UNIVERSIDAD NACIONAL DE LOJA

Área de la Energía, las Industrias y los Recursos Naturales No Renovables

CARRERA DE INGENIERÍA EN SISTEMAS

Nombre: Byron Montaño **Fecha:** 19-05-2019

Ciclo: 6 ”A”

CONGRUENCIAL MIXTO EN JAVA

CODIGO

LINK: https://github.com/byronmb/CONGRUENCIAL.git

El generador congruencial mixto se representa de la siguiente manera:

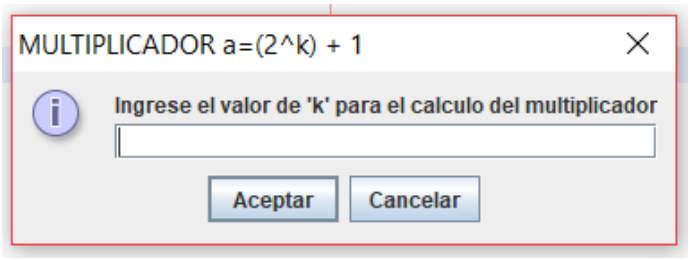
$$X_{n+1} = (aX_n + c) \bmod m$$

- En primer lugar, están definidas las variables principales que se utilizaran para poder realizar el cálculo del generador congruencial mixto. Estas son:
  X0 = la semilla
  a = el multiplicador
  c = constante aditiva
  m = el módulo

```
int a = 0;
int c = 0;
int x0 = 0;
int m = 0;
```

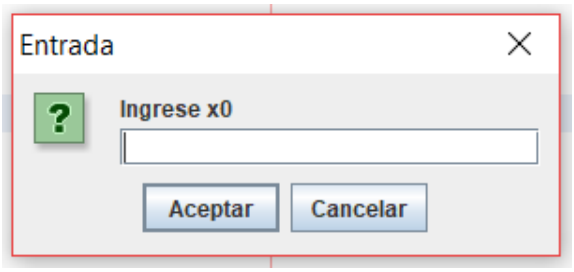
- Para el cálculo del multiplicador “*a*” se lo realiza leyendo una variable “*k*” del usuario basado en la regla que dice que “usualmente se selecciona ‘*a*’ como $2^k + 1$ cuando se trabaja en sistema binario”.
  Además, se controla con el bucle *while* para que el usuario solo pueda ingresar el valor de k mayor a 0.

```
//calculo del multiplicador "a"
while (k <= 0) {
    k = Integer.parseInt(JOptionPane.showInputDialog(null, "Ingrese el valor de 'k' para el calculo del multiplicador", "MULTIPLICADOR \na=(2^k) + 1", JOptionPane.INFORM
    if (k <= 0) {
        JOptionPane.showMessageDialog(null, "El valor de 'k' debe ser mayor a cero");
    }
    a = (int) Math.pow(2, k) + 1;
}
```



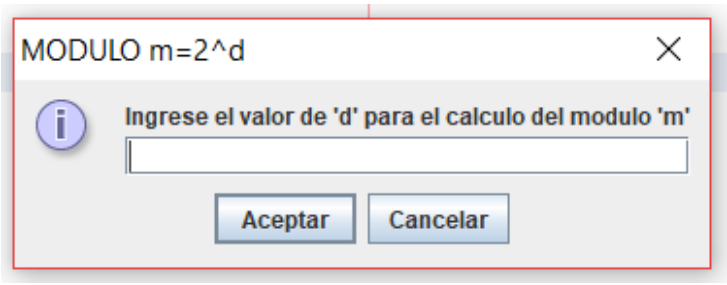
- Para el cálculo de la semilla "*x0*" se lo realiza simplemente leyendo una variable con el mismo nombre, y estableciendo de igual manera que el usuario solo pueda ingresar un valor mayor a cero.

```
//calculo de la semilla "X0"
while (x0 <= 0) {
    x0 = Integer.parseInt(JOptionPane.showInputDialog(null, "Ingrese x0"));
    if (x0 <= 0) {
        JOptionPane.showMessageDialog(null, "El valor de 'x0' debe ser mayor a cero");
    }
}
```



- Para el cálculo del módulo "*m*" se lo realiza leyendo una variable "*d*" del usuario basado en una de las opciones de selección la cual dice " seleccionar '*m'* como p^d ".
  Además, se controla que el valor del módulo sea mayor al multiplicador "a" y a la semilla "x0".

```
//calculo del modulo "m"
int d = 0;
while (m <= a || m <= x0) {
    d = Integer.parseInt(JOptionPane.showInputDialog(null, "Ingrese el valor de 'd' para el calculo del modulo 'm'", "MODULO \nm=2^d", JOptionPane.INF
    m = (int) Math.pow(2, d);
    if (m <= a || m <= c || m <= x0) {
        JOptionPane.showMessageDialog(null, "El valor de 'm' debe ser mayor a los valores de a, c y x0");
    }
}
```

