**Escuela Superior Politécnica del Litoral**

Organización de Computadores

Paralelo Nº1

Proyecto Primer Parcial

Calculadora Decimal-Binaria-Hexadecimal

Integrantes:

Alex Macas

Ricardo Serrano

Guayaquil-Ecuador

II Término – 2018

**Especificaciones Código C**

La implementación en C del programa “Calculadora Decimal-Binaria-Hexadecimal” se realizó en una máquina con sistema operativo Ubuntu 18.04 LTS Bionic Beaver dado que por defecto, Linux implementa C y desde su terminal se puede acceder a las funcionalidades del compilador gcc para, valga la redundancia, compilar el código en C y ejecutarlo.

El código consta de 3 partes:

1. En la carpeta ***src*** se encuentran los archivos .c con las implementaciones de los métodos necesarios para que el programa funcione de manera correcta.

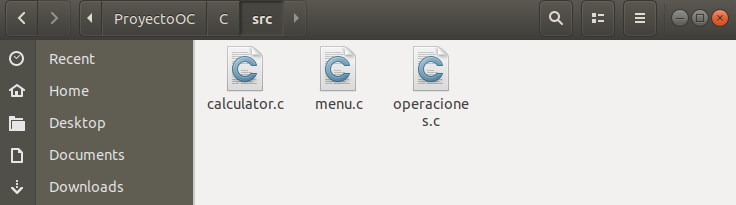


Imagen 1. Archivos calculator.c menu.c y operaciones.c

1. En la carpeta ***include*** se encuentran los archivos .h con los prototipos de las funciones que se utilizaron en los archivos .c además de los #includes pertinentes.

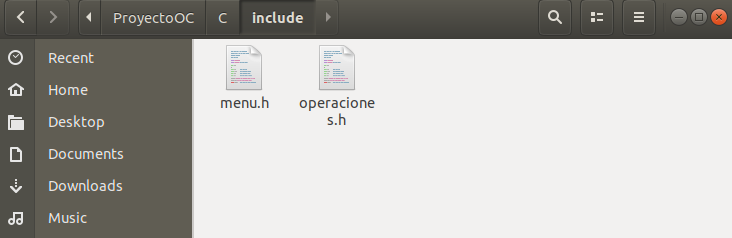


Imagen 2. Archivos menú.h y operaciones.h

1. El archivo ***Makefile***, permite compilar el proyecto con simplemente escribir *make* desde la terminal de Ubuntu (para este caso).

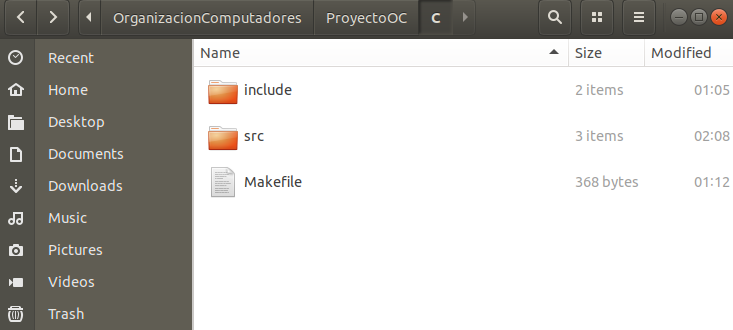


Imagen 3. Vista General del proyecto en C

Cada archivo .c y .h está debidamente comentado, explicando de manera clara y directa la finalidad de cada función.

**Funcionamiento del Programa en C**

Partimos del directorio principal de nuestro proyecto en C

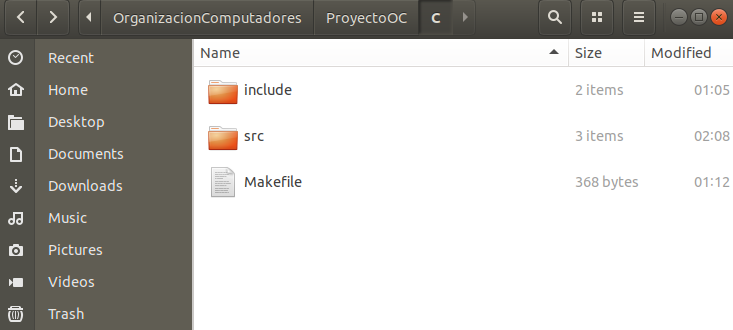


Imagen 4. Proyecto en C

Damos clic derecho en cualquier parte de la ventana de Imagen 4 y escogemos ***Abrir Terminal***

