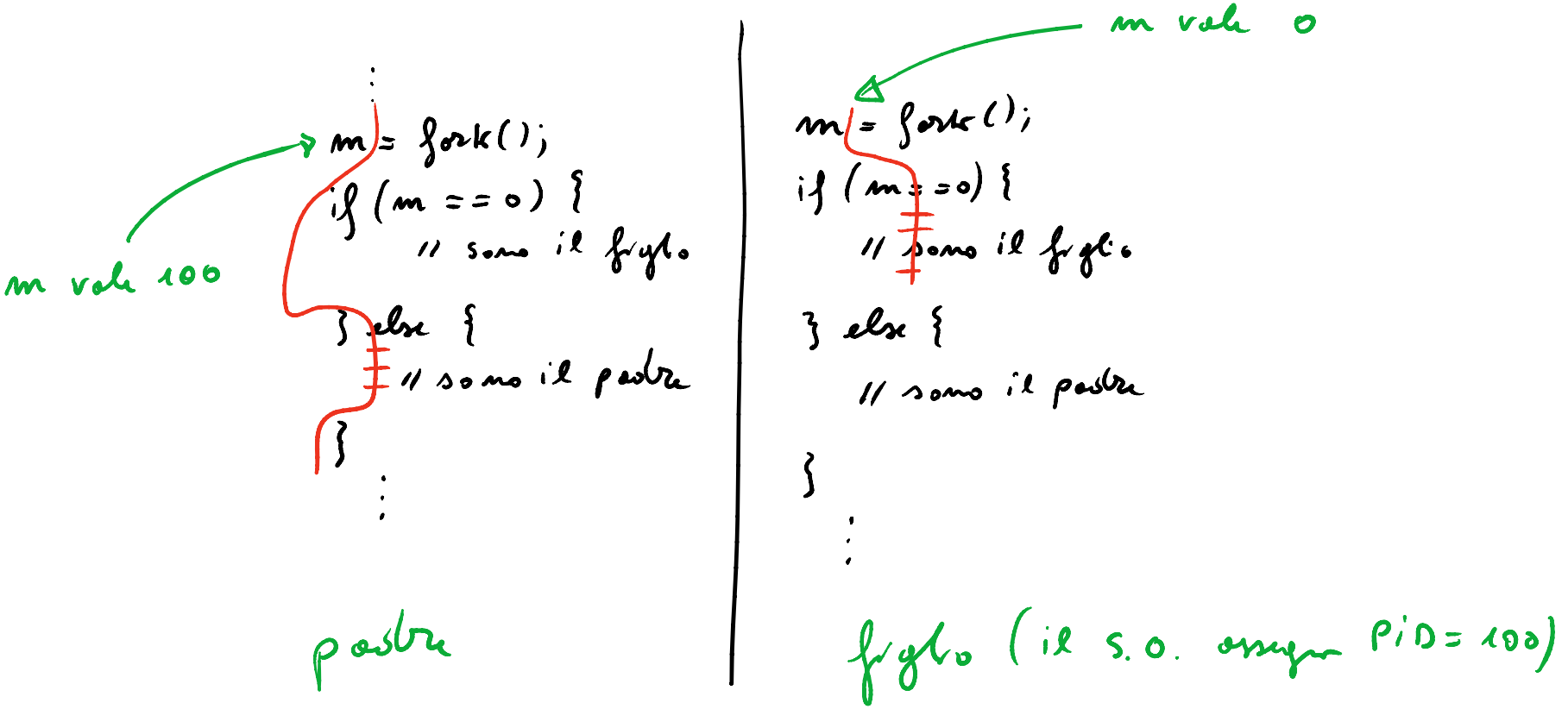
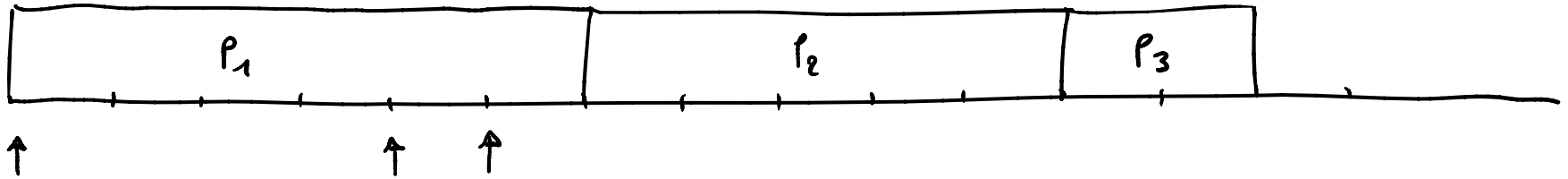

m vale 100
m = fork();
if (m == 0) {
// sono il figlio
} else {
// sono il padre
}
padre
m vale 0
m = fork();
if (m == 0) {
// sono il figlio
} else {
// sono il padre
}
figlio (il S.O. assegna PID = 100)

