Instituto Tecnológico de Costa Rica

Escuela de computación, Campus Tecnológico Local San José

Arquitectura de Computadoras
Grupo: 40

Profesor: Ing. Eduardo A. Canessa Montero, M.Sc.

Tarea Programada #2: Convertidor Malespín/español, español/Malespín.

Estudiantes:
Julián Rodríguez Sarmiento, 2019047635
Ian Murillo Campos, 2020148871

Grupo 4

Entrega: 19 de Junio, 1 Semestre 2023

Introducción:

Se tiene como objetivo de esta segunda tarea programada implementar un convertidor de español a Malespín y viceversa en el lenguaje GAS para ARM con ISA AA32. El programa deberá reconocer las letras que deben ser modificadas y por las cuales se deben sustituir para obtener en un archivo el texto modificado traducido a español o a Malespín.

Además se desplegarán como salidas algunos valores de interés del texto como lo son los totales de palabras y letras, los totales de palabras y letras modificadas y el porcentaje de palabras y letras modificadas.

El programa se realizará en la plataforma Raspberry Pi OS en su versión de 32 bits usando el lenguaje de bajo nivel GAS para ARM. Para entender mejor el programa y su finalidad a continuación se dará una breve explicación de la historia del código Malespín y sus reglas.

Según (Víquez, 2011) el código Malespín se creó y se desarrolló en Centroamérica en el siglo XIX cuando había guerras civiles. Su nombre proviene del general salvadoreño Francisco Malespín, la persona detrás de la creación del código, se utilizó para cambiar las palabras y así confundir al enemigo. Las reglas para formar palabras en malespín son sencillas, solo se deben sustituir ciertas letras las cuales son la “a” por la “e”, la “i” por la “o”, la “b” por la “t”, la “f” por la “g”, la “p” por la “m” y viceversa.

Diseño del programa:

El programa se diseñó pensando en bloques que desempeñan tareas específicas que juntos cumplen en totalidad la funcionalidad esperada.

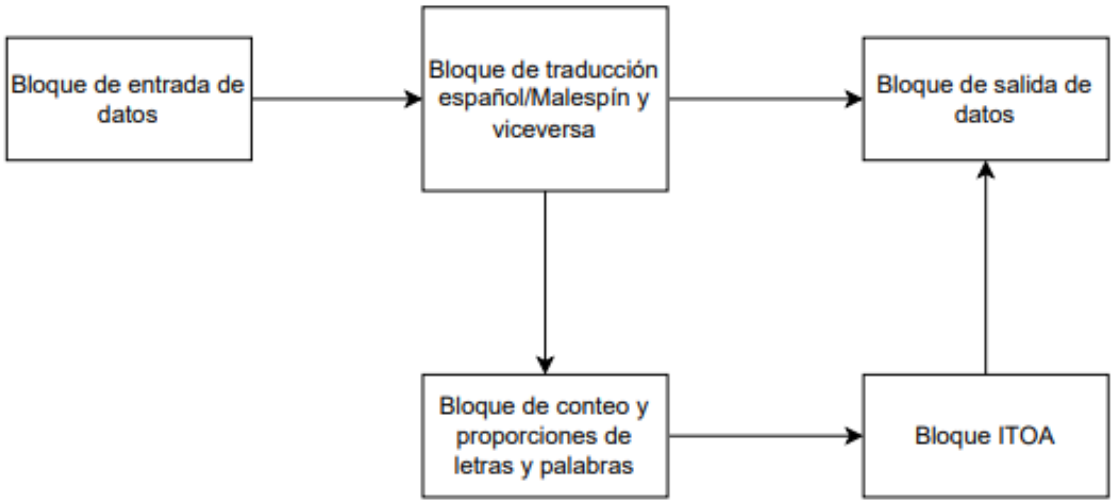


Figura 1. Diagrama de bloques.

Descripción de los bloques:

Bloque de entrada de datos:
Se encuentran las operaciones de entrada, específicamente el manejo de archivos en su modo de leer, leer desde consola el texto, las opciones de escogencia del usuario e imprimir mensajes para comunicar el programa con el usuario.

Bloque de traducción español/Malespín y viceversa:
Se encarga de traducir el texto según el formato solicitado y siguiendo las reglas del código Malespín.