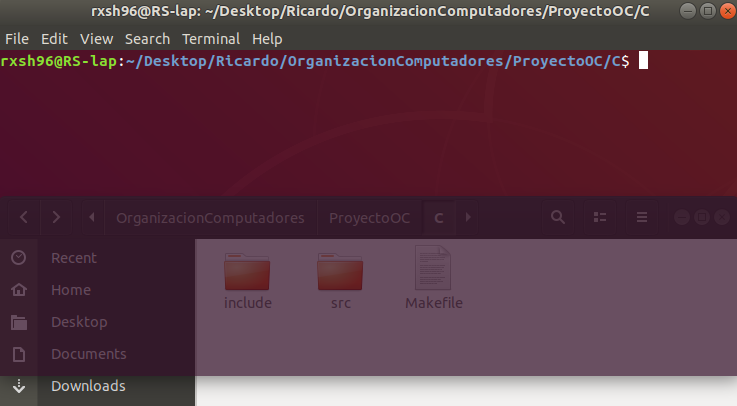


Imagen 5. Terminal de Ubuntu

El siguiente paso es escribir el comando ***make*** en la terminal para compilar los archivos .c y ejecutar el programa.

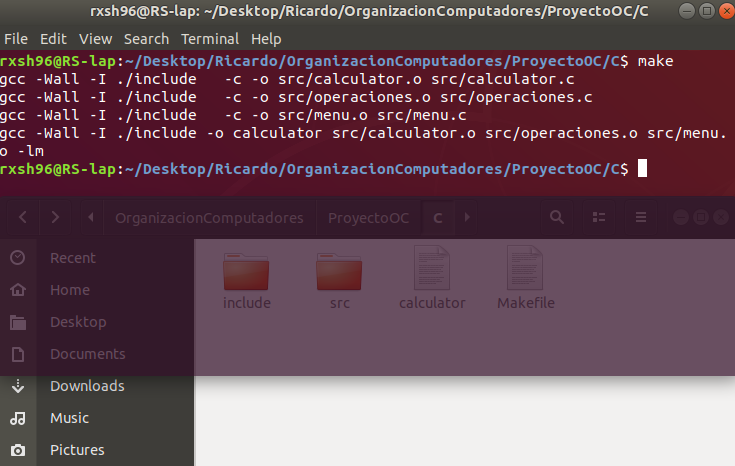


Imagen 6. Proyecto compilado con make

Si nos dirigimos a la carpeta src, se pueden observar archivos .o, resultado de la compilación.

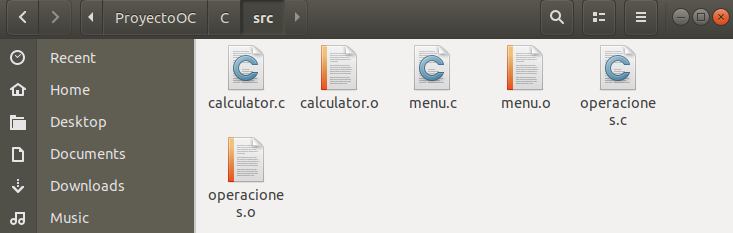


Imagen 7. Archivos conocidos como Object Files

Se observa que, además, en Imagen 6 se crea un nuevo archivo ***calculator***, este corresponde al ejecutable del programa. Para ejecutarlo, se escribe desde la terminal **./** seguido del nombre del archivo ejecutable.

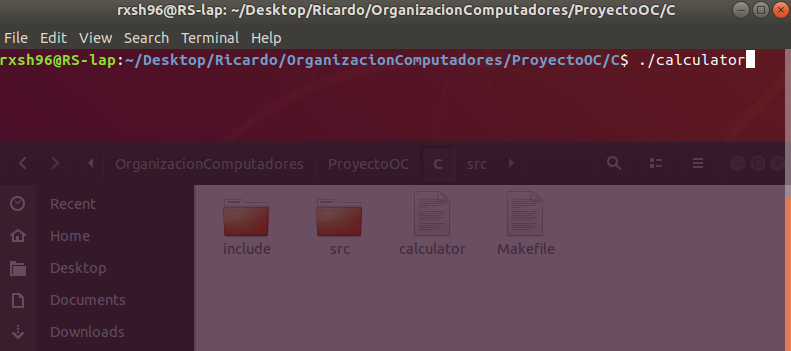


Imagen 8. Ejecutando calculator

Aparece un sencillo menú de opciones que permite escoger el tipo de conversión que se quiere realizar.

