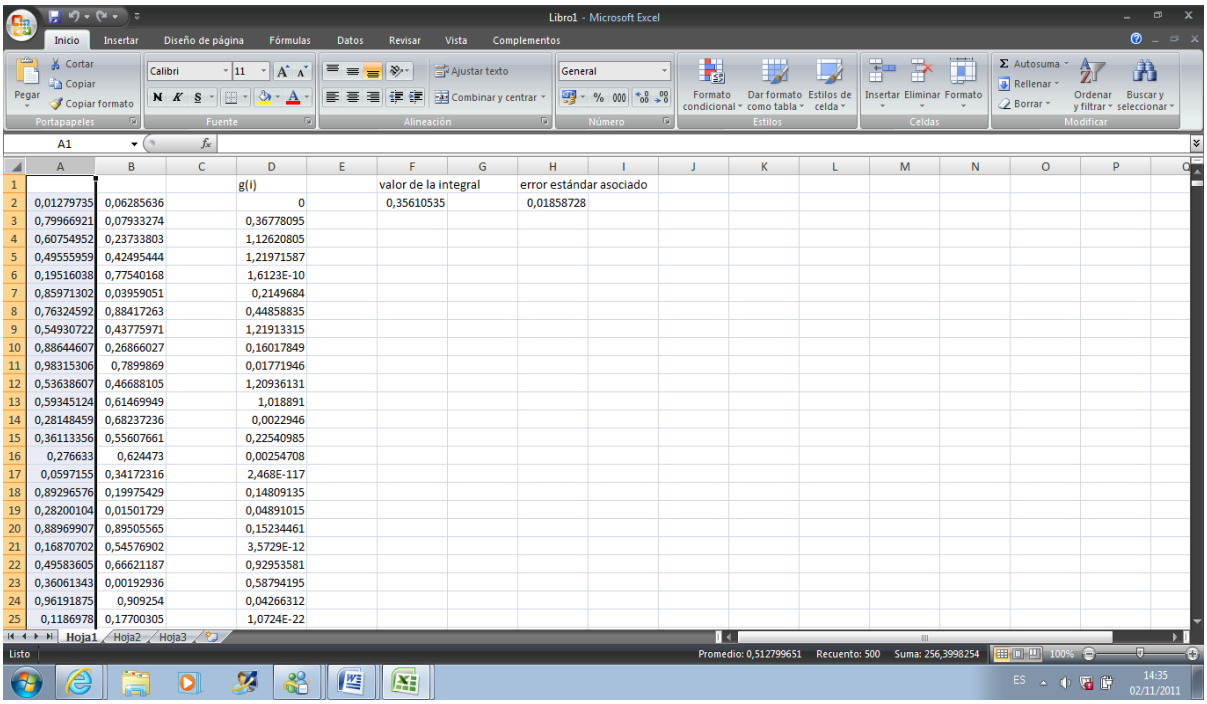
Variables	Tipo de variables	Uso
a	Double (doble precisión)	Parámetro declarado (multiplicador)
c	Double	Parámetro declarado (incremento)
m	Double	Parámetro declarado (modulo)
x0	Double	Parámetro declarado(semilla)
x01	Double	Parámetro declarado (semilla)
x	Double	Almacena temporalmente los datos
x1	Double	Almacena temporalmente los datos
i	Integer (entero)	Contador
N	Integer	Número de aleatorios a generar (solicitado)
u()	Double	Vector, almacena los datos
v()	Double	Vector, almacena los datos
integral	Double	Almacena un dato (valor de la integral)
g()	Double	Vector, almacena los datos
suma	Double	Almacena un dato

g2	Double	Almacena temporalmente un dato
ee	Double	Almacena un dato (error estándar)
desv	Double	Almacena un dato
desvi	Double	Almacena un dato

Tabla 1: se describen y se muestran todas las variables utilizados en la elaboración del programa.

