# Laboratorio 2: Sistemas Operativos

**Profesor:** Viktor Tapia
**Ayudantes de cátedra:** Muryel Constanzo y Nicolás Schiaffino
**Ayudantes de Laboratorio:** Ian Rossi y Luciano Yevenes

15 de septiembre de 2025

## 1. Reglas Generales

Para la siguiente tarea se debe realizar un código programado en lenguaje C o C++. Se exigirá que los archivos se presenten de la forma más limpia y legible posible. Deberá incluir un archivo README con las instrucciones de uso y ejecución de sus programas junto a cualquier indicación que sea necesaria, y un archivo MAKE para poder ejecutar el programa.

## 2. Contexto

Después del éxito al descifrar las pistas del equipo perdido en Siberia, las autoridades decidieron enviar un equipo de rescate especializado. Ustedes, como parte del equipo de apoyo tecnológico, fueron invitados a acompañar la expedición para proporcionar soporte técnico en caso de necesitar descifrar más información. Todo parecía ir según el plan hasta que, durante el vuelo sobre el Océano Ártico, una tormenta inesperada provocó que el avión de rescate se estrellara en una isla remota y desconocida. Afortunadamente, todos los tripulantes sobrevivieron al accidente, pero ahora se encuentran varados en una isla misteriosa con recursos limitados y sin comunicación con el exterior.

El piloto, antes de que los sistemas fallaran completamente, logró enviar las coordenadas aproximadas de la isla, pero el rescate podría tardar semanas en llegar. Mientras tanto, el equipo debe organizarse para sobrevivir, y es aquí donde sus habilidades en programación de sistemas operativos se vuelven cruciales.

El líder de la expedición ha decidido implementar un sistema de supervivencia organizado donde diferentes grupos de sobrevivientes se encarguen de tareas específicas trabajando en paralelo para maximizar las posibilidades de supervivencia del grupo completo.

### 2.1. La Situación

La isla tiene varios recursos dispersos que deben ser recolectados y gestionados eficientemente:

- **Agua dulce**: Fuentes limitadas que deben ser monitoreadas constantemente
- **Alimentos**: Frutas, peces y otros recursos que requieren recolección activa
- **Materiales de construcción**: Madera, piedras y otros elementos para refugio
- **Señales de rescate**: Mantener fuegos y señales visibles para posibles rescatistas

### 2.2. El Sistema de Supervivencia

Deben crear un programa que simule el sistema de gestión de supervivencia donde:

- Existe un **coordinador principal** (proceso padre) que supervisa todas las operaciones
- **4 equipos de recolección** (procesos hijos creados con Fork()) que se encargan de diferentes tareas
- Cada equipo trabaja de forma independiente pero debe reportar sus resultados al coordinador
- El sistema debe funcionar por **turnos de supervivencia** (rondas) donde cada turno representa un día de supervivencia

## 3. Tarea

Concretamente, deben escribir un programa que logre lo siguiente:

### 3.1. Estructura del Sistema

El programa debe crear:

- 1 proceso **Coordinador** (proceso padre)
- 4 procesos **Equipos de Recolección** (procesos hijos)

### 3.2. Equipos de Recolección

Cada equipo tiene una especialización:

1. **Equipo de Agua**: Recolecta y purifica agua
2. **Equipo de Alimentos**: Caza, pesca y recolecta alimentos
3. **Equipo de Construcción**: Recolecta materiales y construye refugios
4. **Equipo de Señales**: Mantiene fogatas y señales de rescate

### 3.3. Equipos de Recolección

Cada turno (día de supervivencia) funciona así:

1. **Asignación de tareas**: El coordinador asigna objetivos a cada equipo
2. **Recolección simultánea**: Todos los equipos trabajan en paralelo usando Fork()
3. **Resultados aleatorios**: Cada equipo tiene éxito variable basado en probabilidades:
    - **Éxito total**: Cumple 100 % del objetivo (probabilidad 30 %)
    - **Éxito parcial**: Cumple 50-80 % del objetivo (probabilidad 50 %)
    - **Fracaso**: Cumple menos del 30 % del objetivo (probabilidad 20 %)
4. **Reporte**: Cada equipo reporta sus resultados al coordinador
5. **Evaluación**: El coordinador evalúa si el grupo sobrevive otro día

### 3.4. Sistema de Supervivencia

El grupo necesita cada día:

- **Agua**: Mínimo 8 unidades (2 por equipo)
- **Alimentos**: Mínimo 12 unidades (3 por equipo)
- **Refugio**: Mínimo 4 unidades (1 por equipo)
- **Señales**: Mínimo 2 unidades (para mantenerse visibles)

Si no se cumplen los mínimos, el grupo pierde "puntos de moral"(**la perdida de puntos de moral debe ser definida por ustedes**). Con 0 puntos de moral, la simulación termina.

