



Tarea 1: MRPV

Instituto Tecnológico de Costa Rica
Escuela de Ingeniería en Computación
Centro Académico de Alajuela

Estudiante:

William Alfaro Quirós

Asignatura:

IC-6600: Principios de Sistemas Operativos

Grupo:

20

Profesor:

Ing. Kevin Moraga, MSc.

Fecha de entrega:

18 de marzo del 2025

Semestre I

Índice

1. Introducción	2
2. Ambiente de desarrollo	2
3. Estructuras de datos usadas y funciones	3
4. Instrucciones para ejecutar el programa	4
5. Instrucciones para Ejecutar el Programa	4
6. Actividades realizadas por estudiante	5
7. Autoevaluación	5
8. Lecciones Aprendidas	6
9. Bibliografía	8
10. Anexos	9

1. Introducción

El proceso de arranque de un sistema operativo es una etapa crítica en el funcionamiento de cualquier computadora, ya que establece la transición entre el encendido del hardware y la carga del software necesario para interactuar con el usuario. Comprender detalladamente esta fase permite profundizar en el conocimiento del hardware y software involucrados en sistemas informáticos.

En este contexto, el presente trabajo aborda la implementación de una pequeña aplicación denominada "Mike Romeo Papa Victor (MRPV)" desarrollada en lenguaje ensamblador para la arquitectura x86, cuyo propósito es facilitar el aprendizaje del alfabeto fonético internacional (OACI). Este alfabeto, creado por la Organización de Aviación Civil Internacional en 1944, es una herramienta clave utilizada en la comunicación radial en ámbitos como la aviación y la marina, para garantizar claridad y precisión en la transmisión de mensajes críticos [1].

El programa "MRPV" genera aleatoriamente cadenas de caracteres y reta al usuario a deletrearlas empleando el alfabeto fonético internacional. A medida que el usuario introduce cada palabra, el sistema evalúa la exactitud del deletreo fonético y asigna un puntaje basado en su desempeño, fomentando un aprendizaje interactivo y práctico.

2. Ambiente de desarrollo

Para llevar a cabo la implementación del proyecto se utilizaron diversas herramientas específicas que facilitaron el desarrollo del código fuente en lenguaje Ensamblador, así como las pruebas y ejecución del programa:

- NASM versión 2.16.01: Ensamblador de código abierto para arquitectura x86, usado para ensamblar el código fuente.
- QEMU: Herramienta de virtualización empleada para probar el arranque y ejecución del programa desarrollado sin necesidad de hardware físico.
- Editor de texto Visual Studio Code: Utilizado para escribir, depurar y administrar el código fuente.
- Git: Utilizado para el control de versiones, facilitando la organización del trabajo y el seguimiento de cambios durante el desarrollo del proyecto.
- Linux Ubuntu 22.04: Sistema operativo utilizado como entorno principal para el desarrollo y ejecución del proyecto.
- Alfabeto radiofónico: Cuadro para el deletreo de letras y cifras [1]

Cuadro 1: Alfabeto Fonético Internacional (OTAN) [2]

Carácter	Palabra
a	alfa
b	bravo
c	charlie
d	delta
e	echo
f	foxtrot
g	golf
g	hotel
i	india
j	juliett
k	kilo
l	lima
m	mike
n	november
o	oscar
p	papa
q	quebec
r	romeo
s	sierra
t	tango
u	uniform
v	victor
w	whiskey
x	x-Ray
y	yankee
z	zulu
1	one
2	two
3	three
4	four
5	five
6	six
7	seven
8	eight
9	nine

3. Estructuras de datos usadas y funciones

Durante el desarrollo del proyecto se definieron estructuras de datos y funciones esenciales que permitieron el correcto funcionamiento del programa. A continuación se detallan claramente cada una:

Las estructuras principales utilizadas son:

- **Tabla de conversión:** Estructura estática que almacena cada letra del alfabeto inglés junto

con su correspondiente palabra en el alfabeto fonético internacional, facilitando la validación precisa de las entradas del usuario.

- **Buffer de entrada:** Espacio temporal en memoria destinado al almacenamiento provisional de las cadenas introducidas por el usuario, previo a su validación mediante la tabla de conversión.

Funciones principales

Entre las principales funciones implementadas en el proyecto destacan:

- **Generador de cadenas aleatorias:** Esta función se encarga de generar cadenas de caracteres de manera aleatoria, las cuales posteriormente deben ser deletreadas por el usuario utilizando el alfabeto fonético internacional.
- **Validación de entrada:** Función responsable de comparar el deletreo fonético proporcionado por el usuario contra la tabla de conversión, evaluando su exactitud y asignando puntos de acuerdo con el resultado.
- **Rutinas de entrada/salida:** Conjunto de funciones encargadas de mostrar información al usuario en pantalla y recoger entradas desde el teclado, permitiendo la interacción efectiva durante la ejecución del programa.