Bloque de conteo y proporciones de letras y palabras:
Maneja los totales de palabras y letras leídas en el programa, los totales de letras y palabras modificadas, además procesa los porcentajes entre las modificadas y las no modificadas.

Bloque ITOA:
Transforma los valores numéricos de todos los totales y los dos porcentajes en texto para poder ser desplegados en pantalla.

Bloque de salida de datos:
Se encuentran las operaciones de salida, en este caso la salida en archivo y la salida en pantalla, además de carteles para dar significado al usuario de lo que está viendo.

Descripción técnica:

Diagrama de flujo

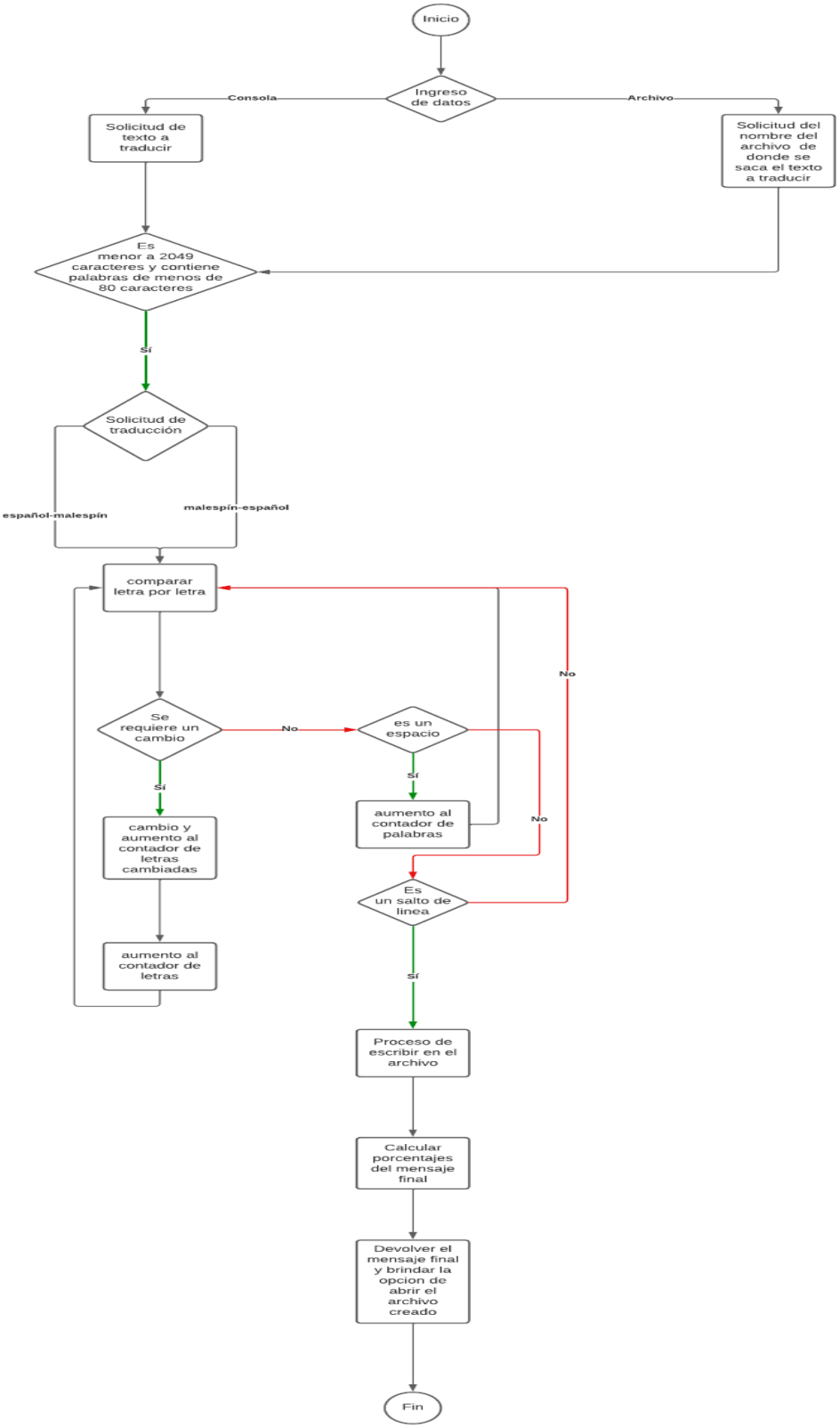


Figura2. Diagrama de flujo.

