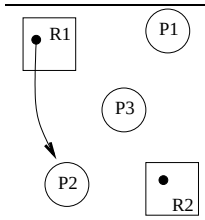
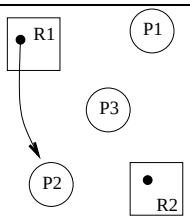
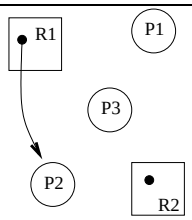
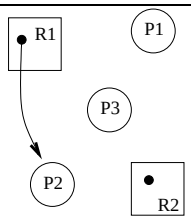
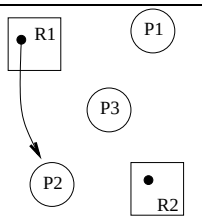
Esercizio 1

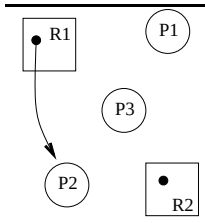
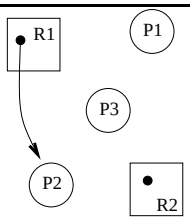
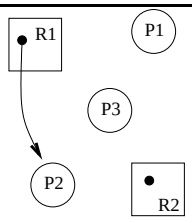
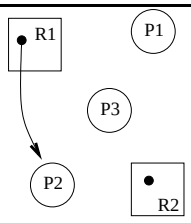
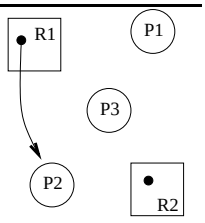
Si consideri un sistema con una risorsa di tipo R1 e una risorsa di tipo R2. Sono presenti tre processi: P1, P2 e P3 e il processo P2 ha già assegnata una risorsa di tipo R1. Completare con le opportune frecce il grafo di assegnazione dopo ogni richiesta nei tre casi sotto menzionati:

1. Nessuna gestione del deadlock da parte del sistema operativo

				
P1 chiede R1	P3 chiede R2	P1 chiede R2	P3 chiede R1	P2 rilascia R1
Spiegazione:	Spiegazione:	Spiegazione:	Spiegazione:	Spiegazione:

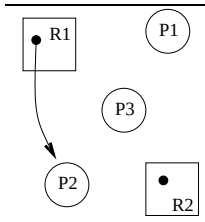
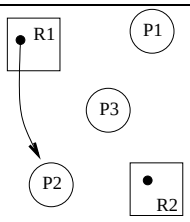
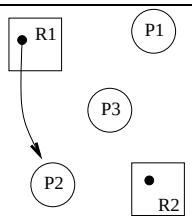
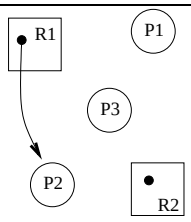
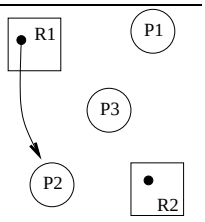
Si verifica una situazione di deadlock? Perchè?

2. Gestione gerarchica delle risorse: R1 < R2

				
P1 chiede R1	P3 chiede R2	P1 chiede R2	P3 chiede R1	P2 rilascia R1
Spiegazione:	Spiegazione:	Spiegazione:	Spiegazione:	Spiegazione:

Si verifica una situazione di deadlock? Perchè?

3. algoritmo con grafo di assegnazione delle risorse e archi di reclamo (le richieste corrispondono a quelle massime dichiarate dai processi)

				
P1 chiede R1	P3 chiede R2	P1 chiede R2	P3 chiede R1	P2 rilascia R1
Spiegazione:	Spiegazione:	Spiegazione:	Spiegazione:	Spiegazione:

Si verifica una situazione di deadlock? Perchè?