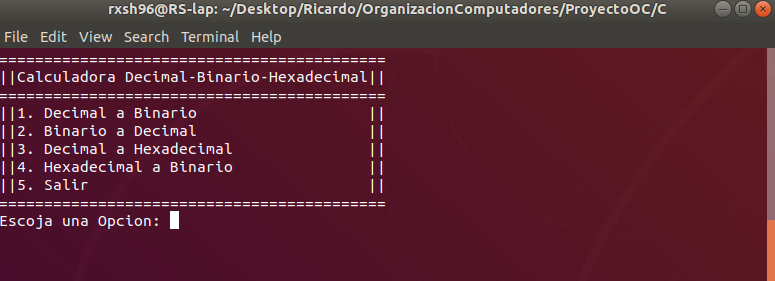


Imagen 9. Menú Principal

Para el caso de la Opción 1. Decimal a Binario, el programa nos pide el ingreso de un número decimal para calcular su respectivo número binario y mostrarlo por consola (*para efectos prácticos, se trabajará con el mismo número decimal, binario y hexadecimal para todas las opciones).*

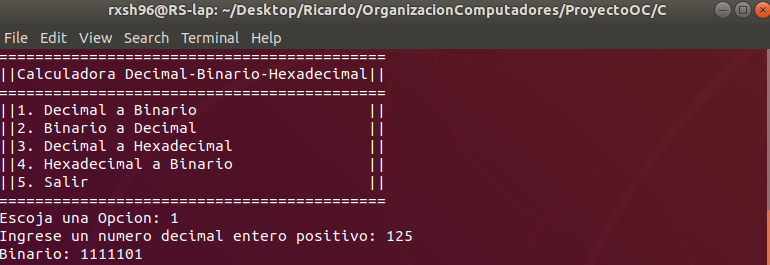


Imagen 10. Conversión decimal a binario

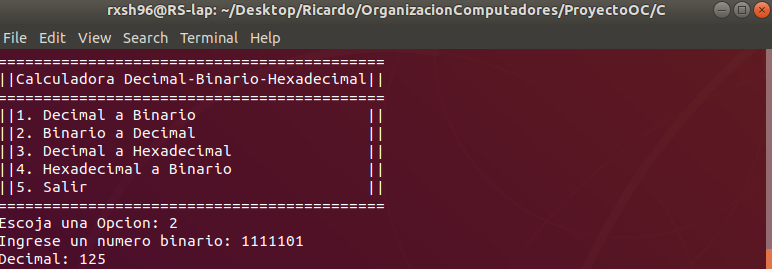


Imagen 11. Conversión binaria a decimal

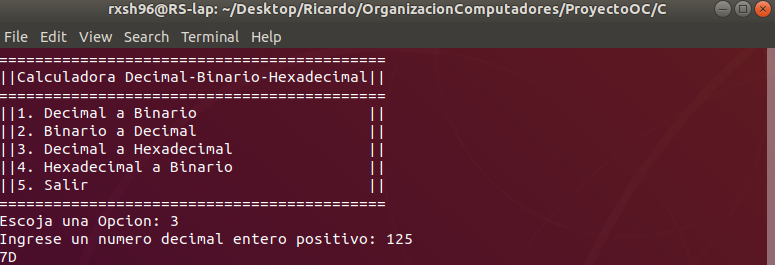


Imagen 12. Conversión decimal a hexadecimal

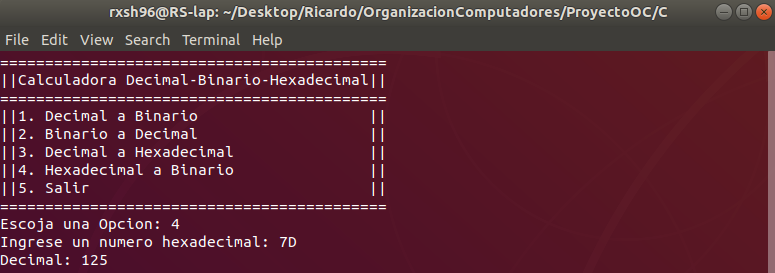


Imagen 13. Conversión hexadecimal a decimal

Se puede comprobar que para el caso de decimal a binario y de binario a decimal los resultados son correctos y de igual manera para las opciones 3 y 4.

Finalmente, al momento que hayamos terminado nuestras conversiones numéricas, podemos salir del programa de una manera correcta con la opción 5.

