# Tarea #2

**Instituto Tecnológico de Costa Rica**

**Carrera:** Bachillerato en Ingeniería en Computación

**Curso:** Principios de Sistemas Operativos

**Profesor:** Kevin Moraga García

**Alumno:** Jonathan Quesada Salas

**Carnet:** 2020023583

**Tarea Corta 2:** Tutormec

---

### Introducción

En cuanto respecta a la segunda tarea se puede encontrar por medio de dos fases, la cual una será el booteo de una USB para que el programa pueda correr en el Master Boot Record, por otro lado se encuentra lo que contendrá la USB la cual será un programa en ensamblador llamado Tutormec, el cual consiste en:

- En presentar al usuario una interfaz donde las letras que tenga que presionar de derecha a izquierda.
- El centro de la interfaz contendrá al menos cinco bloques que puedan tener una separación entre ellos, está separación se usará para que bajen las letras.
- El usuario del programa tendrá que cachar las letrsa cuando estas se encuentren entre las separación de los bloques.
- Habrá una animación de la letra bajando por medio de los bloques
- Como consecuente cuando la letra baje del todo se deberá indicar al usuario cual dedo debería de usar para poder cachar la letra por defeto.

---

### Ambiente de desarrollo

Con lo que respecta al ambiente de desarrollo se usará el sistema operativo Ubuntu, específicamente la versión:

```
Ubuntu 20.04.2 LTS
```

En cuanto al IDLE se usará el **Visual Studio Code** para el desarrollo de la tarea.

Adicionalmente se usará un repositorio de Github para poder almacenar los avances significativos de la asignación de la primera tarea de principios de sistemas operativos llamado tarea1.

Por otra parte para la realización de la tarea se tendrá a disposición una USB para la realización del booteo de la llave maya.

Por último se utilizará el programa de QEMU, el cual es un emulador de procesadores basado en la traducción dinámica de binarios, el cual también tiene capacidades de virtualización dentro de un sistema operativo, ya sea GNU/Linux, Windows, o cualquiera de los sistemas operativos admitidos; de hecho es la forma más común de uso, para que de esta manera se realice de una mejor manera el tema de las pruebas en el transcurso de la realización de la tarea.

---

## Estructuras de datos usadas y funciones

**Principales funciones**

**start_tutormec** Inicia con la ejecución de tutormec

**check_input_buffer:** Lee buffer de entrada y toma el primer caracter que encuentre. Y verifica si el caracter es mismo que el de la pantalla. De ser el caso, enciende la bandera del objecto char correspondiente.

**move_char_objects** Actualiza las cordenadas del objecto char que se encuentra en la pantalla: Se mueve uno hacia la izquierda. Si la bandera esta encendida deja de moverse a la izquierda, y pasa a moverse hacia abajo. Si no se encuentra con ninguna caja entonces le suma uno al score. Y continua con la siguiente letra. Si se encuentra con hace lo mismo que el anterior, pero no suma nada al score.

**draw_screen** Dibuja todos los objetos en la pantalla, el label, la puntuación, las cajas y el carácter que se está moviendo.

**Principales estructuras**

**next_char_on_screen:** Almacena el indice del siguiente caracter que se mostrara en la pantalla.

**current_char_index_array:** Lista de los indices de char_object_array que se muestran en pantalla, por defecto únicamente tiene uno.

**char_object_array:** Lista de todas los objetos array, cada objeto array tiene un tamaño de 5bytes, donde se almacena: el carácter, la bandera de si fue presionada, la posición x, la posición y. E y de salida, en caso de que el programa mostrara más de un carácter.

**boxes_x_cordenates_array:** Almacena una estructura de datos de dos bytes. En el primer byte está el límite horizontal izquierdo de la caja y el segundo el límite horizontal derecho. De modo, que por cada caja hay dos bytes.

