SISTEMAS NUMÉRICOS: Introducción a la Informática

UTP | Pereira

GUILLERMO SIERRA

OCTUBRE DE 2020

2020

# CONTENIDO

[1 CONTENIDO 1](#_30j0zll)

[2 PRESENTACIÓN 2](#_3znysh7)

[3 CONVERSIÓN BASADA EN DIVISIONES SUCESIVAS 3](#_2et92p0)

[4 CONVERSIÓN EXTENDIDA 6](#_tyjcwt)

[5 POR CADA TEMA, UNA SECCIÓN SIMILAR A LAS ANTERIORES, TOMADAS DEL CLASSROOM 7](#_3dy6vkm)

[6 CONCLUSIONES 8](#_1t3h5sf)

[7 BIBLIOGRAFÍA 9](#_4d34og8)

# PRESENTACIÓN

La presente monografía describe la implementación de un conjunto de programas que le dan soporte a la teoría numérica básica de la materia INTRODUCCIÓN A LA INFORMÁTICA.

En los siguientes párrafos se presenta una descripción básica del significado de lo que es un sistema numérico, especialmente el sistema en base 2.

<UTILIZAR LAS PRESENTACIONES EN POWER POINT Y APUNTES EN EL CLASSROOM, Y EL MATERIAL ADICIONAL QUE CONSIDERE IMPORTANTE>

**AUTOR: Guillermo sierra**

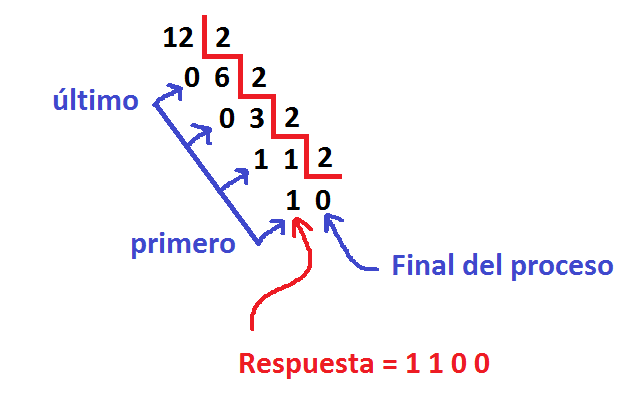
**1000064979**

**guillermo.sierra@utp.edu.co**

**https://github.com/Ravanox/informatica/upload/master/SEGUNDA%20PREVIA**

# CONVERSIÓN BASADA EN DIVISIONES SUCESIVAS

A continuación se presenta el algoritmo básico para la conversión numérica basada en divisiones sucesivas.

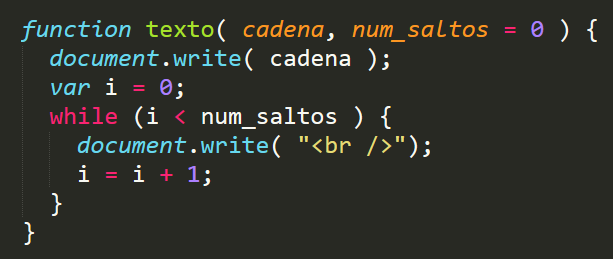


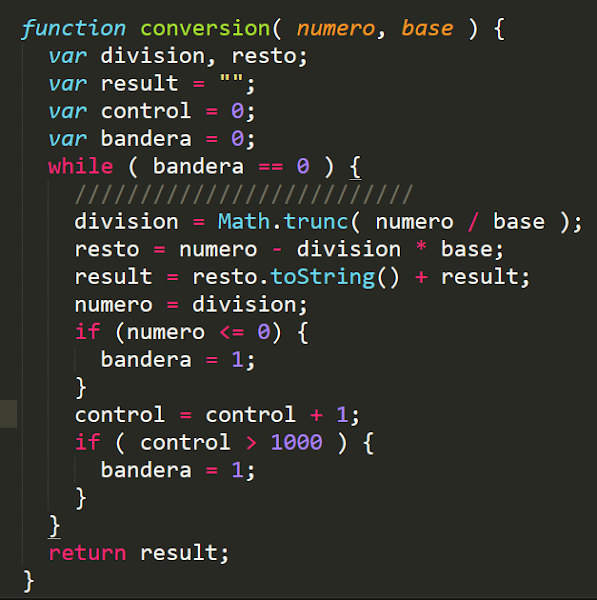
Como se ve en el diagrama, la conversión se realiza dividiendo el número a convertir entre la base seleccionada.