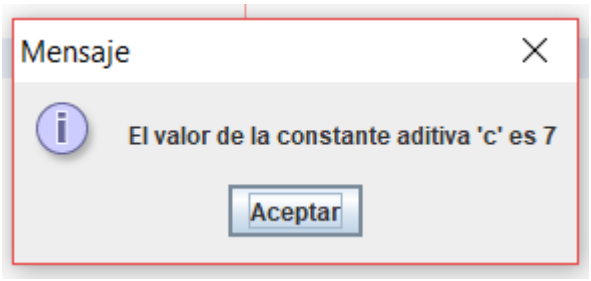


- Para el cálculo de la constante aditiva “c” se lo realiza basado en la regla que dice “el valor de c debe ser un valor impar y relativamente primo a m”.
  Para esto se utilizó un método para el cálculo del número primo más cercano, el cual recibirá como parámetro el valor de m.

```
//calculo de la constante aditiva "c"
c = nPrimo_Cercano(m);
JOptionPane.showMessageDialog(null, "El valor de la constante aditiva 'c' es " + c);
```

```
//Metodo para calcular el numero primo mas cercano
    public static int nPrimo_Cercano(int num) {
        int cont = 2;
        boolean n_primo = true;

        while ((n_primo) && (cont != num)) {
            if (num % cont == 0) {
                n_primo = false;
            }
            cont++;
        }
        if (n_primo) {
            return num;
        } else {
            return nPrimo_Cercano(num - 1);
        }
    }
```



- Para la generación de los numero pseudoaleatorios se la realizo con un ciclo repetitivo el cual se repetirá el valor de m veces, en caso de que la secuencia de los valores posibles se repitan, se controlan con una condición que compara el

primer valor de X_{n+1} con los valores siguientes , en caso de que estos se repitan, el ciclo termina.

Además, se realiza el cálculo de los números pseudoaleatorios mediante la fórmula establecida, y la semilla ira tomando cada valor de estos durante todo el ciclo repetitivo.

```
int n_pseudo = 0;
int cociente = 0;
System.out.println("n" + " | " + "xn" + " | " + a + "Xn+" + c + "/" + m + " | " + "Xn+1" + " | " + "Numeros Uniformes" + "\n");
int aux = x0;
for (int i = 1; i <= m; i++) {
    n_pseudo = ((a * x0) + c) % m; //(aXn+c) mod m
    cociente = ((a * x0) + c) / m; //operacion -> ((a * x0)+c) {divisor}
    System.out.println(i + " | " + x0 + "  |  " + cociente + "+" + n_pseudo + "/" + m + " |   " + n_pseudo + "  |    " + n_pseudo + "/" + m);
    x0 = n_pseudo;
    if (i > 1 && aux == n_pseudo) {
        break;
    }
}
```

PRUEBA DE PROMEDIOS

Para el cálculo de la prueba de promedios se lo realizó mediante un método el cual toma como parámetro el valor del arreglo de números pseudoaleatorios generados. Con este arreglo se puede determinar el tamaño del mismo para calcular la media de los valores del arreglo.

El valor del estadístico Zo se lo realiza basándose en la formula Zo = $\frac{(\bar{x} - 1/2)\sqrt{N}}{\sqrt{1/12}}$.

Luego se compara si el valor absoluto de Zo es menor con el valor de $Z_{\alpha/2}$, si se supone un nivel de significancia (**α**) de 5% entonces $Z_{\alpha/2}$ será igual a 1,96.

Si la comparación es verdadera No se puede rechazar la hipótesis de que los números pseudoaleatorios tienen un nivel esperado de aceptación de 0.5

```
public static void prueba_Promedios(ArrayList<Double> n_pseudoaleatorios) {
    System.out.println("\nPRUEBA DE PROMEDIOS");
    double suma = 0.00;
    int N = n_pseudoaleatorios.size();
    for (int i = 0; i < N; i++) {
        suma = suma + n_pseudoaleatorios.get(i);
    }
    double media = suma / N;
    //Zo=((media-0.5)*raiz N) / (raiz 1/12)
    double Z0 = ((media - 0.5) * Math.sqrt(N)) / Math.sqrt(0.083333);  //ESTADISTICO
    System.out.println("Media x --> " + media);
    System.out.println("Zo --> " + Z0);
    if (Math.abs(Z0) < 1.96) { //nivel se significancia del 95%
        System.out.println("NO se puede rechaza la hipotesis de que los numeros pseudoaleatorios tienen un nivel esperado de aceptacion de 0.5");
    } else {
        System.out.println("SE rechaza  la hipotesis de que los numeros pseudoaleatorios tienen un nivel esperado de aceptacion de 0.5");
    }
    System.out.println("-------------------------------------------------");
}
```

PRUEBA DE FRECUENCIAS

Para el cálculo de la prueba de frecuencias se lo realizó mediante un método el cual toma como parámetro el valor del arreglo de números pseudoaleatorios generados. Con este arreglo se puede determinar el tamaño del mismo para calcular la media de los valores del arreglo.