---

## Instrucciones para ejecutar el programa

1. Primeramente se debe de compilar el programa que va a contener el tutormec, este mismo se deberá pasar a un archivo binario ejecutable, con el siguiente comando de ejemplo:

   ```
   make
   ```

2. Ya habiendo ejecutado dicho comando y obtenido el archivo binario, que será el ejecutable, se procede a probar el programa con un emulador llamado **QEMU** con el siguiente comando:

   ```
   qemu-system-x86_64 bin/os.bin
   ```

3. Ya habiendo probado el binario de una manera éxitosa se procederá a hacer el quemado del binario a la USB que dispongamos, con el siguiente comando:

   ```
   sudo dd if=bin/os.bin of=/dev/sdb
   ```

   Con dicho comando se debe de tener el **CUIDADO** de digitar la dirección de la llave maya correctamente, ya que en este paso se puede realizar un desastre en

su computadora.
4. Adicionalmente se procede a poder probar el archivo binario quemado en la llave maya con ayuda del siguiente comando, más que todo como medida preventiva y cautelosa.

```
sudo qemu-system-x86_64 -hda /dev/sdb
```

5. Por consiguiente se procede a abrir el apartado de "Discos" del ordenador y dirigirse a la pestaña de la llave maya y agregarle una partición a la misma para que de esta manera nos aparezca en "Mis archivos" el apartado de la llave maya, para que de esta manera se pueda copiar y pegar el binario en dicho apartado.
6. Por último se procede a reiniciar la computadora y volverla a prender apretando F12, para poder escoger con que se quiere "arrancar" la computadora y ahí escogemos el nombre de la llave maya ingresada en ese momento.
7. Para finalmente retornar el programa de Tutormec :D

---

**Actividades realizadas por estudiante**

| Fecha | Inicio | Fin | Avance Realizado |
|---|---|---|---|
| 24/08/2022 | 3:00 pm | 5:00 pm | Investigar sobre como hacer un booteo en la llave maya |
| 25/08/2022 | 7:00 pm | 10:00 pm | Investigar sobre QEMU y hacer el quemado en la llave maya |
| 26/08/2022 | 5:00 pm | 7:00 pm | Booteo exitoso de la llave malla y creación de la documentación |
| 26/08/2022 | 3:00 pm | 5:00 pm | Probar el límite de TIMES por medio de tableros y repetir pasos de booteo formateando la llave maya con partición |
| 26/08/2022 | 6:00 pm | 1:00 am | Iniciar aprendizaje de ensamblador con manuales y tutoriales |
| 27/08/2022 | 9:00 am | 7:00 pm | Leer manuales de ensamblador e ir implementando ideas en el código aprendido |
| 28/08/2022 | 1:00 pm | 11:00 pm | Aprender ensamblador, leer manuales, seguir implementando ideas en ensamblador y avanzar la documentación |
| 29/08/2022 | 1:00 pm | 8:00 pm | Realizar investigación de como poder establecer las animaciones para las letras que deben de estar entre los bloques |
| 31/08/2022 | 6:00 pm | 1:00 am | Realizar el commit respectivo del avance investigado y terminar la documentación |
| 1/09/2022 | 5:00 pm | 8:00 am | Realizar el aspecto técnico faltante (decir que dedo presioanar la tecla) y actualizar la documentación |

---

## Autoevaluación

### Estado final

Con lo que respecta al estado final de la tarea, se ve contemplado los requerimientos funcionales y los requerimientos técnicos, por parte de la parte del booteo, pero en cuanto al tutormec también se vieron cubiertos todos los requisitos técnicos de la tarea, los cuales contempla la introducción y la especificación de la tarea

### Problemas y limitaciones

Con lo que respecta al booteo los problemas que llegaron a surgir fueron:

1. No poder encontrar de una manera rápida una información específica para realizar el booteo, hasta que encontré la referencia [2] ya que esta proporcionó la información necesaria para el "quemado" de la llave maya de un archivo binario.
2. Un problema que hubo que casi causa la pérdida total de todo lo que contiene la computadora fue no poner correctamente la dirección de la llave maya ya que esto hay que tener especial cuidado, porque al poner mal la dirección de la llave puede causar que uno como usuario ponga la dirección del disco duro, el cual causaría una pérdida total de la computadora, ya que en su momento no había hecho un respaldo.