- ✓ Declaración de variables.
- ✓ Solicitud de valores (parámetros) necesarias durante la ejecución de programa.
- ✓ Generación de N números aleatorios para dos vectores u (i) y v (i) (lo mismo que se describe en la práctica 1 y con las especificaciones arriba mencionado).
- ✓ Obtención del vector g (i) a través de los vectores u (i) y v (i). Para eso se hace un ciclo de 1 hasta N números aleatorios, usando la formula siguiente $\theta = \left(\frac{1}{u(i)} - 1\right) \varepsilon^{-\left(\frac{1}{u(i)}-1\right)^2 - \left(\left(\frac{1}{u(i)}-1\right)v(i)\right)^2} / u(i)^2$, como se muestra en el código fuente.
- ✓ Después se obtiene el valor de la integral. Se hace un ciclo que permita sumar todos los números g (i) generados con anterioridad y después dividirlo entre los N números aleatorios generados, donde el resultado se imprime en la celda F2 de la hoja de cálculo (Excel).
- ✓ A continuación se calcula el valor del error estándar asociado. También se hace un ciclo de 1 hasta N, usando la formula siguiente $ee = \frac{1}{N\ \hat{}0.5} \sqrt[2]{\sum_1^N \frac{(g(i)-\theta)^2}{N-1}}$, como se muestra en el código fuente y el resultado se imprime en la columna o celda H2 de la hoja de cálculo (Excel).
- ✓ Y por último aparece una ventana de Windows que indica que la ejecución del programa ha terminado exitosamente.

3. *Corrida con N=500.*



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
1				g(i)		valor de la integral		error estándar asociado								
2	0,01279735	0,06285636		0		0,35610535		0,01858728								
3	0,79966921	0,07933274		0,36778095												
4	0,60754952	0,23733803		1,12620805												
5	0,49555959	0,42495444		1,21971587												
6	0,19516038	0,77540168		1,6123E-10												
7	0,85971302	0,03959051		0,2149684												
8	0,76324592	0,88417263		0,44858835												
9	0,54930722	0,43775971		1,21913315												
10	0,88644607	0,26866027		0,16017849												
11	0,98315306	0,7899869		0,01771946												
12	0,53638607	0,46688105		1,20936131												
13	0,59345124	0,61469949		1,018891												
14	0,28148459	0,68237236		0,0022946												
15	0,36113356	0,55607661		0,22540985												
16	0,276633	0,624473		0,00254708												
17	0,0597155	0,34172316		2,468E-117												
18	0,89296576	0,19975429		0,14809135												
19	0,28200104	0,01501729		0,04891015												
20	0,88969907	0,89505565		0,15234461												
21	0,16870702	0,54576902		3,5729E-12												
22	0,49583605	0,66621187		0,92953581												
23	0,36061343	0,00192936		0,58794195												
24	0,96191875	0,909254		0,04266312												
25	0,1186978	0,17700305		1,0724E-22												

Figura 1: simulación realizada en VBA con N=500 y los números aleatorios se imprimen en la hoja de cálculo (Excel). El valor de la integral es 0.35610535 y el error estándar asociado es 0.01858728.

4. *Simulación variando N números aleatorios (500 a 10,000).*

N (números aleatorios)	Valor de la integral	Error estándar asociado
500	0.35610535	0.01858728
2000	0,39044255	0,00956591
4000	0,38432062	0,00680323
6000	0,38984343	0,0056085
8000	0,39159071	0,00483959
10000	0,39109436	0,00432327

Tabla 2: Muestra las 6 corridas con sus respectivos valores de la integral y el error estándar asociado, variando el valor de N.

5. *Conclusión.*

Basándose en los valores del error estándar asociado (para cada valor de N), que nos permite deducir la confiabilidad de la ecuación de estimación de la integral citado anteriormente, se puede decir que con N=10,000 números aleatorios tenemos un valor más cercano o verdadero (más confiable) de la integral, mientras que con N < 500 números aleatorios el valor de la integral está muy alejado del valor correcto. Y en resumen ya es recomendable aceptar el valor de la integral con N>4000 números aleatorios debido a que el valor que nos arroja está muy próximo al valor real.

Parte 2: Método de éxito – fracaso.