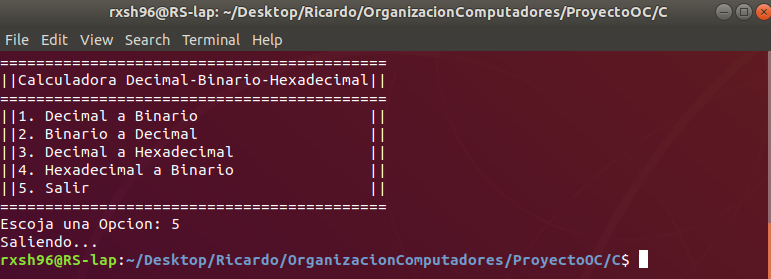


Imagen 14. Saliendo del programa

**Especificaciones del Programa en Assembler**

La implementación en Assembler del programa “Calculadora Decimal-Binaria-Hexadecimal” se realizó en una máquina con sistema operativo Windows 10 Home. Para la implementación de las instrucciones en código en assembler se utilizó el software MARS 4.5. Dicho software permite programar en una interfaz adecuada a estas instrucciones, además, permite ensamblar el código y probar su ejecución.

Para la implementación de “Calculadora Decimal-Binaria-Hexadecimal” se lo realizó en dos archivos .asm. El primero: ***calculator\_binary-decimal.asm*** corresponde a todas las operaciones e instrucciones para llevar un número decimal a binario y viceversa. El segundo: ***calculator\_hexa-decimal.asm*** contiene las instrucciones e instrucciones para llevar un número decimal a hexadecimal y viceversa.

Para ambos casos, se ha tratado de modularizar el código dividiéndolo en varias secciones.

Se observa la parte del menú con sus respectivos mensajes para mostrar por pantalla, bloque en el cual se llaman a las funciones principales, y funciones recursivas que hacen los cálculos correspondientes.

Una observación importante que aclarar para el archivo ***calculator\_binary-decimal.asm*** cuando se selecciona la opción 1: Convertir decimal a binario.

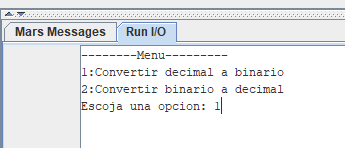


Imagen . Menú en Assembler

El programa pide el ingreso de un número decimal para hacer su conversión a binario y mostrarlo por pantalla. El resultado que se muestra es el correcto leyéndolo de derecha a izquierda.

El equivalente del 6 en binario es 110, sin embargo, el programa muestra 011. Este es un problema secundario muy importante mencionar.

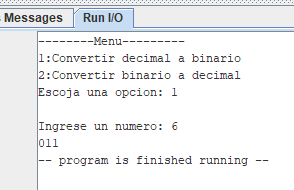


Imagen . Resultados en Orden Invertido

**Funcionamiento del Programa en Assembler**

Partimos del directorio principal de nuestro proyecto y entramos a la carpeta Assembler.

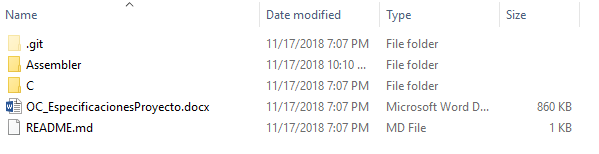


Imagen . Directorio Principal del Proyecto

Una vez dentro de la carpeta observamos dos archivos .asm y un archivo .jar

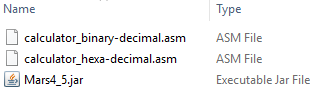


Imagen . Directorio Assembler

Nos ubicamos en la barra de direcciones, escribimos cmdy damos enter.

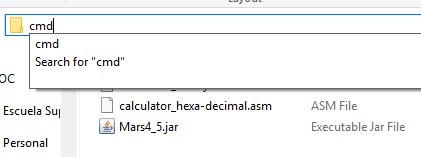


Imagen . Abriendo CMD desde directorio Assembler

Se nos abrirá el CMD de Windows donde escribiremos el siguiente comando: java -jar Mars4\_5.jar