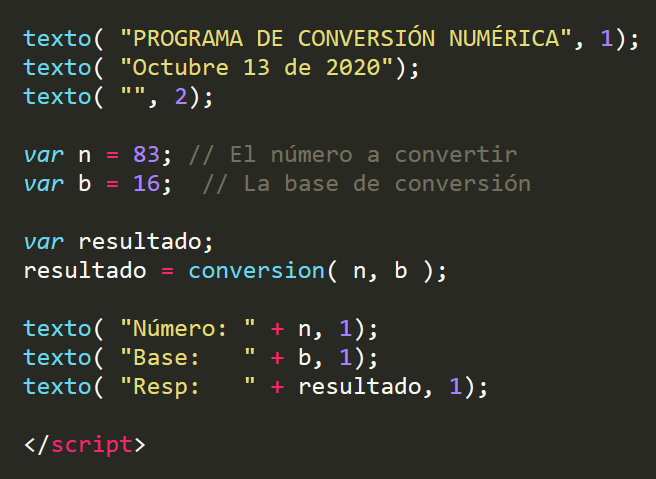
El resultado se obtiene con base en los residuos de las divisiones.

El proceso finaliza cuando se obtiene cero en el resultado de las divisiones.

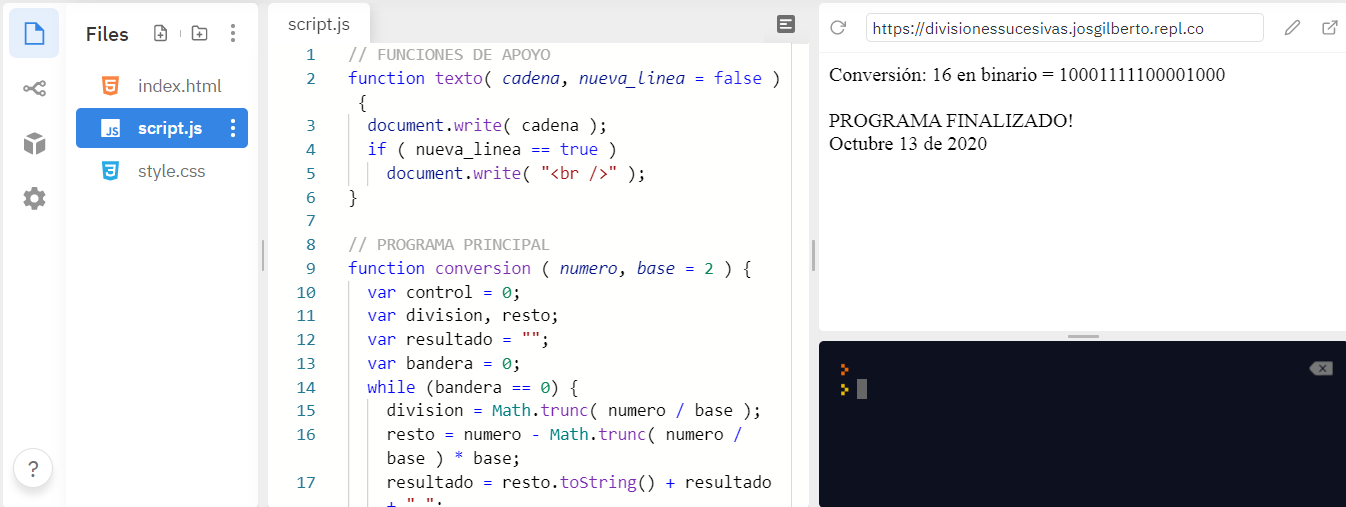
A continuación se presenta las imágenes de los códigos requeridos, para implementar el proceso mostrado en JavaScript. Cada imagen presenta una función distinta, o la ejecución final del programa. Se debe escribir en un solo archivo el código mostrado, y se sugiere un entorno como repl.it.