4. Instrucciones para ejecutar el programa

A continuación se presentan los pasos necesarios para compilar, crear y ejecutar la imagen del programa:

5. Instrucciones para Ejecutar el Programa

Para compilar, crear la imagen de disco, probarla en un emulador y finalmente escribir la imagen en una memoria USB, se deben ejecutar los siguientes comandos:

1. Compilar el bootloader:

```
nasm -f bin -o boot.bin boot.asm
```

2. Compilar el programa principal:

```
nasm -f bin -o program.bin program.asm
```

3. Crear una imagen de disco usando la herramienta dd:

```
dd if=boot.bin of=disk.img bs=512 count=1  
dd if=program.bin of=disk.img bs=512 seek=1
```

4. Ejecutar y probar la imagen en un emulador como QEMU:

```
qemu-system-x86_64 -fda disk.img
```

5. Escribir la imagen en la memoria USB:

```
sudo dd if=boot.bin of=/dev/sdb bs=512 status=progress
```

Cada uno de estos comandos debe ejecutarse en el entorno de desarrollo adecuado (por ejemplo, Ubuntu 24.04.2 LTS) para garantizar el correcto funcionamiento del programa.

6. Actividades realizadas por estudiante

A continuación se resumen las principales actividades llevadas a cabo durante el desarrollo del proyecto:

Cuadro 2: Actividades realizadas por estudiante

Fecha	Descripción breve de la actividad	Horas
5 de marzo	Investigación inicial sobre el proceso de booteo y sector de arranque MBR.	3
6 de marzo	Instalación y configuración del ambiente con NASM y QEMU en Ubuntu 22.04.	2
7 de marzo	Diseño inicial y pseudocódigo de la aplicación MRPV.	2
8 de marzo	Implementación y prueba inicial del bootloader.	4
9 de marzo	Desarrollo y validación de la tabla del alfabeto fonético.	4
10 de marzo	Implementación de generación aleatoria de cadenas y validación de entradas.	4
11 de marzo	Pruebas finales del programa completo.	5
16 de marzo	Documentación final y preparación del informe del proyecto.	3
Total horas invertidas		27

7. Autoevaluación

El programa implementado cumplió satisfactoriamente con los objetivos planteados al inicio del proyecto. Se logró desarrollar exitosamente la generación aleatoria de cadenas de caracteres y validar las entradas del usuario mediante el alfabeto fonético internacional. Asimismo, se adquirió experiencia valiosa en el desarrollo con ensamblador, manejo del sector de arranque MBR y uso de herramientas como NASM y QEMU.

Considerando los objetivos cumplidos y los resultados obtenidos, se autoevalúa esta tarea con una calificación de **100 (nota máxima)**.

Limitaciones identificadas

Entre las principales limitaciones del programa destacan:

- El programa únicamente admite caracteres pertenecientes al alfabeto inglés (A-Z). Caracteres especiales, símbolos o letras adicionales no son soportadas actualmente.
- No existe validación de entradas fuera del rango mencionado, lo que podría conducir a comportamientos inesperados.

A continuación, se resume en la Tabla 3 el registro principal de commits realizados durante el desarrollo del proyecto. El repositorio completo del proyecto se encuentra disponible en el siguiente enlace:

<https://github.com/BillyyBoyy/Tarea1-S0>

Cuadro 3: Registro de Commits realizados en el repositorio

Fecha	Descripción del commit
5 de marzo	Creación del repositorio e investigación inicial sobre el sector de arranque (MBR).
6 de marzo	Instalación y configuración del entorno de desarrollo con NASM y QEMU en Ubuntu 22.04.
7 de marzo	Diseño inicial y pseudocódigo del programa MRPV.
8 de marzo	Implementación y pruebas iniciales del bootloader.
9 de marzo	Desarrollo y validación de la tabla del alfabeto fonético internacional.
10 de marzo	Implementación de generación aleatoria de cadenas y validación de entradas.
11 de marzo	Corrección de errores relacionados con validación y contador de puntuación.
16 de marzo	Documentación final y preparación del informe del proyecto.
18 de marzo	Documentación completa y actualizada para entrega.