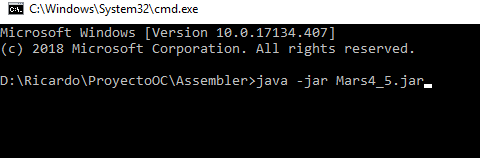


Imagen . Iniciando Mars

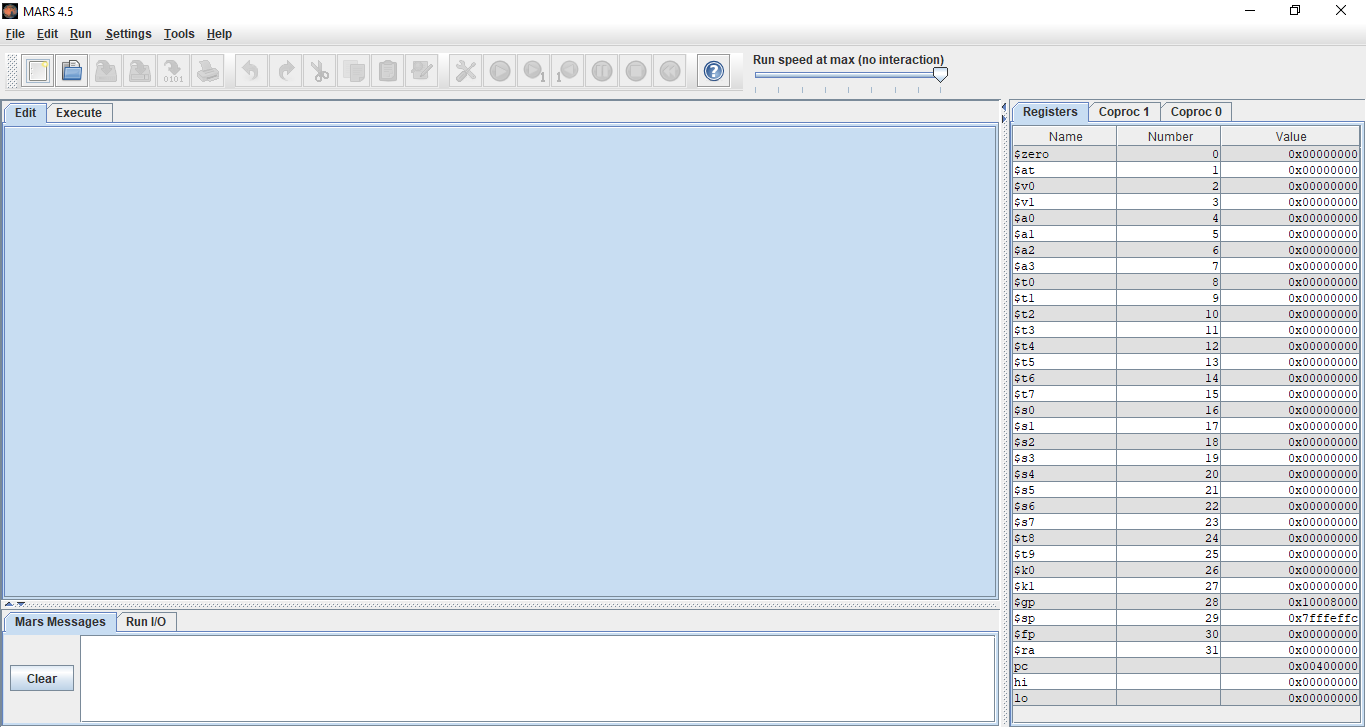


Imagen . Pantalla Principal Mars

Una vez iniciado el programa, se abre el archivo .asm que queramos ensamblar y ejecutar. Para esto damos clic en File -> Open -> Buscamos el archivo a través de los directorios y abrimos.

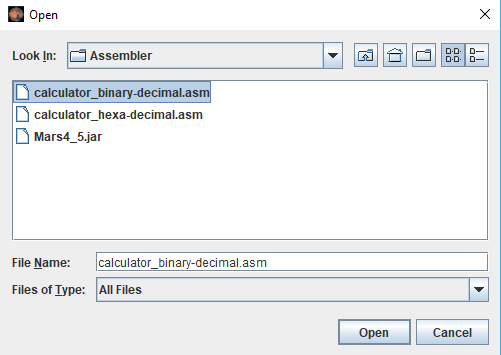


Imagen . Abriendo el archivo .asm

Lo siguiente es Ensamblar el código, para esto, damos clic en Run -> Assemble. O de manera más directa, presionamos F3.

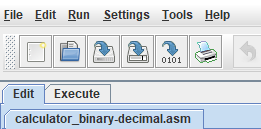


Imagen . Ensamblando

El programa nos redirige a la pestaña Execute, donde encontraremos todas las direcciones, instrucciones, variables y registros que utilizamos.

