Proyecto 2

Convertir a Real

Manuel Alejandro Hernández Peña

A01022089

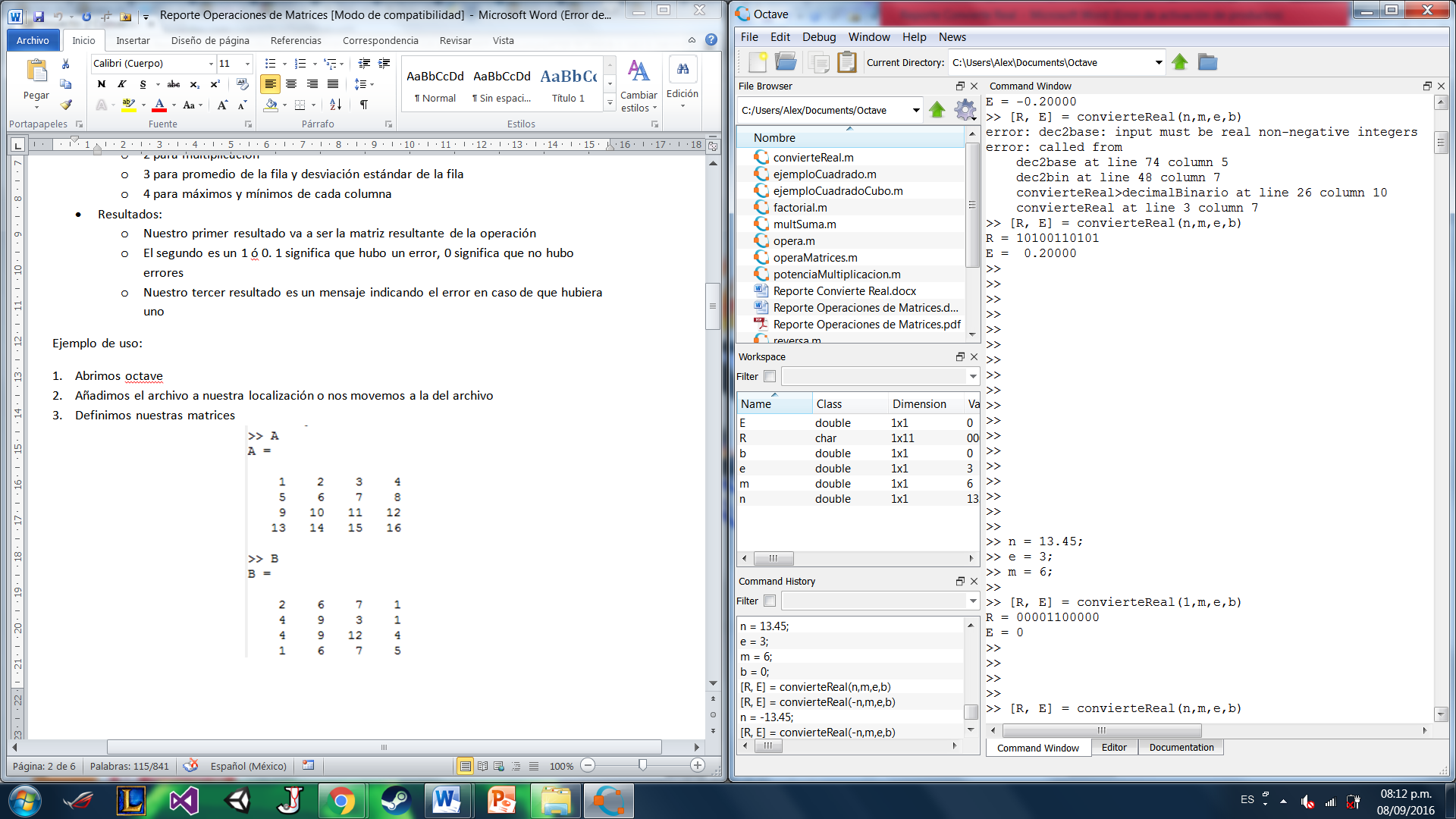
Manual de usuario

Uso:

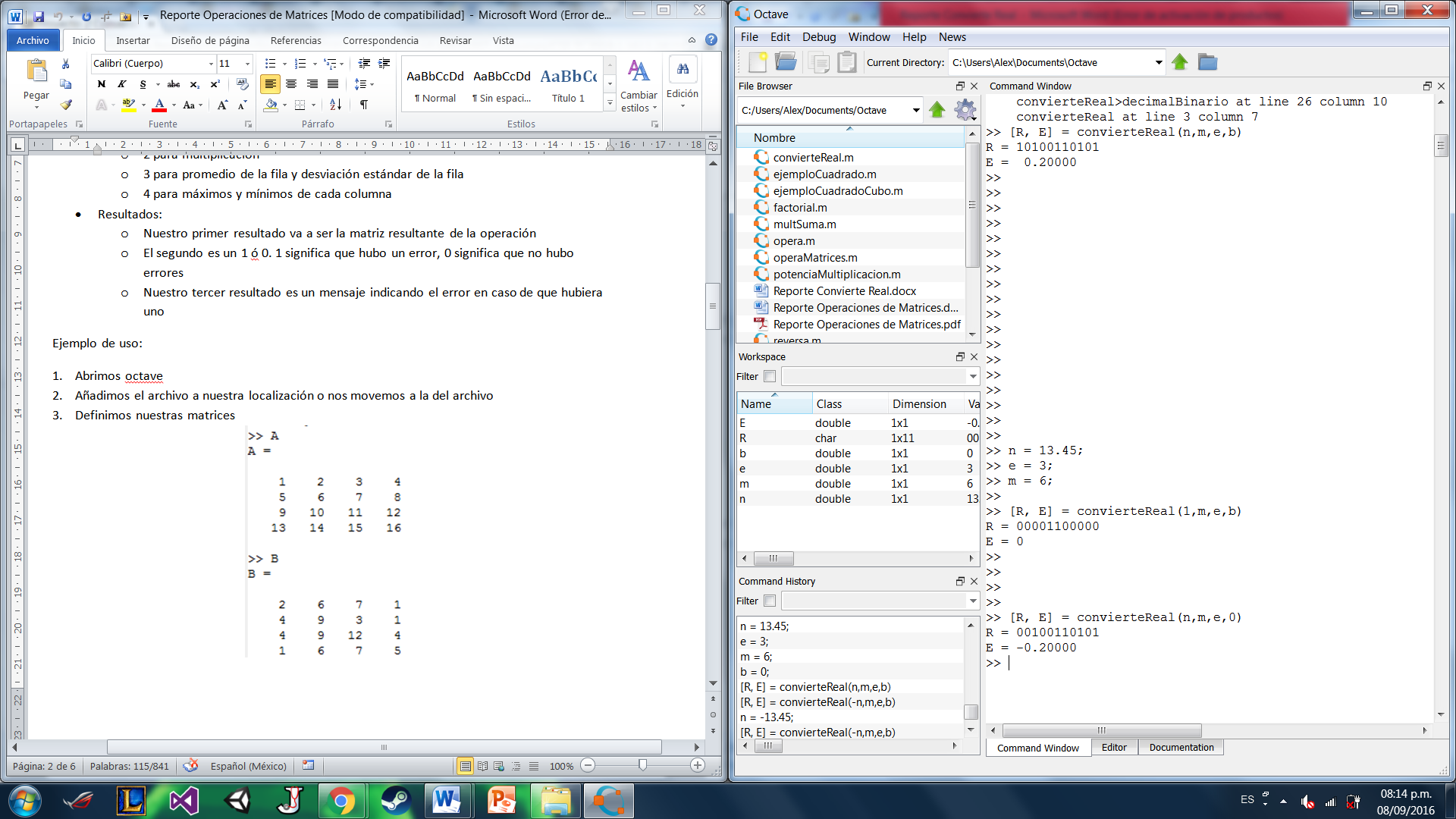
* Para llamar a la función correctamente necesita:
  + Un número que nos indique si va a ser conversión de binario a decimal o decimal a binario
  + Un número en caso de querer pasar a binario o un string en binario si se desea pasar a decimal. El valor en decimal de los números debe ser mayor a 1 o menor a -1 para que funcione correctamente.
  + Un número que indique los espacios que ocupa el exponente
  + Un número que indique los espacios que ocupa la mantisa
  + Dos variables donde se almacenara el resultado y el error en caso de existir
* Operaciones:
  + 0 para pasar de decimal a binario
  + 1 para pasar de binario a decimal
* Resultados:
  + Nuestro primer resultado, en caso de pasar de decimal a binario, será nuestro número en binario con formato punto flotante. En caso de pasar de binario a decimal el resultado será nuestro número en decimal
  + El segundo resultado, si es de decimal a binario, es el error que se produce de guardar el número en la computadora con respecto al original. En caso de ser de binario a decimal no regresa error.
  + Nuestro tercer resultado es un mensaje indicando el error en caso de que hubiera uno.