6. Código fuente.

```
Option Explicit

Sub aleatorios()

'declaración de variables

Dim a As Double, c As Double, m As Double, x0 As Double, x01 As Double, area As Double

Dim x As Double, x1 As Double, circulo() As Double, y() As Double, suma As Double

Dim i As Integer, N As Integer, u() As Double, v() As Double, suma1 As Double, pi As Double

N = InputBox("¿Número aleatorios a generar?") 'valor solicitado al usuario

'declaración de parámetros como constantes

a = 214013

m = 2 ^ 32

c = 2531011

x0 = 245

x01 = 2 ^ 18 - 1

ReDim u(N) As Double 'redimensionar el vector u

'hacer el siguiente ciclo a partir de 1 hasta N números de aleatorios a generar

For i = 1 To N

x = Int((a * x0 + c) / m) 'hacer la primera operación

x1 = (a * x0 + c) - x * m 'hacer la segunda operación utilizando el valor de x que salió en la primera operación.

u(i) = x1 / m

x0 = x1                    'x0  toma el valor de x1 que salió de la operación anterior

Cells(i + 1, 1).Value = u(i) 'se imprime todos los números aleatorios en la columna Ai

Next i

ReDim v(N) As Double
```

```
'ciclo que permite generar otro vector de N números aleatorios

For i = 1 To N

x = Int((a * x01 + c) / m) 'hacer la primera operación

x1 = (a * x01 + c) - x * m 'hacer la segunda operación utilizando el valor de x que salió en la primera operación.

v(i) = x1 / m

x01 = x1                'x01  toma el valor de x1 que salió de la operación anterior

Cells(i + 1, 2).Value = v(i) 'se imprime todos los números aleatorios en la columna Bi

Next i

ReDim circulo(N) As Double

'cilo que permite evaluar la funcion u(i)^2 + v(i)^2,utilizando los dos vectores de números aleatorios

For i = 1 To N

circulo(i) = u(i) ^ 2 + v(i) ^ 2

Cells(i + 1, 4).Value = circulo(i)

Next i

ReDim y(N) As Double

Cells(1, 6).Value = "Y(i)"

For i = 1 To N

If circulo(i) <= 1 Then 'y(i)=1 si los valores de circulo(i) generados anteriormente se encuentra en el área del circulo

y(i) = 1

Cells(i + 1, 6).Value = y(i)

ElseIf circulo(i) > 1 Then 'y(i)=0 en caso contrario

y(i) = 0

Cells(i + 1, 6).Value = y(i) ' todos los valores de y(i) se imprimen en la columna Fi

End If
```

```
Next i

suma = 0

suma1 = 0

For i = 1 To N

If y(i) = 1 Then 'conteo de los números de éxitos o los valores y(i)=1 que se encuentran en el área del circulo

suma = suma + 1

ElseIf y(i) = 0 Then 'conteo de los fracasos o los valores de y(i)=o que se encuentran fuera del área del circulo

suma1 = suma1 + 1

End If

Next i

Cells(1, 7).Value = "Numero de éxitos"

Cells(2, 7).Value = suma 'se imprime el total de éxitos

Cells(1, 9).Value = "Numero de Fracasos"

Cells(2, 9).Value = suma1 'se imprime el total de fracasos

pi = (suma / N) * 4 'el total de éxitos se divide entre N y se multiplica por 4 y el resultado es el valor de Pi

Cells(2, 11).Value = pi 'el valor de Pi se imprime en la celda J2

Cells(1, 11).Value = "Pi (estimación)"

MsgBox ("¡La ejecución ha terminado exitosamente!") 'mensaje que le avisa al usuario que la ejecución a terminado

End Sub
```

7. Descripción del código fuente.

El programa anterior tiene como finalidad calcular la estimación del número π, usando la función de un círculo inscrito en un cuadrado centrado en el origen (0,0) y de lado con longitud igual a $x^2 + y^2 = 1$.

Nota. Es esta parte también se genera dos vectores u(i) y v(i) y nuevamente la generación de estos vectores de números aleatorios es exactamente lo mismo al de la práctica 1, los

parámetros (constantes) a, c y m no cambian para los dos vectores, lo único que cambia es valor de la semilla, para el vector u(i) el valor de semilla es x0=245 y el vector v(i) es X01= 2 ^ 18 – 1.

A continuación se hace una lista de todas las variables que se utilizó en la elaboración del código fuente.

