📚 POPURRÍ DE EXÁMENES - SISTEMAS OPERATIVOS Y C

📖 ESTRUCTURA DEL DOCUMENTO

Este document contiene **115+ preguntas** de exámenes anteriores organizadas por temas:

1. **Fundamentos de C y Punteros** (7 preguntas)
2. **Memoria Dinámica** (6 preguntas)
3. **Procesos y Fork** (5 preguntas)
4. **Llamadas al Sistema** (8 preguntas)
5. **Gestión de Ficheros** (4 preguntas)
6. **Tuberías (Pipes)** (4 preguntas)
7. **Shell y Variables** (5 preguntas)
8. **Memoria Virtual** (3 preguntas)
9. **Planificación** (3 preguntas)
10. **Concurrencia** (6 preguntas)
11. **Ficheros ELF** (5 preguntas)
12. **Sistemas de Ficheros** (2 preguntas)
13. **Señales** (3 preguntas)

✅ CÓMO USAR ESTE DOCUMENTO

Para prepararte para el examen:

1. **Lee un tema completo** (ej: "Procesos y Fork")
2. **Intenta responder** sin mirar la solución
3. **Compara tu respuesta** con la correcta
4. **Entiende la explicación** (no solo memorices)
5. **Repite** temas donde tengas dudas

Si tienes dudas sobre una pregunta:

- Pregunta directamente: "¿Por qué en la pregunta X sobre fork la respuesta es Y?"

- Me basaré en el temario de la asignatura para explicarlo

🎯 TEMAS CRÍTICOS (Aparecen en todos los exámenes)

⭐ ⭐ ⭐ Muy frecuentes (20%+ de preguntas):

- Procesos y fork()
- Memoria dinámica (malloc/free)
- Valores de retorno de syscalls
- Punteros y arrays

⭐ ⭐ Frecuentes (10-20% de preguntas):

- Tuberías y pipes
- Variables de entorno
- Ficheros y I/O
- Concurrencia

⭐ Menos frecuentes (5-10% de preguntas):

- Paginación
- Planificación
- Señales
- ELF y enlazado

💡 ERRORES MÁS COMUNES EN EXÁMENES

Error	Tema	Solución
Olvida que `sizeof(p)` devuelve tamaño del puntero	Memoria	Usa `sizeof(*p)` o `sizeof(int)`
Piensa que malloc debe estar en un for	Punteros	malloc asigna memoria para toda la vida

Error	Tema	Solución
No entiende fork devuelve diferente en padre e hijo	Procesos	Padre recibe PID, hijo recibe 0
Cree que las variables de shell se heredan	Shell	Solo se heredan si se exportan
Olvida que los pipes tienen tamaño limitado	Pipes	El buffer es típicamente 64KB
No controla bien el valor de retorno de read/write	Syscalls	Pueden devolver menos bytes de los pedidos
Piensa que exec devuelve si tiene éxito	Syscalls	Si tiene éxito, reemplaza el proceso
Cree que sizeof(array) en puntero devuelve su tamaño	Arrays	Devuelve el tamaño del puntero (4 u 8 bytes)

🔍 REFERENCIAS RÁPIDAS

Return Values de Funciones Críticas

```
fork():
  - Padre: PID del hijo (>0)
  - Hijo: 0
  - Error: -1

read(fd, buf, n):
  - Éxito: número de bytes leídos (0-n)
  - EOF: 0
  - Error: -1

write(fd, buf, n):
  - Éxito: número de bytes escritos (≤n)
  - Error: -1

exec():
  - Éxito: NO RETORNA (reemplaza el proceso)
  - Error: -1

open():
  - Éxito: descriptor de fichero (3+)
  - Error: -1

malloc():
  - Éxito: puntero válido
  - Error: NULL
```

Constantes Importantes

```
0 = stdin
1 = stdout
2 = stderr

O_RDONLY = leer
O_WRONLY = escribir
O_CREAT = crear si no existe
O_TRUNC = truncar (borrar contenido)
O_APPEND = añadir al final
```

📊 ESTADÍSTICAS DE PREGUNTAS

Por año:

- 2015-2016: 10 preguntas ⭐ ⭐ ⭐
- 2017-2018: 5 preguntas ⭐ ⭐
- 2018-2019: 20 preguntas ⭐ ⭐ ⭐
- 2019-2020: 15 preguntas ⭐ ⭐ ⭐
- 2020-2021: 20 preguntas ⭐ ⭐ ⭐
- 2021-2022: 20 preguntas ⭐ ⭐ ⭐
- 2022-2023: 20 preguntas ⭐ ⭐ ⭐

Total: 115 preguntas analizadas

🎓 PRÓXIMOS PASOS

Después de revisar este documento:

1. **Repasa el tema de memoria:** malloc, free, sizeof
2. **Practica con fork():** dibuja árboles de procesos
3. **Entiende syscalls:** read, write, open, close, exec
4. **Domina pipes:** tamaño, buffering, deadlock
5. **Memoriza retornos:** cada función devuelve algo diferente

📝 NOTAS IMPORTANTES

- ✅ Todos los ejemplos de código compilarían (sin warnings)
- ✅ Las explicaciones se basan en el temario oficial
- ✅ Los años de referencia son aproximados (2015-2023)
- ✅ Algunas preguntas pueden tener variaciones en cada examen

¡Ahora revisa los temas y pregunta sobre lo que no entiendas!

Para estudiar de manera efectiva, enfócate en:

- Entender EL POR QUÉ (no solo memorizar la respuesta)
- Trazar código paso a paso
- Dibujar árboles de procesos y memoria
- Compilar y ejecutar ejemplos reales