Procedimientos en la solución:
Módulo entrada de datos:
Pregunta al usuario por medio de carteles como desea introducir el texto(Archivo/consola) y guarda en una variable el texto ingresado y verifica que no sobrepase el límite de establecido. También pregunta el formato de la traducción de español a Malespín o de Malespín y guarda la opción en una variable.

Módulo de traducción español/Malespín y viceversa:
Utiliza un ciclo que recorre el texto letra por letra consultando por las que deben ser modificadas y de paso se aprovecha para llevar los contadores de letras y palabras. Identifica las palabras por medio de los espacios en blanco. Sale del ciclo cuando encuentra el caracter de salto de línea.

Módulo de conteo y proporciones de letras y palabras:
Prepara los totales para ser enviados al módulo ITOA y que salgan con su respectivo mensaje en pantalla. Opera los totales para obtener ambos porcentajes y los prepara para el ITOA.

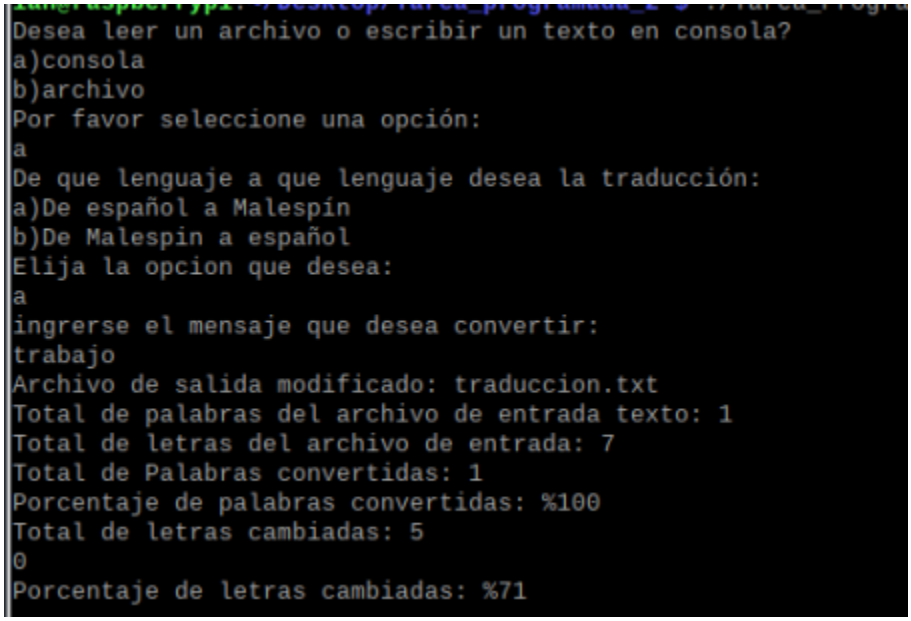
Módulo ITOA:
Utiliza las potencias de diez para sacar los dígitos del más significativo al menos significativo y así obtener el el valor ascii del dígito, además que el número queda ordenado, trabaja con un ciclo que empieza en la potencia nueve y termina cuando llega a la primera.