Variables	Tipo de variables	Uso
a	Double (doble precisión)	Parámetro declarado (multiplicador)
c	Double	Parámetro declarado (incremento)
m	Double	Parámetro declarado (modulo)
x0	Double	Parámetro declarado(semilla)
x01	Double	Parámetro declarado (semilla)
x	Double	Almacena temporalmente los datos
x1	Double	Almacena temporalmente los datos
i	Integer (entero)	Contador
N	Integer	Número de aleatorios a generar (solicitado)
u()	Double	Vector, almacena los datos
v()	Double	Vector, almacena los datos
circulo()	Double	Vector, almacena los datos
y()	Double	Vector, almacena los datos
suma	Double	Almacena datos
suma1	Double	Almacena datos
pi	Double	Almacena datos (Valor de π)

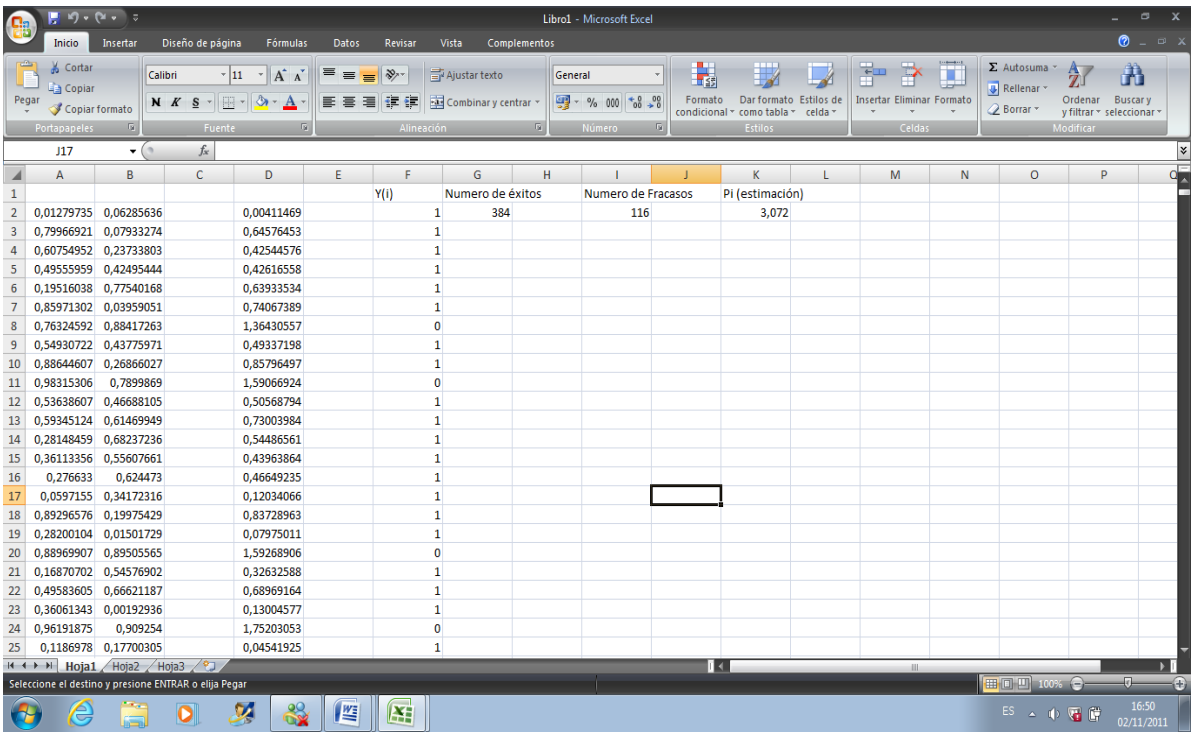
Tabla 3: se describen y se muestran todas las variables utilizados en la elaboración del programa.

- ✓ Declaración de variables.
- ✓ Solicitud de valores (parámetros) necesarias durante la ejecución de programa. Generación de N números aleatorios para dos vectores u (i) y v (i) (lo mismo que se describe en la práctica 1 y con las especificaciones arriba mencionado).
- ✓ Generación del vector circulo (i), usando los vectores u (i) y v (i). Para esto se hace un ciclo de 1 hasta N números aleatorios, usando la fórmula siguiente: circulo (i)= u (i) ^ 2 + v (i) ^ 2 (función del circulo inscrito en un cuadrado centrado en el origen (0,0) y con radio igual a 1) y los datos se imprime en la columna Di de la hoja de cálculo (Excel).
- ✓ Generación del vector y (i), usando los datos del vector circulo (i). Para esto se hace un ciclo se 1 hasta N (que permite recorrer todos los datos de vector circulo (i)), y (i) es igual a 1 si los valores de circulo (i) generados anteriormente se encuentra en el área del circulo con radio igual 1 (y (i) <=1). Y y(i) es igual a 0 en caso contrario, si los valores del circulo están fuera del área del circulo (y (i) >1) y todos los valores de y(i) se imprimen en la columna Fi de la hoja de cálculo (Excel).
- ✓ Conteo del numero de éxitos, es decir todos los valores de y (i)=1, que se encuentran dentro del área del circulo. Para esto se hace otro ciclo que permita recorrer todos los datos de y (i) de 1 hasta N y mientras y (i)=1 se hace el coteo, hasta encontrar el número total de éxitos (una vez recorrido los N números o datos).
- ✓ Obtención de la estimación del número π. Para esto el numero de éxitos encontrados en el ciclo anterior se divide entre N y este valor representa la cuarta parte del valor

