UNIVERSIDAD NACIONAL JORGE BASADRE GROHMANN

FACULTAD DE INGENIERÍA

Escuela Profesional de Ingeniería en Informática y Sistemas

Mini-Shell

Curso: Sistemas operativos

Docente: Hugo Manuel Barraza Vizcarra

Integrantes

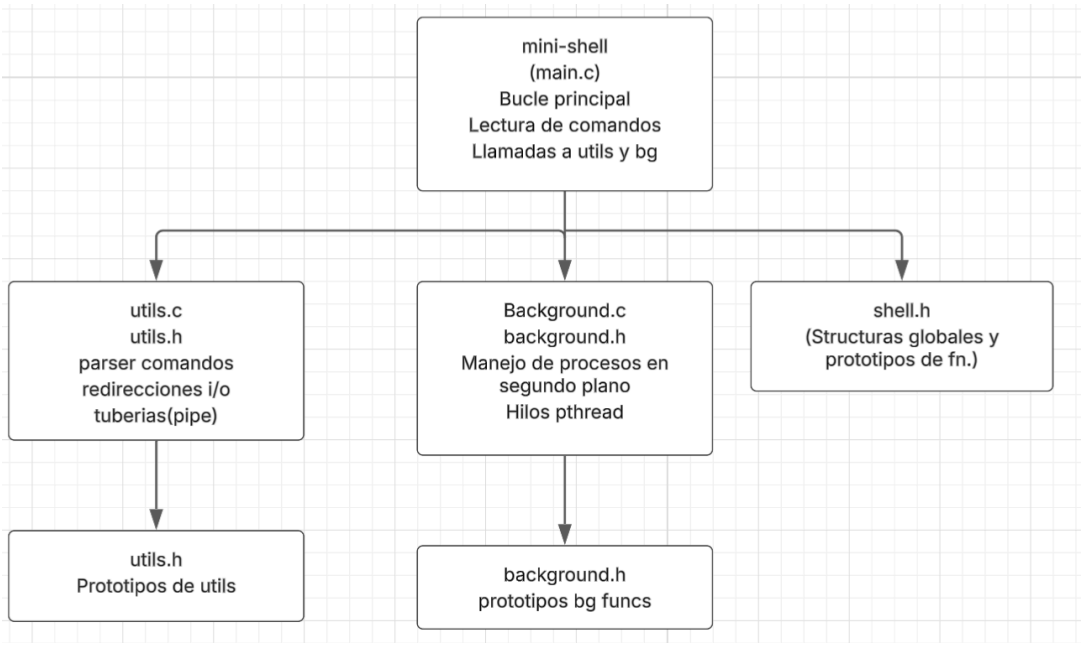
Jhon Alfredo Mamani Tacora 2023-119036
Jhon Sebastian Mamani Ancco 2022-119022

Tacna - Perú
2025

1. **Objetivo y alcance:**

   El objetivo principal de este proyecto fue desarrollar una mini-shell funcional en lenguaje C que permita al usuario ejecutar comandos del sistema de forma interactiva, similar a una terminal de Linux. Esta shell soporta operaciones básicas como cambio de directorio con el comando `cd`, ejecución de procesos en primer y segundo plano, manejo de señales como `SIGINT`, y la implementación de redirecciones de entrada y salida, así como tuberías para la comunicación entre procesos. Además, se incorporó la gestión de tareas en segundo plano mediante hilos para monitorear su finalización de manera asíncrona. El alcance del proyecto abarca la comprensión y aplicación práctica de conceptos fundamentales de sistemas operativos, como procesos, señales, sincronización con mutex y comunicación entre procesos, ofreciendo una base sólida para ampliar sus funcionalidades en el futuro.

2. **Arquitectura y diseño:**



   La arquitectura del proyecto Mini-Shell está organizada de forma modular y jerárquica, separando las responsabilidades en distintas carpetas y archivos para mejorar la legibilidad y el mantenimiento del código. En la carpeta include se encuentran los archivos de cabecera (.h) que contienen las declaraciones de funciones, estructuras y constantes utilizadas en todo el programa, como background.h, shell.h y utils.h. En la carpeta src se ubican las implementaciones (.c), donde cada módulo desarrolla una funcionalidad específica: background.c gestiona los procesos en segundo plano, utils.c contiene funciones auxiliares como el manejo de señales, y

main.c actúa como el punto de entrada principal que coordina la ejecución de los comandos. Finalmente, mini_shell.c integra las diferentes partes del sistema, demostrando una arquitectura clara, extensible y bien estructurada.

Módulo/Archivos	Descripción	Funciones claves
main.c	Punto de entrada del shell	Bucle principal, lectura del prompt, manejo de comandos y redirecciones
background.c	Módulo de procesos en segundo plano	Usa hilos (`pthread`) para monitorear procesos y mostrar su estado
utils.c	Funciones auxiliares	Tokenización, detección de operadores (>, <, >>)
shell.h	cabecera principal	Estructuras de datos globales, definiciones y funciones comunes
background.h	Cabecera de segundo plano	Prototipos y estructuras de manejo de tareas background
utils.h	Cabecera de utilidades	Prototipos de funciones de parsing, redirección y ayuda

3. Detalles de implementación

El proyecto Mini-Shell en C consiste en la implementación de un intérprete de comandos desarrollado en lenguaje C, capaz de ejecutar procesos, manejar tuberías (pipes), realizar redirecciones de entrada y salida, ejecutar tareas en segundo plano y mantener una estructura modular que facilita su mantenimiento y comprensión. La organización del proyecto se divide en las carpetas src/, que contiene el código fuente en archivos .c, y include/, donde se encuentran los archivos de cabecera .h, junto con un archivo README.md que documenta su uso y funcionamiento. Entre sus principales características destacan su interfaz interactiva, que muestra un prompt (mini-shell@usuario>) para recibir comandos del usuario; la ejecución de comandos externos del sistema; y la inclusión de comandos internos como cd (para cambiar de directorio), salir (para finalizar la ejecución) y help (para mostrar la lista de comandos disponibles). Asimismo, soporta redirecciones (>, >>, <) para manipular archivos y

tuberías (|) que permiten conectar comandos, como en ls | grep ".c". Para su compilación y ejecución, se requiere un sistema operativo tipo Unix/Linux con el compilador gcc instalado, tras lo cual el programa puede compilarse con gcc src/*.c -Iinclude -o mini-shell -lpthread y ejecutarse mediante ./mini-shell.