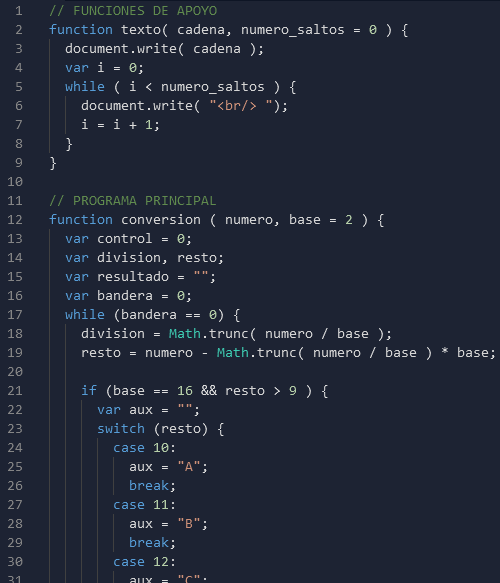
A continuación se muestra el programa en el entorno repl.it, con los datos de ejecución del programa.



Un punto importante de este código es como al final control evita que el programa se vuelva infinito o se haga excesivamente largo dándole un límite de 1000 repeticiones

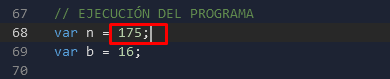
# CONVERSIÓN EXTENDIDA

A continuación presentamos el programa de conversión extendida, la cual se encarga de dar tratamiento a los números en base 16.

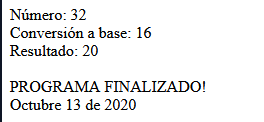


Pero en este caso si el residuo da más de 9 imprimirá ciertas letras que representen cuanto se pasó para que no se haga muy largo ni rompa el programa.

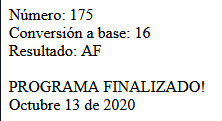
Por ejemplo si ingresamos el número 32 en la parte señalada



nos debe dar el número 20:



Y si ingresamos el 175 nos debe dar AF



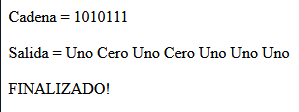
# VISTO EN CLASE

**Binarios**

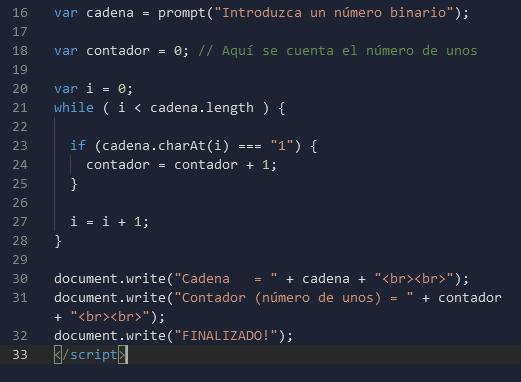
Código 1:



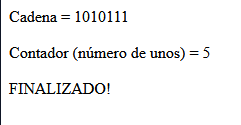
Este código primeramente lo que hace es que con prompt pregunta un número binario y el programa lee el número que el usuario ingrese después dentro de los if si un número en la cadena de números se lee como un 0 se convertirá en la palabra “Cero” y si es 1 se convertirá en la palabra “Uno” y así al final escribe la misma cadena que escribimos y el resultado escrito:



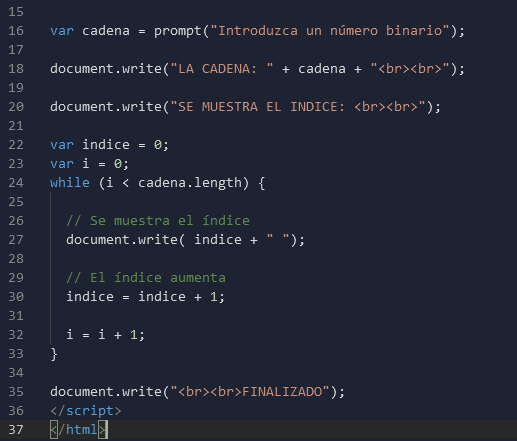
Código 2:



