UNIVERSITÀ DEGLI STUDI DI BERGAMO

Corso di laurea in ingegneria informatica
Appello straordinario di sistemi operativi – 5 aprile 2006

Cognome_____Nome_____Matricola_____

2.
Si consideri il programma seguente, sapendo che il file "testo" esiste ma è **vuoto**:

```
00  int main ( ) {
01     int i, pid, fd;
02     int status = 1;
03     char A = 'A';
04     char B = 'B';
05     char C = 'C';
06     fd = open ("testo", O_WRONLY);
07     for (i = 1; i <= 2; i++) {
08        pid = fork ( );
09        if (pid == 0) {
10           if (i == 1) {
11              write (fd, &A, 1);
12           } else {
13              write (fd, &B, 1);
14              exit (0);
15           } /* if */
16        } else if (i == 2) {
17           wait (&status);
18        } /* if */
19     } /* for */
20     if (getpid ( ) > 80) {
21        write (fd, &C, 1);
22     } /* if */
23     exit (0);
24  } /* main */
```

Rispondere alle domande seguenti, sapendo che il processo padre ha PID 80, tutti i PID successivi sono liberi e il sistema operativo li assegna andando in successione, e che la schedulazione della CPU è tale da far eseguire a ciascun processo presente nel sistema un'istruzione per volta (attenzione: la verifica delle condizioni in if, else, for non sono contate come istruzioni).

- **a.** Ci sono 4 processi in totale: come sono organizzati gerarchicamente, cioè chi crea chi ?
- **b.** Quale stringa contiene il file testo quando tutti i processi sono terminati ?
- **c.** Quante volte viene eseguita la fork alla riga 08 ?
- **d.** Quante volte viene eseguita la write alla riga 13 ?
- **e.** Quante volte viene eseguita la wait alla riga 17 ?

Soluzione:

Ecco in sequenza le istruzioni che vengono eseguite con i relativi esecutori. Gli identificatori dei processi sono in **grassetto**, le etichette numeriche delle istruzioni sono in *corsivo*.

1. **80** esegue *08* e crea **81** (pid vale 81 in **80**, e vale 0 in **81**; in entrambi i vale 1)
2. **81** esegue *11* e scrive 'A' sul file (file: "A")
3. **80** esegue *08* e crea **82** (pid vale 82 in **80**, e vale 0 in **82**; in entrambi i vale 2)
4. **81** esegue *08* e crea **83** (pid vale 83 in **81**, e vale 0 in **83**; in entrambi i vale 2)
5. **82** esegue *13* e scrive 'B' sul file (file: "AB")
6. **83** esegue *13* e scrive 'B' sul file (file: "ABB")
7. **80** esegue *17* e si pone in attesa della terminazione di un figlio
8. **81** esegue *17* e si pone in attesa della terminazione di un figlio
9. **82** esegue *14* e termina
10. **83** esegue *14* e termina
11. **80**, a cui getpid() restituisce 80, esegue *23* e termina
12. **81**, a cui getpid() restituisce 81, esegue *21*
13. **81** esegue *23* e termina

Alla luce di questa sequenza di eventi possiamo rispondere alle domane:

- **a.** 80 crea 81 e 82; 81 crea 83
- **b.** "ABBC"
- **c.** 3 volte
- **d.** 2 volte
- **e.** 2 volte