### 3.5. Condiciones de Victoria

El juego puede terminar de tres formas:

- **Rescate exitoso**: Sobreviven 10 días consecutivos manteniendo señales activas
- **Fracaso**: Los puntos de moral llegan a 0 (inician en 100)
- **Límite de tiempo**: El usuario define cuántos días simular (máximo 30, minimo 10)

## 4. Especificaciones Técnicas

### 4.1. Comunicación entre Procesos

- Usar **shared memory** para comunicación entre coordinador y equipos
- Cada equipo debe reportar: ID del equipo, recursos recolectados, estado del equipo

### 4.2. Interfaz de Usuario

- El coordinador debe mostrar por consola el progreso de cada día
- Mostrar recursos disponibles, moral del grupo, y estado de cada equipo
- Al final de cada día, mostrar un resumen de supervivencia

### 4.3. Interfaz de Usuario

```
=== DÍA 1 DE SUPERVIVENCIA ===
Iniciando equipos de recolección...

[EQUIPO AGUA - PID: 1234] Explorando fuentes de agua...
[EQUIPO ALIMENTOS - PID: 1235] Explorando territorio para cazar...
[EQUIPO CONSTRUCCIÓN - PID: 1236] Buscando materiales de construcción...
[EQUIPO SEÑALES - PID: 1237] Recolectando combustible seco...

[EQUIPO AGUA - PID: 1234] Recolectando agua del arroyo encontrado...
[EQUIPO CONSTRUCCIÓN - PID: 1236] Cortando ramas útiles...
[EQUIPO ALIMENTOS - PID: 1235] Intentando pescar en la laguna...
[EQUIPO SEÑALES - PID: 1237] Manteniendo fogata de señales...
```

```
[EQUIPO AGUA - PID: 1234] Purificando agua recolectada...
[EQUIPO ALIMENTOS - PID: 1235] ¡Capturado pez pequeño! Preparando...
[EQUIPO CONSTRUCCIÓN - PID: 1236] Construyendo refugio básico...
[EQUIPO SEÑALES - PID: 1237] Creando señales de humo...

REPORTES FINALES:
- Equipo Agua completó ciclo: 3 unidades obtenidas
- Equipo Alimentos completó ciclo: 4 unidades obtenidas
- Equipo Construcción completó ciclo: 2 unidades obtenidas
- Equipo Señales completó ciclo: 3 unidades obtenidas

RESULTADOS DEL DÍA:
- Agua recolectada: 9/8 (SUFICIENTE)
- Alimentos obtenidos: 11/12 (INSUFICIENTE)
- Materiales de construcción: 5/4 (SUFICIENTE)
- Señales mantenidas: 3/2 (SUFICIENTE)

Estado: SUPERVIVENCIA CRÍTICA (falta alimento)
Moral del grupo: 85/100 (-15 por falta de comida)

Todos los procesos terminaron correctamente.
```

## 5. Consideraciones Específicas

## 6. README

Debe contener como mínimo:

- Nombre, Rol y Paralelo de los integrantes.
- Especificación de los algoritmos y desarrollo realizado.
- Supuestos utilizados.

## 7. Consideraciones Generales

- Deben hacer uso de Make, **no** de CMake.
- Se deberá trabajar **OBLIGATORIAMENTE** en parejas.
- Se deberá entregar a través de Github a mas tardar el día 10 de Octubre a las 23:59 horas.
- Se descontarán 10 puntos por cada hora o fracción de atraso.
- Las copias serán evaluadas con nota 0 en el promedio de las tareas.
- Si se detecta uso excesivo de IA, se llamará al grupo a una interrogación presencial con los ayudantes, de la cual también será parte el profesor.
- La tarea debe ser hecha en el lenguaje C o C++. Se asume que usted sabe programar en este lenguaje, ha tenido vivencias con el, o que aprende con rapidez.

- El código debe ser entregado en forma de **1 solo archivo** `.cpp` o `.c`, nombrado en base al formato `"LAB2_ApellidoIntegrante1_ApellidoIntegrante2"`
- Los códigos serán pasados por un software anti-plagio, en caso de ser detectada copia se pondrá nota 0, hasta que se realice una reunión con los grupos involucrados.
- Pueden crear todas las funciones auxiliares que deseen, siempre y cuando estén debidamente comentadas.
- Las tareas serán ejecutadas en Linux, cualquier tarea que no se pueda ejecutar en dicho sistema operativo, partirá de nota máxima 60.
- Las preguntas deben ser hechas por Aula a través del foro o del servidor de Discord. De esta forma los demás grupos pueden verse beneficiados también.
- Si no se entrega README o MAKE, o si su programa no funciona, la nota es 0 hasta la recorrección.
- Se descontarán hasta 50 puntos por:
    - Mala implementación del Makefile.
    - No respetar el formato de entrega.
    - No respetar las especificaciones del README.
    - Solicitar edición de código al momento de revisar.
    - Código poco prolijo y mal estructurado (ausencia de indentación adecuada, falta de consistencia en el estilo, nombres de variables confusos o poco descriptivos, comentarios insuficientes o irrelevantes).
- **Una vez publicadas las notas tendrán 5 días para apelar con el corrector que les revisó, después de este plazo las notas no tendrán ningún tipo de cambio**.