Esercizio 2

Considerare una variante del problema dei filosofi in cui, oltre alle due bacchette, ogni filosofo deve attendere che il proprio piatto sia pieno. A tale scopo ogni filosofo ha un cameriere personale che provvede a riempire il piatto ogni volta che esso si svuota. I filosofi, quindi, attendono le bacchette sinistra e destra ed il piatto pieno, mangiano svuotando completamente il piatto, appoggiano le bacchette e tornano a pensare. Aggiungere le necessarie sincronizzazioni utilizzando i semafori ed indicandone il valore di inizializzazione. Spiegare il funzionamento della soluzione indicata discutendo eventuali possibilita' di stallo.

```
filosofo(i) {

  while(true) {

    <pensa>



    <mangia>



  }
}
```

```
cameriere(i) {

  while(true) {



     <riempi piatto i-esimo>



  }

}
```

Esercizio 3

Cosa si intende per *sezione critica*? Perchè le soluzioni software al problema della sezione critica sono insoddisfacenti? Discutere brevemente le soluzioni hardware e quelle a livello di Sistema Operativo.

Esercizio 4

Si consideri un seggio elettorale con tre cabine per il voto. In ogni cabina non è consentita la presenza simultanea di più elettori. Appena una delle tre cabine si libera (`cabina[i]==true`) il primo elettore in attesa può accedervi. Il processo `elettore` esegue il seguente codice:

```
process elettore{
    int i;
    i=seggio.attendi_cabina();
    <vota>
    seggio.libera_cabina(i);
}

monitor seggio {
    boolean cabina[3]={true,true,true};
    condition attendi;

    int attendi_cabina() {

    }
```

SPIEGAZIONE SOLUZIONE:

```
    libera_cabina(int j) {

    }
```

1. Completare il codice di `attendi_cabina()` e `libera_cabina()` (direttamente negli appositi spazi vuoti) spiegando la soluzione proposta. Ipotizzare che le code di attesa sulle variabili condition siano gestite con politica *Fist Come First Serve*.
2. Discutere la possibilità di attesa illimitata (*starvation*) nella soluzione proposta e, nel caso si riscontri tale problema, proporre possibili soluzioni.

Esercizio 5

Cosa si intende per scheduling batch con prelazione (preemptive) e scheduling time-sharing? Evidenziare differenze vantaggi e svantaggi.

Esercizio 6

In un sistema, sono pronti per essere eseguiti i 3 processi P1, P2, P3. L'algoritmo di scheduling è Round-Robin con quanto uguale a 1 e con due priorità statiche: Alta e Bassa. P1 e P2 sono a priorità Alta mentre P3 è priorità Bassa. I tre processi sono tutti CPU-bound ed utilizzeranno la CPU per un tempo molto elevato (molto maggiore della dimensione della tabella qui sotto).

1. Illustrare lo scheduling (scrivere CPU per il processo in esecuzione P_A, P_B per i processi pronti ad Alta e Bassa priorità)

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
P1																				
P2																				
P3																				

2. Che tipo di problema si presenta nell'esecuzione riportata sopra? Proporre una soluzione a tale problema ed illustrarla nello schema qui sotto

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
P1																				
P2																				
P3																				

Esercizio 7

Si consideri un processo che fa riferimento a 4 pagine virtuali nel seguente ordine:

$$1, 2, 3, 4, 2, 3, 1, 3, 2, 3, 4, 2, 3$$

Si consideri una memoria fisica (inizialmente vuota) di 3 frame e si mostri il funzionamento degli algoritmi seguenti:

1. sostituzione First In First Out (FIFO),

Spiegare brevemente:

	1	2	3	4	2	3	1	3	2	3	4	2	3
Frame 1													
Frame 2													
Frame 3													
Page Fault													

2. LRU (Least Recently Used),

Spiegare brevemente:

	1	2	3	4	2	3	1	3	2	3	4	2	3
Frame 1													
Frame 2													
Frame 3													
Page Fault													

3. Sostituzione "seconda-chance" (clock): indicare puntatore e reference-bit

Spiegare brevemente:

	1	2	3	4	2	3	1	3	2	3	4	2	3
Frame 1													
Frame 2													
Frame 3													
Page Fault													