Ejemplo de uso

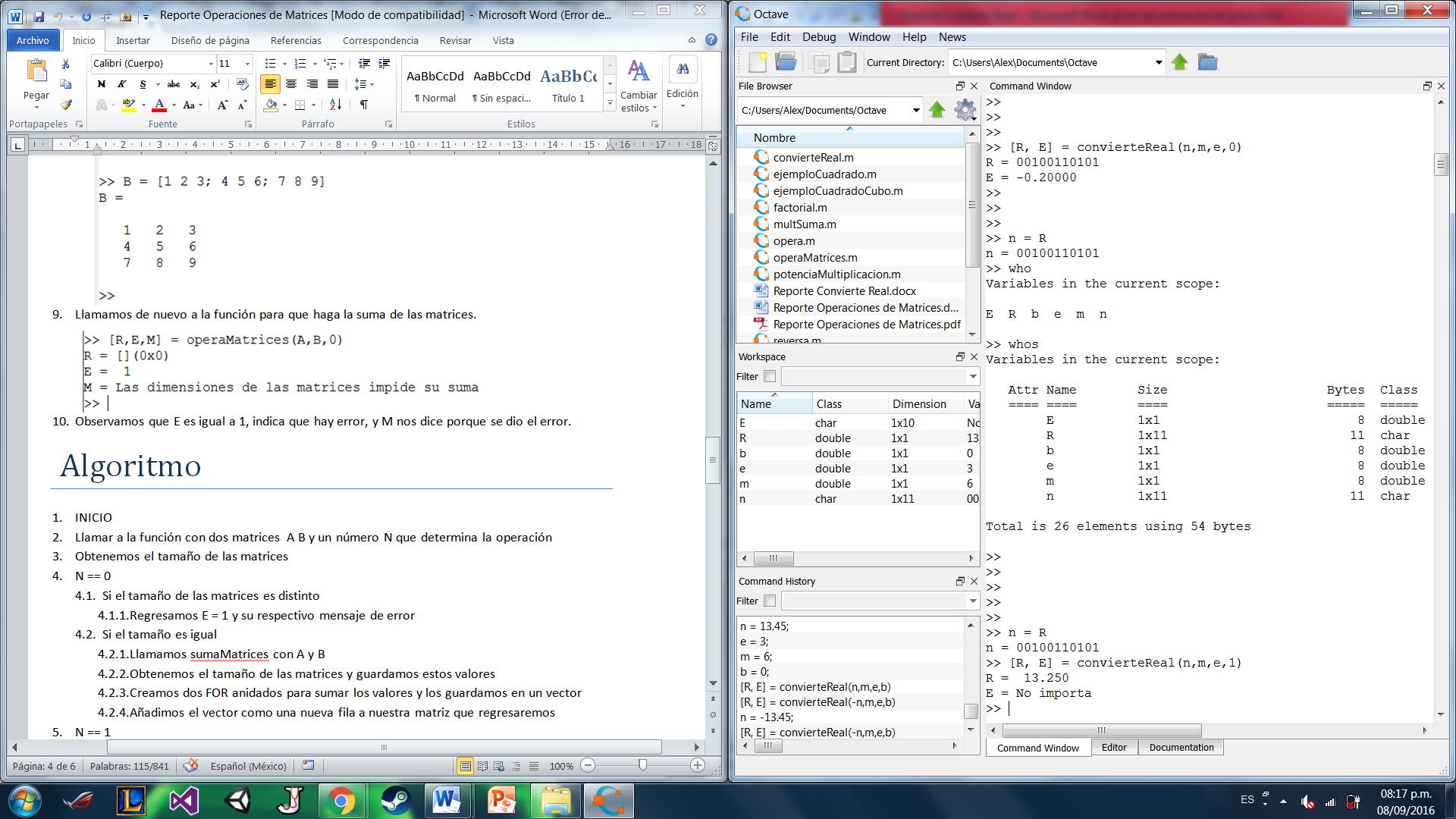
1. Abrimos octave
2. Añadimos el archivo a nuestra localización o nos movemos a la del archivo
3. Definimos nuestro número, el tamaño que ocuparemos para nuestro exponente y el tamaño de nuestra mantisa



1. Llamamos a nuestra función con 0 para pasar de decimal a binario en formato de punto flotante.



1. Ahora llamamos a la función pero con R y 1 para ver como guardo nuestro primer número. Aprovechamos que ya tenemos el tamaño de la mantisa y el exponente guardados en m y e.



