Ricordarsi di mettere il proprio nome, cognome, e numero di matricola in tutti i fogli. Motivare sempre le risposte date. Non e' necessario dare risposte molto lunghe, ma e' importante rispondere in modo motivato ed esauriente alle domande poste (in altre parole, molto meglio una frase in piu' che una in meno).
Per avere la sufficienza, e' **necessario svolgere ognuno dei primi 4 esercizi.**

Non sono ammesse macchinette calcolatrici o altre macchine elettroniche; non e' consentito uso di appunti o libri.

Malacopia: consegnare, se necessario **solo** gli esercizi che devono essere corretti (non riportati in bella copia); barrare quindi gli altri

---

Esercizio 1 1. Riportate il diagramma di stato della vita di un processo.

2. In quali transizioni da uno stato all'altro deve intervenire il sistema operativo?

Risposta(Sketch) In tutte, in quanto e' il SO che sposta i processi (i loro PCB) da uno stato all'altro.

Esercizio 2 Quali sono i principali vantaggi e svantaggi della paginazione della memoria?

Esercizio 3 In un file system, come funziona l'allocazione di file di tipo "concatenato"? Quali sono gli inconvenienti principali?

Esiste un problema di frammentazione interna o esterna per questo tipo di allocazione?

Esercizio 4 []

1. Considerate i 2 processi sotto

```
P1 =                  P2 =
    print(A);              print(B)
    print(D)               print(C)
                           print(E)
```

   Agite, se necessario, sul codice, inserendo opportune operazioni su semaforo, in modo che nell'output di questa esecuzione concorrente sia la stringa ABCDE.

2. Assumete ora che le stampe sopra siano include dentro un loop:

```
P1 = loop(             P2 =loop(
    print(A);              print(B)
    print(D) )             print(C)
                           print(E)  )
```

   Occorre apportare dei cambiamenti alla soluzione sopra in modo da avere come stringa stampata la stringa infinita ABCDEABCDEABCDE... ? Se si, indicare i cambiamenti.

Risposta(Sketch)

```
P1 =                    P2 =
                             P(S)
    print(A);                print(B)
      V(S); P(T)
    print(D)                 print(C)
      V(R)                   V(T); P(R)
                             print(E)
```

con tutti i semafori a 0

Stringa infinita: si aggiunge un semaforo U inizializzato a 0, mettendo

```
P(U)
```

come ultima istruzione del loop sul primo processo e

```
V(U)
```

come ultima istruzione del loop sul secondo.

Esercizio 5 Si consideri uno scheduler che riceve 4 job A,B,C,D con le seguenti caratteristiche:

```
Job durata     tempo di inizio
A    8              0
B    3              2
C    1              4
D    2              7
```

Riportare il diagramma Gantt relativo ad una esecuzione con round robin e con quanto di tempo pari a 3. Qual'e' il waiting time medio?

Risposta(Sketch)

```
0 [A] 3 [B] 6 [A] 9 [C] 10 [D] 12 [A] 14
```

Quindi waiting time A : 14-8 = 6
Waiting time B : 6-3-2 = 1
Waiting time C : 10-1-4 = 5
Waiting time D : 12 - 7-2 = 3
Waiting time medio: 6+1 + 5+ 3 /4 = 15/4

Esercizio 6 Si consideri il problema dei lettori e scrittori visto a lezione, dove i codici del generico scrittore e del generico lettore sono riportati qui di seguito. Inserite le operazioni di decremento/incremento su semaforo mancanti necessarie per il corretto funzionamento del sistema, indicando anche il semaforo mancante. Ricordarsi dei valori di inizializzazione. Come per tutte le risposte, motivate brevemente anche le aggiunte fatte.

```
semafori e variabili condivise necessarie
semaphore write = 1;
int numlettori = 0;

scrittore {
...Esegui la scrittura del file  ...
```

```
 }

lettore {

numlettori++;
if numlettori == 1   ;
... leggi il file ...
numlettori--;
if numlettori == 0   ;
```

Risposta(Sketch)

```
semafori e variabili condivise necessarie

semaphore mutex = 1;
semaphore write = 1;
int numlettori = 0;

scrittore {
P(scrivi);
...Esegui la scrittura del file  ...
V(scrivi) }

lettore {
P(mutex);
numlettori++;
if numlettori == 1 P(scrivi);
V(mutex);
... leggi il file ...
P(mutex);
numlettori--;
if numlettori == 0 V(scrivi);
V(mutex);
```

Esercizio 7 Cosa e' la starvation? La soluzione del problema dei lettori e scrittori sopra garantisce l'assenza di starvation?