de π, así que a continuación se multiplica por 4 y se obtiene la estimación de π y el resultado se imprime en la celda K2 de la hoja de cálculo (Excel).

- ✓ Y por último aparece una ventana de Windows que indica que la ejecución del programa ha terminado exitosamente.

8. *Corrida con N=500.*



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
1						Y(i)	Numero de éxitos		Numero de Fracasos		Pi (estimación)					
2	0,01279735	0,06285636		0,00411469		1	384		116		3,072					
3	0,79966921	0,07933274		0,64576453		1										
4	0,60754952	0,23733803		0,42544576		1										
5	0,49555959	0,42495444		0,42616558		1										
6	0,19516038	0,77540168		0,63933534		1										
7	0,85971302	0,03959051		0,74067389		1										
8	0,76324592	0,88417263		1,36430557		0										
9	0,54930722	0,43775971		0,49337198		1										
10	0,88644607	0,26866027		0,85796497		1										
11	0,98315306	0,7899869		1,59066924		0										
12	0,53638607	0,46688105		0,50568794		1										
13	0,59345124	0,61469949		0,73003984		1										
14	0,28148459	0,68237236		0,54486561		1										
15	0,36113356	0,55607661		0,43963864		1										
16	0,276633	0,624473		0,46649235		1										
17	0,0597155	0,34172316		0,12034066		1										
18	0,89296576	0,19975429		0,83728963		1										
19	0,28200104	0,01501729		0,07975011		1										
20	0,88969907	0,89505565		1,59268906		0										
21	0,16870702	0,54576902		0,32632588		1										
22	0,49583605	0,66621187		0,68969164		1										
23	0,36061343	0,00192936		0,13004577		1										
24	0,96191875	0,909254		1,75203053		0										
25	0,1186978	0,17700305		0,04541925		1										

Figura 2: simulación realizada en VBA con N=500 y los números aleatorios se imprimen en la hoja de cálculo (Excel). El numero de éxitos es 384 y el valor de Pi= 3.072.

9. *Simulación variando N números aleatorios (500 a 10,000).*

N (números aleatorios)	Valor de estimación de π	Error relativo porcentual (Ep)
500	3.072	2.215203 %
2000	3.13	0.369005 %
4000	3.1384	0.101625 %
6000	3.14133333	0.008254 %
8000	3.1285	0.416752 %
10000	3.144	0.076628 %

Tabla 4: Muestra las 6 corridas con sus respectivos valores de estimación de π y el error relativo porcentual (Ep), variando el valor de N.

Nota: El error relativo porcentual se calcula de la siguiente manera.

$$Ep = \frac{Valor\ teórico - IValor\ \ simuladoI}{Valor\ teórico}\ (100\%)$$

Valor teórico de π=3.14159265……

10. Conclusión.

Basándose en el error relativo porcentual (Ep), cuando N<500 números aleatorios el valor de la estimación de π está muy alejado del valor teórico y dicho resultado no es confiable, mientras que cuando N=6000 números aleatorios el valor de la estimación de π es confiable porque hay un acercamiento importante del valor simulado con el valor teórico y se puede comprobar o corroborar con el valor del error relativo porcentual mostrado en la tabla 4. No se puede decir que entre más grande sea N, tendremos un valor de estimación cercano al valor teórico, por que se estaría cometiendo un error y muestra de ello son los datos, cuando N=8000 o N=10000, nos alejamos un poco del valor real y cuando simulamos con N números aleatorios mayores a 10000 la simulación se vuelve muy tedioso (lento) entonces no es recomendable probar con números muy grandes, y sostengo que con N=6000 ya existe un aproximación de valor del numero π.