1. Observamos que efectivamente el error de guardar 13.45 en la computadora es de -0.2.

Algoritmo

1. INICIO
2. La función recibe un numero n, el tamaño de la mantisa m, el tamaño del exponente e y un número que determina la operación b.
3. Si b es 0
   1. Checamos si el número es mayor a o igual a 0
      1. Si es el primer número de nuestro resultado es 0
      2. Si no lo es el primer número es 1
   2. Ponemos que el segundo número de nuestro resultado es 0
   3. Guardamos en una variable el número sin decimales
   4. Guardamos en una variable distinta solo los decimales del numero
   5. Copiamos el tamaño de la mantisa a otra variable
   6. Transformamos nuestro número sin decimales en binario
   7. Iniciamos una variable para guardar la parte decimal de nuestro binario
   8. Iniciamos una variable para recorrer el array de nuestros decimales
   9. Creamos un ciclo para convertir el decimal a binario con el tamaño de la mantisa
      1. Multiplicamos lo decimal \* 2
      2. Redondeamos para abajo el resultado de la multiplicación y lo convertimos a string para guardarlo en un array
      3. Si nuestro decimal es igual o mayor a 1 le restamos 1
   10. Checamos si nuestro número es mayor a 1
       1. Si lo es dividimos la parte entera del binario entre 10 hasta que no nos quede como entero
       2. Las veces que se repite este ciclo es el exponente y lo convertimos a binario
       3. Si no es mayo a cero significa que es negativo y el segundo valor de nuestro resultado lo cambiamos por 1
       4. Dividimos nuestro resultado por un 10 a la menos n hasta que sea mayor a 1
       5. Nuestro exponente es el número de veces que hacemos esto
       6. Convertimos nuestro exponente por -1 a binario
   11. Añadimos nuestro exponente en binario de derecha a izquierda en el tamaño de nuestro exponente dado por el usuario
   12. Juntamos nuestra parte entera y decimal del binario en un solo string para que quede normalizado
   13. Rellenamos nuestro resultado con los valores de izquierda a derecha de nuestro binario normalizado dependiendo de cuantos quepan en la mantisa.
   14. Regresamos nuestro resultado y convertimos nuestro resultado a número decimal.
   15. Le restamos al resultado de la conversión nuestro numero dado por el usuario y esto lo guardamos como nuestro error.
   16. Regresamos los dos resultados
4. En caso de que b no se a 1
   1. Separamos la parte del binario normalizado y el exponente
   2. Convertimos el exponente de binario a base 10
   3. Si nuestro segundo elemento de n es 1 pasamos nuestro exponente a negativo
   4. Si nuestro exponente es positivo
      1. Separamos nuestra parte entera del binario de la decimal
      2. Convertimos la parte entera del binario a base 10
      3. Convertimos la parte decimal del binario a base 10
      4. Los sumamos y lo guardamos en nuestra variable resultado
   5. Si el exponente es negativo
      1. Añadimos tantos 0 como el exponente nos indique
      2. Convertimos el binario a base 10 y lo guardamos en nuestra variable resultado
   6. Si el primer número del número que mando el usuario es 1 regresamos la variable resultado como negativa.
   7. El error lo mandamos con el mensaje “no importa”

Descripción Técnica

Tenemos en total 3 funciones. El main, decimalBinario que se encarga de pasar el numero base 10 a binario en formato punto flotante y binarioDecimal que se encarga de convertir de binario en formato punto flotante a base 10.

* En la función decimalBinario:
  + num = nuestro numero base 10 sin la parte decimal
  + dec = nuestro numero base 10 sin la parte entera
  + numBin = la parte entera en un string ya como binario
  + decBin = aquí guardamos la parte decimal del número base 10 en binario
  + magExp = la magnitud del exponente en base 10
  + binNorm = la parte decimal y entera del binaria ya normalizada
  + R = la variable que guarda el resultado
* En la función binarioDecimal
  + binNorm = la parte decimal y entera del binaria ya normalizada
  + exponente = nuestro exponente ya en base 10
  + binEnt = la parte entera del binario
  + binDec = la parte decimal del binario
  + decimal = nuestro número ya en base 10

# Referencias

Manual de referencia de Octave:

http://www.gnu.org/software/octave/doc/interpreter/index.html