La prueba de frecuencias se basa en la formula $X_0^2 = \sum_{i=1}^{n} \frac{(FO_i - FE_i)^2}{FE_i}$ en donde:
FO es la frecuencia observada y
FE es la frecuencia esperada.
La frecuencia esperada se calcula mediante la fórmula N/n en la cual N es el total de números pseudoaleatorios y n es el número de intervalos que en esta ocasión tienen el valor de 5.
Se creó un arreglo que almacenara los valores que se tomaran en cuenta para calcular la frecuencia observada (1/n ,2/n,3/n …..).
Al determinar la frecuencia observada se procede a calcular el valor del estadístico de acuerdo a la fórmula $X_0^2 = \sum_{i=1}^{n} \frac{(FO_i - FE_i)^2}{FE_i}$.

El valor del estadístico X_0^2 se lo compara con el valor de chi cuadrado con (n-1) grados de libertad y un nivel de significancia **α.**
Si se supone un nivel de significancia de 5% --> 0.05 entonces $X^2_{\mathbf{0.05\,,4}}$ = 9,49.
Si el valor del estadístico X_0^2 < 9,49 entonces NO se puede rechazar la hipótesis de que los números pseudoaleatorios provienen de una distribución uniforme.

```java
public static void prueba_Frecuencias(ArrayList<Double> n_pseudoaleatorios) {
    System.out.println("\nPRUEBA DE FRECUENCIAS");
    int n = 5;   // # de subintervalos
    int N = n_pseudoaleatorios.size();
    int f_esperada = N / n;
    int f_observada[] = new int[n];
    double valores[] = new double[n]; //los valores de .2  .4 .....
    double estadistico = 0.00; //valor del estadistico X
    System.out.println("FE\tFO");
    for (int i = 0; i < n; i++) {
        int aux = 0;
        for (int j = 0; j < N; j++) {
            valores[i] = (double) (i + 1) / n; //.2 .4 ...
            if (i == 0) {
                if (n_pseudoaleatorios.get(j) < valores[i]) {
                    aux++;
                    f_observada[i] = aux;
                }
            } else {
                if ((n_pseudoaleatorios.get(j) < valores[i]) && (n_pseudoaleatorios.get(j) >= valores[i - 1])) {
                    aux++;
                    f_observada[i] = aux;
                }
            }
        }
        System.out.println(f_esperada + "\t" + f_observada[i] + "  ->  " + valores[i]);
        estadistico = estadistico + Math.pow((f_observada[i] - f_esperada), 2); // (F0-FE)^2
    }
    double X = (double) estadistico / f_esperada;  //SUM((F0-FE)^2)/FE

    System.out.println("n--> " + n);
    System.out.println("(Xo)^2 --> " + X);
    //α nivel de significancia   n-1 grados de libertad    NC=1-α
    //con un vlaor de  α de 0.05; y n=5 el valor de chi cuadrado seria --> X α,n-1 == 9,49
    double chi_cuadrado = 9.49;
    if (X < chi_cuadrado) {
        System.out.println("NO se puede rechaza la hipotesis de que los numeros pseudoaleatorios provienen de una distribucion uniforme");
    } else {
        System.out.println("SE rechazar la hipotesis de que los numeros pseudoaleatorios provienen de una distribucion uniforme");
    }
```

PRESENTACION DE LOS RESULTADOS

```
a-->5
x0-->4
m-->8
c-->7
n       xn      5Xn+7/8 Xn+1    Numeros Uniformes

1       4       3+3/8   3       3/8 ==> 0,37500
2       3       2+6/8   6       6/8 ==> 0,75000
3       6       4+5/8   5       5/8 ==> 0,62500
4       5       4+0/8   0       0/8 ==> 0,00000
5       0       0+7/8   7       7/8 ==> 0,87500
6       7       5+2/8   2       2/8 ==> 0,25000
7       2       2+1/8   1       1/8 ==> 0,12500
8       1       1+4/8   4       4/8 ==> 0,50000

PRUEBA DE PROMEDIOS
Media x --> 0.4375
Zo --> -0.6123736604443402
NO se puede rechaza la hipotesis de que los numeros pseudoaleatorios tienen un nivel esperado de aceptacion de 0.5
--------------------------------------------------

PRUEBA DE FRECUENCIAS
FE      FO
1       2  ->  0.2
1       2  ->  0.4
1       1  ->  0.6
1       2  ->  0.8
1       1  ->  1.0
n--> 5
(Xo)^2 --> 3.0
NO se puede rechaza la hipotesis de que los numeros pseudoaleatorios provienen de una distribucion uniforme
BUILD SUCCESSFUL (total time: 19 seconds)
```