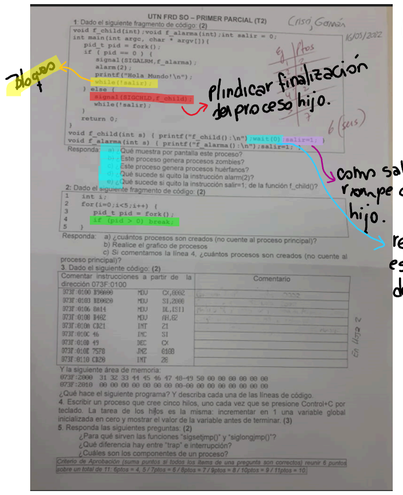
Parcial 16-05-2022

domingo, 21 de mayo de 2023 07:42 p. m.



① A) El proceso imprime "Hola Mundo", luego al ejecutarse la alarma imprime "f_alarma()"; por ultimo imprime "f_child():" al ejecutar la señal SIGCHLD

B) Esto no ocurrira ya que cuando se ejecuta la señal SIGCHLD el padre con la instrucción wait() recupera el estado del proceso zombie.

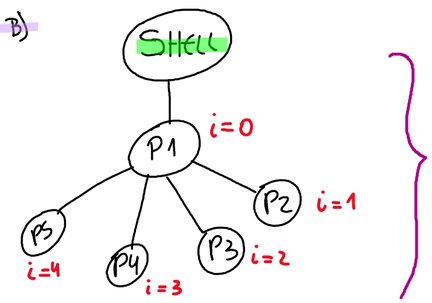
C) Esto no ocurre ya que nunca ocurrira que el proceso padre finalice antes que el hijo; debido a la función SIGCHLD.

D) Si quitamos la instrucción alarm(2) nunca se ejecutara el handle de la señal SIGALARM y no se cambiaria el valor de la variable salir; de esta manera quedará en un bucle infinito el hijo

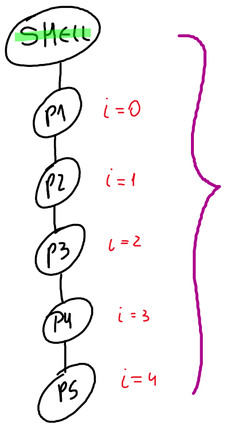
E) El padre quedará en un bucle infinito ya que no cambiaria el valor de la variable salir.

② A) El código crea 5 procesos

B)

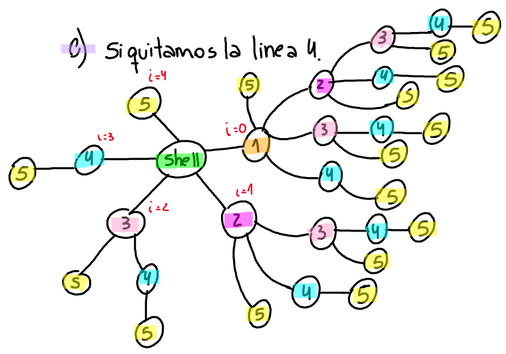


Si (pid == 0) ya que el padre nunca saldría del ciclo



el padre hace un fork y sale del ciclo; el hijo copia imágen del padre

C) Si quitamos la linea 4.



③

```
_ Mov  CX,0002   mover 0002 al registro C
_ Mov  SI,2000   mover el puntero a la dirección 2000
_ Mov  DL,[SI]   mover el contenido de si a la parte baja del registro D.
_ Mov  AH,02     mover a la parte alta del registro A, la funcion 02 que permite mostrar en pantalla
_ INT  21        ejecuto int 21 que ejecuta lo guardado en AH, mostrando el contenido de DL
_ INC  SI          incremento en 1 el puntero
_ DCC  CX          decremento CX.
_ JNZ  0108          salto a la instrucción 0108 si CX≠0
_ INT  20            Finalización del programa con INT 20.
```

02 → muestro contenido de DL

El programa muestra el Contenido de la posición 2000 y 8 posiciones mas

4) Proceso que cree cinco HILOS.
crea c/u presionando CONTROL+C ⇒ Como se debe reiniciar c/vez que apreto CTRL+C debo usar función pause() (*)
Incrementar en 1 una variable global (INICIALIZADA EN 0)
Mostrar el valor de la variable antes de terminar.

```
Void func_sigint(int),
Void Hilo1(int);
int i =0;
contador (VAR GLOBAL) = 0;

Int main(void) {
   pthread h1,
   signal (SIGINT, func_sigint)

 //ciclo creación hilo          → hilos(5)
    For(i=0, i<5, i++){
      Printf("Ctrl+C p/crear hilo"),
      pause(),  (*)
     }
      return 0; (p/ determinar si el proceso termino correctamente)
  }
     //func Sigint
   Void func_sigint(int status){
      pthread_create (&h1, NULL, (void)*Hilo1, NULL),
      pthread_join (hilo1, NULL);  (espera de la terminación del hilo)
    }

     //func.Hilo
   Void Hilo1 (int ){
   Contador++, → VAR
   Printf("%d", (Contador));
}
```

5) A). SIGSETJMP(): guarda el contexto actual de un programa para poder utilizarse en cualquier momento

↳ int sigsetjmp (sigjmp_buf var1, int savesigs)

determina si se deben guardar señales bloqueadas (0) no se guardaran. (!0) se "

SIGLONGJMP(): permite saltar a un pto de retorno establecido en sigsetjmp().

↳ void siglongjmp(sigjmp_buf var1, int val);
↳ retorno

B) Dif. Trap & interrupción

- TRAP ⇒ excepciones internas generadas por el programa en ejecución.
- INTERRUPCIÓN ⇒ eventos externos generados por dispositivos de hardware.

C) Componentes

- Programa ejecutable
- Datos asociados al programa
- Contexto de ejecución