8. Lecciones Aprendidas

Durante el desarrollo del presente proyecto se identificaron aprendizajes clave que fortalecieron la formación técnica y metodológica del estudiante. A continuación se resumen los más importantes:

- **Proceso de arranque del sistema operativo:** Comprender detalladamente el proceso de booteo del sistema operativo aportó habilidades técnicas valiosas que trascienden el ámbito académico, reforzando la comprensión del funcionamiento entre software y hardware.
- **Dominio de herramientas específicas:** El manejo adecuado de herramientas de desarrollo y prueba como NASM, QEMU y Git desde etapas tempranas facilitó significativamente la eficiencia en la programación, depuración y administración del código fuente.
- **Planificación y diseño previo:** La creación anticipada de diseños y pseudocódigo es crucial al trabajar en ensamblador, especialmente debido a su complejidad y cercanía al hardware, permitiendo detectar errores tempranamente y optimizar el proceso de desarrollo.

- **Ventajas del uso temprano de control de versiones:** Utilizar Git desde el inicio del proyecto facilitó considerablemente la gestión ordenada del código, permitiendo un seguimiento claro y eficiente de los cambios durante el desarrollo, además de agilizar la colaboración en equipo.

9. Bibliografía

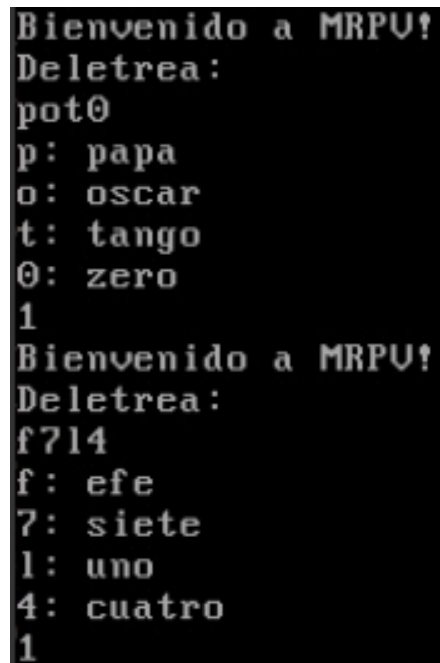
- [1] International Civil Aviation Organization (ICAO), *Aeronautical Telecommunications: Procedures for Air Navigation Services*, Vol. II, 7th ed., Montreal, Canada: ICAO, 2016.
- [2] Wikipedia, "Alfabeto fonético internacional," *Wikipedia, la enciclopedia libre*, 16 marzo 2025. [En línea]. Disponible en: [https://es.wikipedia.org/wiki/Alfabeto_fonÃ©tico_de_la_OTAN](https://es.wikipedia.org/wiki/Alfabeto_fon%C3%A9tico_de_la_OTAN). [Accedido: 16-mar-2025].

10. Anexos

Anexo A: Ejemplos de ejecución del programa MRPV

En este anexo se presentan ejemplos ilustrativos del funcionamiento del programa *Mike Romeo Papa Victor (MRPV)*, destacando dos escenarios de uso y la interacción con el usuario.

Ejemplo 1:



```
Bienvenido a MRPV!  
Deleetrea:  
pot0  
p: papa  
o: oscar  
t: tango  
0: zero  
1  
Bienvenido a MRPV!  
Deleetrea:  
f714  
f: efe  
7: siete  
1: uno  
4: cuatro  
1
```

Figura 1: Captura de pantalla del Ejemplo 1

- **Cadena generada: pot0**

*El usuario deleetreó correctamente todos los caracteres con el alfabeto fonético internacional: **papa, oscar, tango, zero.***

Resultado: Puntuación completa (+1 punto).

- **Cadena generada: f714**

El usuario cometió errores en los caracteres numéricos, utilizando el deleetreo en español (“siete, uno, cuatro”) en lugar del estándar OACI (“seven, one, four”).

Resultado: Sin puntuación (+0 puntos).

Ejemplo 2:

```
Bienvenido a MRPU!  
Deletrea:  
net8  
n: nine  
e: echo  
t: tango  
8: eight  
2  
Bienvenido a MRPU!  
Deletrea:  
s2me  
s: sierra  
2: two  
m: mike  
e: echo  
4
```

Figura 2: Captura de pantalla del Ejemplo 2

- **Cadena generada: net8**

*El usuario deletreó correctamente todos los caracteres utilizando el alfabeto fonético internacional: **november, echo, tango, eight.***

Resultado: Puntuación completa (+2 puntos).

- **Cadena generada: s2me**

*El usuario deletreó correctamente todos los caracteres utilizando el alfabeto fonético internacional: **sierra, two, mike, echo.***

Resultado: Puntuación completa (+2 puntos).