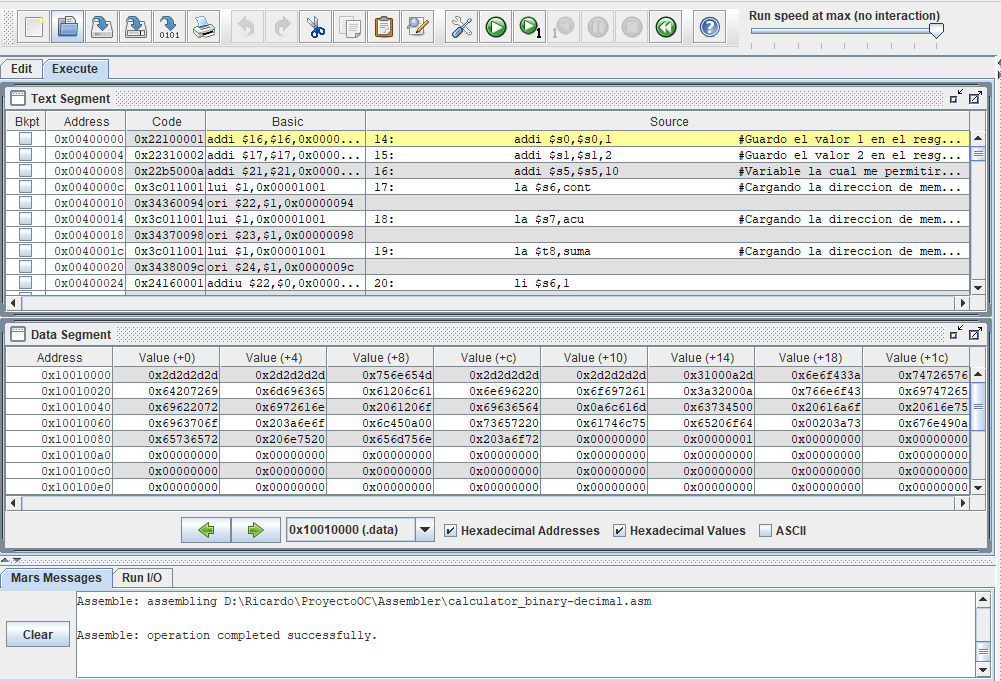


Imagen . Información del ensamblado

Para correr el programa, damos clic en el botón verde de Play más grande.



Imagen . Ejecutando el programa

Se visualiza la ejecución del programa y nos aparece el menú principal, en el cual al elegir cualquiera de las opciones, nos mostrará los resultados respectivos.

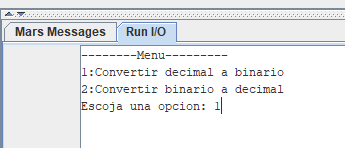


Imagen . Opción 1 seleccionada

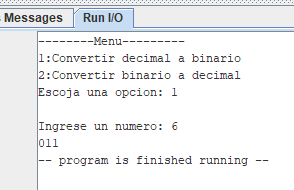


Imagen . Resultados en Orden Invertido

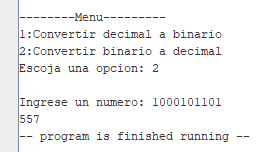


Imagen . Convesión Binario a Decimal

El programa realiza la conversión de binario a decimal mientras el número de dígitos binarios sea menor a 11. Caso contrario, se produce un runtime exception.

Para probar el archivo ***calculator\_hexa-decimal.asm*** se realizan los mismos pasos mencionados para ***calculator\_binary-decimal.asm***.

# Referencias

|  |  |
| --- | --- |
| [1] | Missouri State University, "courses.missouristate.edu," 2014. [Online]. Available: http://courses.missouristate.edu/KenVollmar/mars/download.htm. |
| [2] | "stackoverflow.com," [Online]. Available: https://stackoverflow.com/questions/40643473/whats-the-best-way-to-create-a-menu-in-mips-assembly?rq=1. [Accessed November 2018]. |
| [3] | "stackoverflow.com," [Online]. Available: https://stackoverflow.com/questions/23282527/convert-binary-to-decimal-in-mips-assembly-mars. [Accessed November 2018]. |
| [4] | "experts-exchange.com," [Online]. Available: https://www.experts-exchange.com/questions/21590524/How-to-convert-decimal-to-hex-in-MIPS.html. [Accessed November 2018]. |
| [5] | "cs.colby.edu," [Online]. Available: http://www.cs.colby.edu/maxwell/courses/tutorials/maketutor/. [Accessed November 2018]. |
| [6] | "stackoverflow.com," [Online]. Available: https://stackoverflow.com/questions/13478035/decimal-to-binary-conversion-in-mips-assembly. [Accessed November 2018]. |