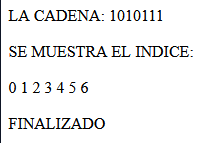
Aquí mismamente como empieza el anterior lee un número binario pero en este caso se cuenta el número de unos(1) que se encuentre en el número que se ingresó e imprime eso junto con la cadena ingresada anteriormente



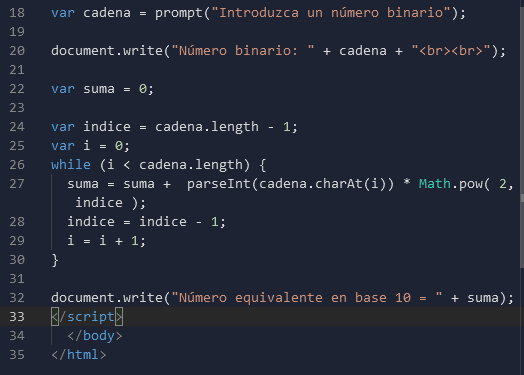
Código 3:



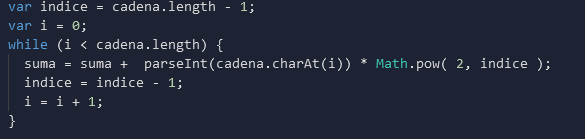
En este tercer caso lo que hace el código con el número binario que le damos es escribir desde 0 la cantidad de números que hay en ese digito binario.



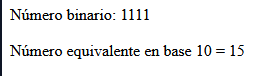
Código 4:



Igual que la anterior está vez lee u número binario pero en este caso devuelve la cadena original y el número binario ahora traducido en número decimal esto gracias al proceso matemático representado en esta parte del código:

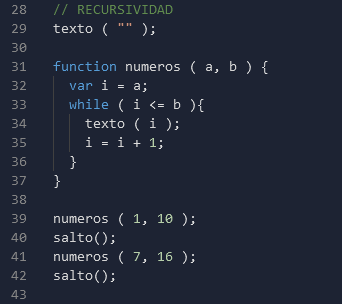


E imprime:



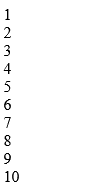
**Recursividad 1**

En este programa vemos diferentes formas de usar recursividad para acortar procesos o generarlos

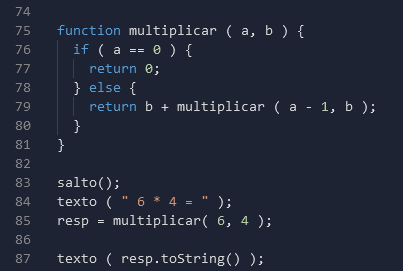


En esta parte podemos ver como se crea la función “números” en el orden en que va el código se empieza desde casi el final en la línea 39 donde define “a” como 1 y “b” como 10 proseguimos en la línea 31 donde como en la 39 llamo a esta función y la definió como dijimos antes se ejecuta.

En la línea 32 define a “i” como “a” por lo tal i ahora vale 1, siguiente a eso le dice que mientras que “i” sea menor o igual a “b” imprima “i” y sume 1 a “i” así hasta que i se convierte en una suma menor o igual a b dando como resultado:



En las siguientes partes del código hace un poco más de lo mismo hasta que llega a esta parte



En la que se crea un proceso de multiplicación con recursividad el cual nos da como resultado:



# CONCLUSIONES

El desarrollo de las temáticas elaboradas en clase utilizando el lenguaje JavaScript prueba ser un mecanismo de gran valor para el aprendizaje de los conceptos básicos de la materia.

Trabaja el lenguaje pero por sobre todo y más importante la lógica en la programación al poder entender un lenguaje y que es lo que hace.

# BIBLIOGRAFÍA

<https://repl.it>

<OTROS ENLACES ADICIONALES>