4. Concurrencia y sincronización

En el proyecto Mini-Shell, la concurrencia y sincronización se implementan principalmente mediante el uso de hilos (threads) y mutex para gestionar la ejecución de procesos en segundo plano de forma segura. Cada vez que un comando se ejecuta con el operador &, se crea un hilo independiente encargado de monitorear el estado del proceso hijo sin bloquear la ejecución principal de la shell, lo que permite que el usuario siga introduciendo nuevos comandos. Para evitar condiciones de carrera durante la impresión de mensajes simultáneos desde distintos hilos, se emplea un mutex global (print_mutex), garantizando que solo un hilo tenga acceso al recurso compartido (la salida estándar) en un momento dado. De esta manera, el proyecto demuestra una aplicación práctica de los conceptos de concurrencia, sincronización y comunicación entre procesos, fundamentales en el diseño de sistemas operativos modernos.

5. Gestión de memoria

En el proyecto Mini-Shell, la gestión de memoria se maneja de manera dinámica y controlada para asegurar un uso eficiente de los recursos del sistema. Cuando el usuario ejecuta comandos en segundo plano, se utiliza la función malloc() para asignar memoria a estructuras del tipo bg_job_t, que almacenan información del proceso y el comando ejecutado. Una vez que el hilo encargado de monitorear el proceso termina su tarea, dicha memoria se libera con free(), evitando fugas de memoria. Además, el programa maneja cadenas y argumentos mediante arreglos de punteros y funciones como strtok() y strncpy(), cuidando no exceder los límites definidos por constantes como MAX_INPUT o CMD_STR_LEN. Esta implementación refleja una correcta aplicación de la memoria dinámica y estática, garantizando estabilidad y eficiencia en la ejecución de la shell.

6. Prueba y resultado:

Durante la etapa de pruebas del proyecto Mini-Shell, se realizaron diferentes ensayos con el objetivo de comprobar el correcto funcionamiento de sus principales componentes: la ejecución de comandos, la gestión de procesos, la concurrencia, el manejo de señales y la administración de memoria. En primer lugar, se verificó la ejecución de comandos básicos como ls, pwd, echo y cat, confirmando que la shell interpreta correctamente las órdenes del usuario y muestra los resultados en pantalla.

Asimismo, se evaluó el comportamiento del comando cd, tanto con rutas absolutas como relativas, comprobándose que al no especificar una ruta el programa redirige adecuadamente al directorio principal del usuario.

Posteriormente, se realizaron pruebas relacionadas con la ejecución de procesos en segundo plano mediante el uso del operador &. Al ejecutar comandos como sleep 5 & o ping localhost &, se observó que la shell permite seguir introduciendo nuevas órdenes mientras las tareas en segundo plano se ejecutan. Al finalizar, se muestra un mensaje indicando el PID del proceso terminado, evidenciando el correcto funcionamiento de los hilos de monitoreo y la sincronización entre procesos. También se evaluó la gestión de señales, en particular la respuesta al comando Ctrl + C, comprobando que la shell ignora esta señal en el proceso principal sin cerrarse, pero la transmite correctamente a los procesos hijos cuando corresponde.

En cuanto al manejo de entrada y salida, se probaron redirecciones como ls > salida.txt y cat < entrada.txt, confirmando que el programa redirige de forma adecuada la entrada y salida estándar hacia los archivos especificados. Finalmente, se realizaron pruebas de estabilidad y gestión de memoria, ejecutando múltiples procesos de forma consecutiva y simultánea. Los resultados demostraron que el programa no presenta pérdidas de memoria ni bloqueos, gracias al uso apropiado de la liberación de recursos y la sincronización de hilos. En general, las pruebas confirmaron que la Mini-Shell cumple con los objetivos propuestos: ejecuta comandos de manera estable, gestiona correctamente la concurrencia, mantiene la integridad de la memoria y ofrece un funcionamiento eficiente y confiable.

7. Conclusiones

En conclusión, el desarrollo de la Mini-Shell permitió aplicar de manera práctica conceptos fundamentales de los sistemas operativos, como la creación y gestión de procesos, la concurrencia mediante hilos, la sincronización con mutex y la administración de memoria dinámica. A lo largo del proyecto se logró implementar una shell funcional capaz de ejecutar comandos básicos, manejar redirecciones de entrada y salida, controlar señales del sistema y gestionar tareas en segundo plano sin afectar la ejecución principal.

El diseño modular del código, dividido en archivos de encabezado e implementación, facilitó la organización, reutilización y mantenimiento del programa. Además, las pruebas realizadas demostraron la estabilidad y eficiencia del sistema, evidenciando una correcta coordinación entre los procesos y el uso seguro de los recursos compartidos. En conjunto, este proyecto representa una sólida integración de teoría y práctica, reforzando la comprensión del funcionamiento interno de una terminal y sentando las bases para futuras ampliaciones o mejoras, como la inclusión de tuberías múltiples, historial de comandos o manejo avanzado de errores.