Práctica 2 SOPER
Marcos Alonso y Daniel Cruz

pregunta del apartado e):
Es muy complicado implementar la concurrencia sin semáforos, y en muchos casos falla, ya que el semáforo es una forma de comunicarse entre procesos teniendo todos el mismo código. Sin el semáforo, muchas veces no da tiempo a que el candidato se proclame candidato (en el método que hemos realizado) en solitario, y mas de una vez se proclaman mas de 1 candidato. Además, en el fichero se escriben cosas de manera errónea de forma frecuente.

Explicación apartados:
a)
El programa se encarga de leer los argumentos, y si son erróneos muestra el funcionamiento con exit_use() y finaliza.
En un bucle del tamaño del número de procesos, el proceso padre va creando hijos que manda a la función votantes donde esperan sigusr1 de forma no activa con sigsuspend. El padre hace waitpid con WNOHANG para recoger al hijo cuando acabe pero para poder seguir con el programa. Cuando ha lanzado todos, a su vez ha ido escribiendo sus pids en un fichero .pids, y cuando acaba el bucle, les envía a estos pids (también almacenados en un array por eficiencia) les envía la señal SIGUSR1 para indicar que se comienza.
Se establece un handler para cuando llega SIGINT o SIGALRM, que acaba con el programa liberando los recursos.
b)
Esperan a sigusr1 con sigsuspend los votantes, y cuando sigusr1 llega, en el manejador se elige el candidato que pone la variable global candi a 1. El candidato crea el archivo de votación y envía sigusr2 a los votantes para indicar que está preparado, usando los pids de .pids. Los votantes escriben o Y o N aleatoriamente para votar, uno a uno gestionado por un semáforo. El candidato va comprobando cada 1ms, y cuando detecta todos los votos, muestra los resultados en la función print_votes(). El candidato deja de serlo y vuelve a enviar sigusr1 a los votantes para elegir uno nuevo. Cuando se recibe SIGINT se envía SIGTERM a los votantes, que terminan, pero sus recursos ya son liberados en el manejador de SIGINT.
c)
Las señales son bloquedas con sigprocmask, y las esperas son realizadas con sigsuspend, como el enunciado indica.
d)
se establece con alarm() pasándole como argumento argv[2] pasado a int, los segundos hasta que se manda SIGALRM. El manejador de SIGALRM es handler, que libera los recursos, envía SIGTERM a los hijos y termina el programa.
f)
Se utilizan semáforos en:
Cuando reciben sigusr1, el primero hace down para crear el archivo temporal .candi y establecerse como candidato poniendo la variable global a 1. Antes de acabar el programa hace up.
Cuando están escribiendo el voto, hacen un down cuando va a escribir y un up cuando terminan.

Naturalmente, los dos semáforos tienen tamaño 1 inicial, para que cuando uno haga sig wait, ya esté la cuenta en 0 y no puedan entrar más en esa zona.

PRUEBAS REALIZADAS:

Se ha probado con distintos números de procesos y el programa a partir de 20 procesos empieza a fallar, lo vemos un límite razonable.
Se ha probado con argumentos incorrectos y los errores están controlados.