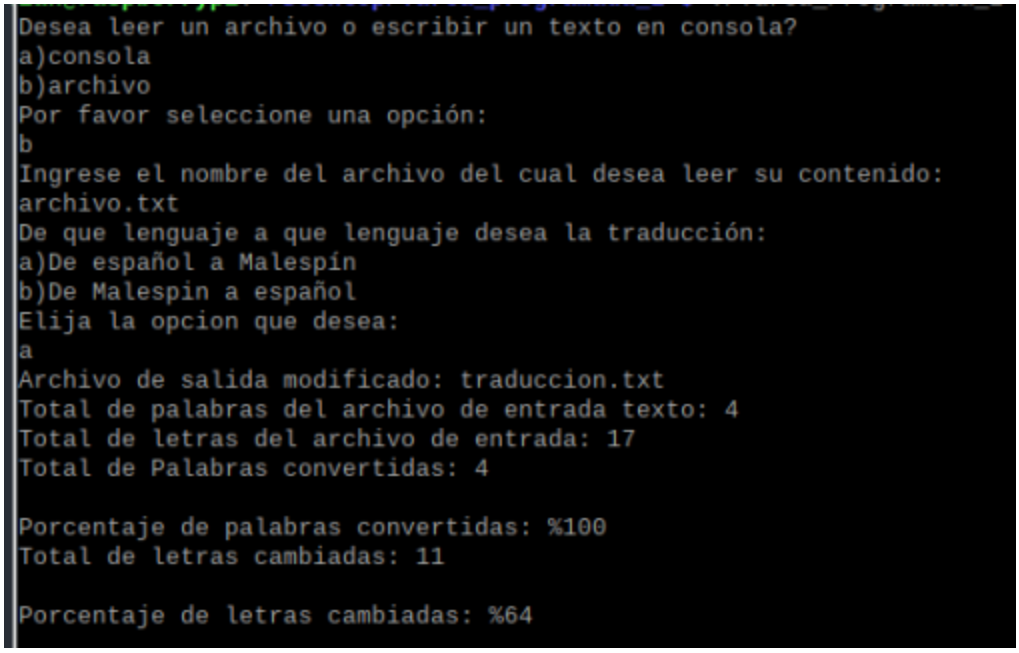
Módulo de salida de datos:
Crea y escribe en un archivo de texto la traducción realizada, además imprime los totales, porcentajes y el nombre del archivo que contiene el texto traducido. Contiene los mensajes de error.

Manual de usuario:
Para correr el programa en el Raspberry Pi 3 cos sistema Raspbian de 32 bits es necesario el uso de un “Makefile” el cual viene en el archivo comprimido del proyecto, para correr el programa se debe abrir un terminal en la ubicación donde esté el archivo “.s” y el “Makefile” en la terminal se deberá escribir la palabra Make, esto creará los archivos objeto y el ejecutable del programa, posteriormente se deberá escribir un “./” seguido del nombre del ejecutable(“./Tarea_Programada_2”) en la terminal para correr el ejecutable e iniciar el programa.
El programa inicia preguntando al usuario desde consola si desea que el texto a traducir provenga de un archivo de texto o brindado en la misma consola, la opción se

elige presionando la letra “a” o “b”, en caso de que se elija la opción de brindar el texto mediante consola, se le pedirá al usuario que indique que tipo de traducción desea (Malespín a español o viceversa) para luego solicitar que brinde el texto a traducir, al terminar, el programa traducirá el texto ingresado al idioma solicitado y brindará una serie de datos relacionados a los cambios realizados al texto y procederá a guardar la traducción en un archivo .txt llamado “tradcción.txt”.

De la misma forma, si se solicita que se traduzca el texto de un archivo brindado, el programa le preguntará al usuario el nombre de dicho archivo, debe ingresarse utilizando el formato “nombre.txt”, luego procederá a preguntar que tipo de traducción desea (Malespín a español o viceversa) y terminará de la misma forma dicha anteriormente, brindando datos relacionados a las conversiones realizadas y creando el archivo “traducción.txt” donde se hallará el texto traducido (nota, el archivo de entrada sólo deberá contener texto sin salto de línea, debido a que este último se toma como el final del texto brindado).





Conclusiones:

- Como primera conclusión se tiene que una buena forma de elaborar el código es es por bloques funcionales manteniendo en cierta parte un código simple y fácil de entender, primordial para los lenguajes de bajo nivel porque el código se puede volver sumamente pesado además de la posible falta de registros sin usar.
- En tareas establecidas las cuales a grandes rasgos son, el manejo de entradas desde archivo y consola, los límites, la traducción de español a Malespín y viceversa, los totales y porcentajes y la salida en archivo en la opinión de los autores se completan satisfactoriamente cada una de ellas.
- El manejo de cadenas de caracteres es algo sumamente importante para la computación y ejercicios como este ayudan mucho a la manipulación de textos tanto pequeños como grandes. Se debe hacer una muy buena programación para los textos pequeños para poder confiar al cien por ciento la manipulación de textos grandes.

Bibliografía:

Diana Lucía Salas Víquez. (2011, May 01). El código de un general: un breteji tuani. *El Financiero*
https://www.proquest.com/newspapers/el-código-de-un-general-breteji-tuani/docview/864157623/se-2
Guilherme Thomazi Bonicontro, (2020). Arm.syscalls.sh
https://arm.syscall.sh/
(2020). ARM System Calls
http://www.ee.ic.ac.uk/pcheung/teaching/ee2_computing/swi.pdf