Ahora bien con lo que respecta a la otra parte de la tarea la cual se refiere al Tutormec, se encontraron los siguientes problemas:

1. Primeramente con solo iniciar la tarea me percarté sobre la gran desinformación sobre el lenguaje ensamblador en cuestión, ya que esto fue un problema latente en dicha realización de la tarea, en cuanto al manejo de registros y el entendimiento del mismo.
2. Hilado al problema 1, fue la desinformación que tenía sobre algunas funcionalidades que pueda tener ensamblador, en cuanto a condicionales, saltos, instrucciones en general, para esto la referencia [1] me ayudó en cuento a este problema.
3. Cuando se inicializa la construccón del programa pesaba menos de 512 bytes, sin embargo, con el transcurso de la ejecución y desarrollo del programa de la tarea llego a presentar una limitación en cuento respecta a los 512 bytes que debe de tener para inicalizar dicho programa en la zona de master boot record, siendo esto un problema que se debió ver como simplificar procesos, como puede ser la creación del tablero.
4. Los registros para el almacenamiento en el trascurso de la tarea, adicionalemnte en ocasiones necesitaba direccionar a más de un lugar a la vez, la solución práctica se puede considerar que fue el uso de la pila, basicamente poder encontrar la forma de redireccionar registros a otros registros para poder administrar de una mejor manera los recursos y que el programa por lo menos pudiera compilar.
5. Adicionalmente uno de los problemas que llegué a encontrar con esta tarea fue el rango que debe de estar el programa para que no salga de los límites del ordenador en cuento se refiere a los monitores que estan en el LAIMI ya que en mi computadora es un espacio simulado, el cual lo hace un mundo ideal para que todo salga bien.

### Reporte de commits

```
commit 9549c0945683cdf41882f0597d9b17a654923518 (HEAD -> master, origin/master,
origin/HEAD)
Author: Nuwidra <https://github.com/Nuwidra/IC-2101-POO-2020-ii->
Date:   Thu Sep 1 00:47:26 2022 -0600

    documentation and development of last details of the program

commit ae64aeb7382d006c1b5c6db9e0cb456858aaf2f1
Author: Nuwidra <https://github.com/Nuwidra/IC-2101-POO-2020-ii->
Date:   Thu Sep 1 00:42:31 2022 -0600

    functional code

commit 717baa017652fa9a0f7ceb896b2446bb39726b1b
Author: Nuwidra <https://github.com/Nuwidra/IC-2101-POO-2020-ii->
Date:   Sat Aug 27 15:13:22 2022 -0600

    Board Creation

commit 37b53b31c8fe53dc069265658386b056b427da35
Author: Nuwidra <https://github.com/Nuwidra/IC-2101-POO-2020-ii->
Date:   Fri Aug 26 18:20:55 2022 -0600

    Boot advance successfully
(END)
```

**Calificación**

| Rubro | Porcentaje |
|---|---|
| Sector de Arranque | 30% |
| Tutormec | 50% |
| Documentación | 20% |

---

## Lecciones Aprendidas

1. Primeramente con lo que respecta a esta tarea se tiene en cuenta dos partes la de booteo y la de Tutormec, la lección más importante es que se empiece a programar desde una vez, desde el primer en el programa de Tutormec ya que el lenguaje ensamblador es sumamente tedioso de aprender por el manejo de variables y registros del mismo.
2. En cuanto respecta al ámbito necesario para que el programa pueda apreciarse correctamente en las computadoras del LAIMI, ya que en mi computadora tiene un ambiente virtual el cual todo es ideal para que todo funcione y aún más con ayuda de QEMU ya que este mismo es un emulador de esta forma fue tener que reducir los recursos que se lleguen a necesitar para el desarrollo del mismo para que este programa este en el ámbito de toda computadora con ayuda del quemdo de la llave maya con el programa extensión .bin
3. Una lección que llegué a presenciar por parte de la zona del booteo es tener el miedo de casi arruinar mi computadora por completo, ya que en el siguiente comando se debe de tener muy bien seleccionado cual es la dirección de la llave

maya ya que dicha dirección a comparación a la dirección del disco duro de la computadora se parecen mucho, por lo cual es propicio a errores manuales en cuanto al manejo del siguiente comando:

```
sudo dd if=boot.bin of=/dev/sdb
```

4. Algo adicional con lo que respecta al booteo es que nunca había hecho un booteo de este estilo ya que dicho programa debe de estar contemplado dentro de los 512 bytes que tiene la computadora para que esta misma pueda ejecutar en su kernel el sistema operativo y dicho programa debe de estar antes de eso, adicionalmente encontrar dichas lineas de código en ensamblador fueron muy útiles para el desarrollo de la tarea.

```
TIMES 510 - ($ - $$) db 0
dw 0xAA55
```