First Come - First Served

P_{RX}	tempo d'arrivo	durata
P_1	0	6
P_2	4	5
P_3	5	2



tempo di attesa in coda

per $P_1 = 0$

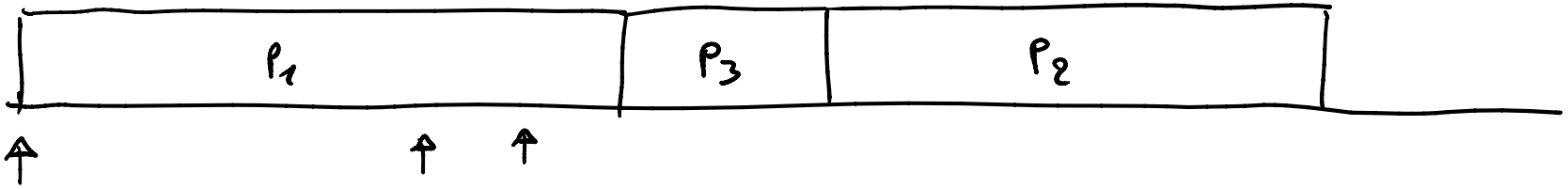
per $P_2 = 6 - 4 = 2$

per $P_3 = 11 - 5 = 6$

tempo medio di attesa in coda $\frac{0 + 2 + 6}{3} = \frac{8}{3}$

Shortest Job First

scelgo il processo pronto con durata minore



tempo di attesa in coda

per $P_1 = 0$

per $P_2 = 8 - 4 = 4$

per $P_3 = 6 - 5 = 1$

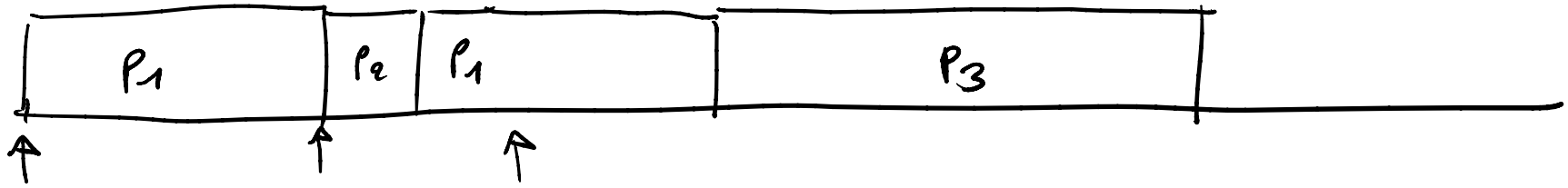
tempo medio d'attesa in coda

$$\frac{0 + 4 + 1}{3} = \frac{5}{3}$$

Shortest Remaining Job First

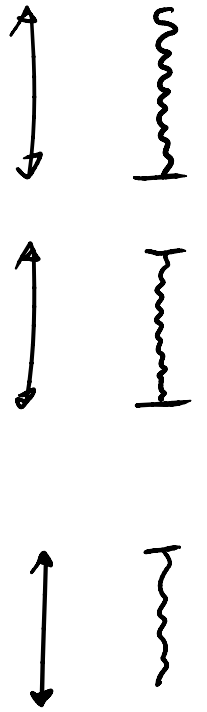
Proc.	tempo di arrivo	durata
P_1	0	6
P_2	3	1
P_3	5	5

Quando arriva in coda pronti un nuovo processo, controllo se la sua durata è minore della durata residua di quello attualmente in esecuzione (e nel caso eseguo commutazione).



Problema: la durata dei processi?

- chiedo all'utente
- provo a stimarla



esecuzione

I/O

esecuzione

I/O

esecuzione

$$S_{n+1} = \alpha \cdot d_n + (1-\alpha) \cdot S_n$$

S_n = è la mia stima n-esima

d_n = l'effettiva durata del burst di esecuzione n-esimo

α = un coefficiente che consente di dare più o meno importanza alla storia recente (d_n) rispetto a quella passata (S_n).

Round Robin

P_{RUN}	tempo di arrivo	durata
P_1	0	6
P_2	4	5
P_3	5	2