Risposta(Sketch) Starvation: vedi note corso.

No, infatti un qualsiasi processo scrittore potrebbe dover attendere all'infinito senza riuscire a ad entrare in sezione critica. Al contrario i processi lettori sono liberi da starvation.

Esercizio 8 In un sistema con paginazione della memoria centrale, un indirizzo fisico e' scritto su 37 bit, l'offset piu' grande all'interno di una pagina e' pari a 11 1111 1111, e la tabella delle pagine piu' grande del sistema occupa 32 megabyte.

1. Quanti frame ci sono nel sistema?
2. Quante entry ci sono nella tabella di pagine? (potete assumere che ci sia solo il bit di validita' e nessun altro bit di controllo)

3. Quanto e' grande lo spazio di indirizzamento logico del sistema?
4. E' necessaria la paginazione a piu' livelli della tabella di pagine? Se si, su quanti livelli?
5. E' necessaria la memoria virtuale?

Risposta(Sketch)

1. Nell'indirizzo fisico, 37-10= 27 bit sono usati per numero di frame. Quindi ci sono 2^{27} frame
2. Una entry prende 4 B (che servono per i 27 bit del num di frame e poi il resto dei bit e' inutilizzato o usato per bit validita') Quindi ci sono $2^{25}/2^2 = 2^{23}$ entry
3. Spazio di indirizzamento logico: $2^{23} \times 2^{10} = 2^{33}$
4. si (tab pagine piu' grande del singolo frame) Siccome $2^{25}/2^{10} = 2^{15}$, la TP di secondo livello avra' 2^{15} entry, e siccome una entry prende 4B, abbiamo una taglia di 2^{17} B per tale TP di secondo livello, che a sua volta non e' quindi contenibile in un frame. Avremo quindi bisogno di un terzo di livello di TP (al terzo livello servono 2^7 entry, quindi 2^9 B e questi stanno dentro un singolo frame)
5. si: Lo spazio logico non e' piu' grande dello spazio fisico (2^{33} contro 2^{37}); quindi strettamente parlando non e' necessaria la memoria virtuale. Ma comunque le grandezze non sono molto diverse (tra l'altro c'e' anche da tenere presente che parte dello spazio fisico dovra' essere dedicato al SO), e quindi una memoria virtuale e' altamente consigliabile per il discorso multiprogrammazione.

Esercizio 9

1. Nella cifratura a chiave simmetrica, perche' una permutazione sull'alfabeto non e' considerata una buona chiave?
2. Usare le proprieta' dell'aritmetica modulo per calcolare in modo semplice $(5^2 \times 8^3) \bmod 7$, e anche $11^6 \bmod 7$

Risposta(Sketch)

1. metodi statistici consentono di risalire alla chiave facilmente, per messaggi abbastanza lunghi.
2. $$\begin{aligned} & (5^2 \times 8^3) \bmod 7 \\ = \; & ((5^2) \bmod 7 \times (8^3) \bmod 7) \bmod 7 \\ = \; & ((5^2) \bmod 7 \times (8^3) \bmod 7) \bmod 7 \\ = \; & (25 \bmod 7 \times ((8 \bmod 7)^3) \bmod 7) \bmod 7 \\ = \; & (4 \times 1) \bmod 7 = 4 \end{aligned}$$

   Per $11^6 \bmod 7$: Con simili proprieta' ci si riconduce a calcolare $4^6 \bmod 7$.

   $$4^6 \bmod 7 = (4^2 \bmod 7 \times 4^4 \bmod 7) \bmod 7$$

   Similmente, siccome $4^2 \bmod 7 = 2$, $4^4 \bmod 7 = 4$

   Quindi abbiamo $2 \times 4 \bmod 7 = 1$.

Notare che $11^6 \bmod 7 = 1$ si poteva anche dedurre direttamente dal teorema di Fermat (secondo il quale, per p primo e n non multiplo di p vale $n^{p-1} \bmod p = 1$).