5. Tener un orden establecido para las referencias para el desarrollo del programa de Tutormec fue de mucha utilidad, ya que este mismo me sirvió para tener un mejor orden para el desarrollo de la tarea en ensamblador, ya que este lenguaje de programación esta sujeta a mucha observación y lógica para el programa.
6. En cuanto respecta al booteo conlleva más pasos de los que pensé ya que cuando aplicaba el comando descrito en la lección 4, no me funcionaba el comando ya que este solamente hace el quemado de la llave maya, para poder hacer que la llave maya logrará bootear de la mejor manera fue necesario hacer una partición en el apartado de "Discos" de la compu y dirigirse al apartado de la llave maya y realizar una partición y agregar el archivo de extensión .bin a la partición de la llave maya previamente realizada.
7. Un tema adicional que me logró hacer los respectivos avances de la tarea de tutormec fue la indexación el cual consiste en variantes del direccionamiento indexado en que se obtiene la dirección del operando sumando el contenido de varios registros con el desplazamiento, esto puede servir para especificar el comienzo de un vector mediante un desplazamiento respecto a un registro y el elemento del vector.

---

## Bibliografía


[1] D. A. Alpern. Instrucciones y directivas de los procesadores 8086 y 8088. [Online]. Available: https://www.alpertron.com.ar/INST8088.HTM. [Accessed: 27-Aug-2022].

[2] How to burn a boot loader onto a USB drive (boot-loader as a .bin). (s. f.). Super User. https://superuser.com/questions/1645622/how-to-burn-a-boot-loader-onto-a-usb-drive-boot-loader-as-a-bin. [Accessed: 23-Aug-2022].

[3] How to use dd command in linux. (s. f.). Linux Hint. https://linuxhint.com/dd_command_linux/. [Accessed: 23-Aug-2022].

[4] How to create a live Ubuntu USB drive with persistence for BIOS using only terminal? (s. f.). Ask Ubuntu. https://askubuntu.com/questions/817931/how-to-create-a-live-ubuntu-usb-drive-with-persistence-for-bios-using-only-termi. [Accessed: 23-Aug-2022].

[5] Peter Abel. Programación para IBM PC y Compatibles, Tercera Edición, Peter Abel. Recuperado de: https://scholar.google.es/scholar?

hl=es&as_sdt=0%2C5&q=lenguaje+ensamblador+y+programaci%C3%B3n+para+PC+IBM&btnG=. [Accessed: 27-Aug-2022].

[6] Assembly Programming Tutorial. (s. f.). Online Tutorials Library. https://www.tutorialspoint.com/assembly_programming/index.htm. [Accessed: 27-Aug-2022].

[7] Assembly Language Tutorial => Getting started with Assembly Language. (s. f.). Learn programming languages with books and examples. https://riptutorial.com/assembly. [Accessed: 27-Aug-2022].

[8] Guide, P. Intel® 64 and ia-32 architectures software developer's manual. Volume 3B: System programming Guide, Part, 2(11). https://read.seas.harvard.edu/~kohler/class/aosref/IA32-1.pdf. [Accessed: 28-Aug-2022].

[9] freeCodeCamp.org. (2022, 27 de abril). Assembly Language Programming with ARM – Full Tutorial for Beginners [Video]. YouTube. https://www.youtube.com/watch?v=gfmRrPjnEw4. [Accessed: 26-Aug-2022].

[10] Source Files in Assembly Language Format (IA-32 Assembly Language Reference Manual). (s. f.). Moved. https://docs.oracle.com/cd/E19455-01/806-3773/6jct9o0ad/index.html. [Accessed: 29-Aug-2022].

[11] SunSoft. (s. f.). x86 Assembly Language Reference Manual. Moved. https://docs.oracle.com/cd/E19641-01/802-1948/802-1948.pdf

[12] Mghacademy. (2021, 30 de enero). how to create animation(pictures) using assembly language [Video]. YouTube. https://www.youtube.com/watch?v=0jgXNk3npNs

[13] How to fix "os.asm:113: error: TIMES value -138 is negative" in assembly language. (s. f.). Stack Overflow. https://stackoverflow.com/questions/53858770/how-to-fix-os-asm113-error-times-value-138-is-negative-in-assembly-languag