SOLUZIONI DEI PROBLEMI DEL CAPITOLO 4

1. **Soluzione:**
   Adottando lo schema best-fit la partizione libera utilizzata per allocare un segmento di dimensione pari a 2560 byte è quella le cui dimensioni siano le più piccole tra tutte quelle di dimensioni maggiori o uguali a quelle richieste. Nel nostro caso ciò corrisponde alla partizione libera caratterizzata dai parametri: (I:41472-D:3072). Pertanto il segmento verrebbe allocato in una partizione il cui indirizzo iniziale corrisponderebbe a 41472 e di dimensioni pari a quelle del segmento stesso: 2560. Dopo tale allocazione rimarrebbe ancora disponibile una partizione caratterizzata dai seguenti parametri: (I:44032-D:512) corrispondente alla parte porzione non utilizzata della partizione originaria.
   Adottando lo schema first-fit la partizione libera utilizzata per allocare un segmento di dimensione pari a 2560 byte è quella di indirizzo più basso tra tutte quelle di dimensioni maggiori o uguali a quelle richieste. Nel nostro caso ciò corrisponde alla partizione libera caratterizzata dai parametri: (I:1024-D:8192). Pertanto il segmento verrebbe allocato in una partizione il cui indirizzo iniziale corrisponderebbe a 1024 e di dimensioni pari a quelle del segmento stesso: 2560. Dopo tale allocazione rimarrebbe ancora disponibile una partizione caratterizzata dai seguenti parametri: (I:3584-D:5632) corrispondente alla parte porzione non utilizzata della partizione originaria.

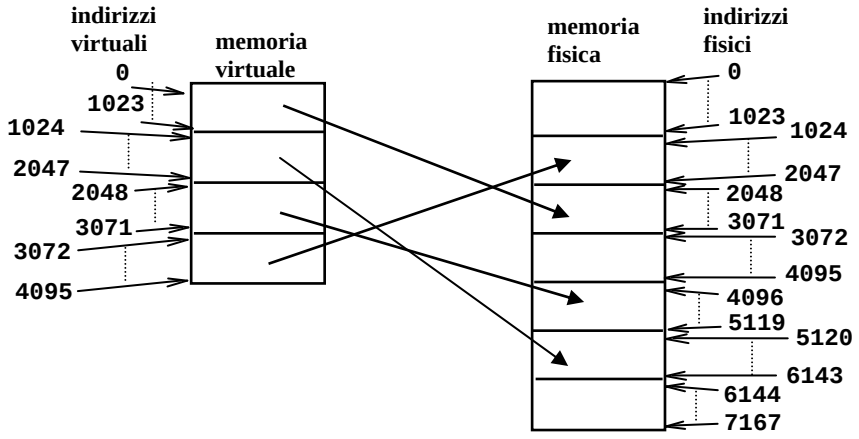
2. **Soluzione:**
   Di seguito viene riportata la figura 4.23 a cui il problema si riferisce:
   Nell'esempio di figura, la dimensione d delle pagine è pari a 2^{10}=1024 byte. Come si può vedere, l'indirizzo virtuale 2054 appartiene alla terza pagina logica, quella che inizia con l'indirizzo 2048. Più in particolare, indicando con p l'indice della pagina logica a cui l'indirizzo virtuale appartiene e con o lo scostamento di tale indirizzo all'interno della sua pagina logica, abbiamo:

   $$p = \text{quoziente intero di } 2054/1024 = 2, \quad o = \text{resto di } 2054/1024 = 6$$

   poiché la pagina logica in questione è allocata nella pagina fisica che inizia all'indirizzo 4096, l'indirizzo fisico corrispondente all'indirizzo virtuale 2054 è quindi 4096+6 = 4102
   Analogamente, possiamo ricavare gli indirizzi fisici corrispondenti agli altri indirizzi virtuali.
   Nella seguente tabella viene riportata la corrispondenza tra gli indirizzi virtuali indicati dal problema e i corrispondenti indirizzi fisici:

indirizzo virtuale	indirizzo fisico
2054	4102
980	3028
3126	1078
4098	errore
3060	5108

Il caso dell'indirizzo virtuale 4098 corrisponde ad un errore poiché il massimo indirizzo virtuale del processo il cui spazio virtuale è indicato nella figura corrisponde a 4095.



3. <u>Soluzione</u>:

La stringa dei riferimenti di pagina relativa agli indirizzi virtuali indicati nel precedente problema (2054, 980, 3126, 4098, 3060) è la segiente:

2, 0, 3, 4, 3

4. <u>Soluzione</u>:

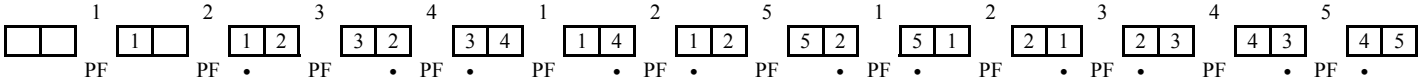
Poiché si suppone assenza di prepaging, la prima volta che viene fatto riferimento a ciascuna delle 5 pagine virtuali, questo rifrimento genera un page fault per caricare la pagina virtuale richiesta. Ma, poiché il numero delle frame di memoria fisica coincide col numero delle pagine virtuali a cui il processo si riferisce, nessuna pagina virtuale, una volta caricata, verrà rimpiazzata. Quindi il numero esatto dei page fault generati è 5 e tale valore è ovviamente indipendente dall'algoritmo di rimpiazzamento utilizzato.

5. <u>Soluzione</u>:

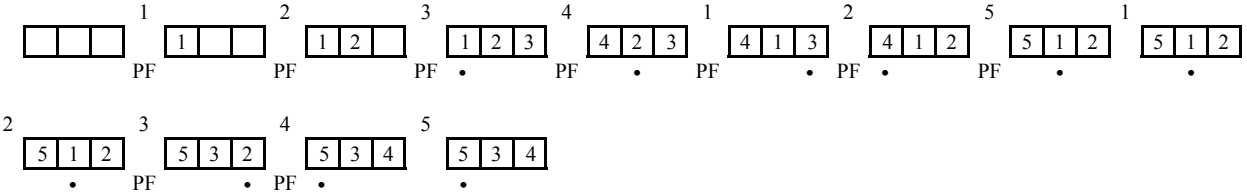
<u>a)</u>:

Nel caso di una sola frame di memoria (m=1), ogni riferimento ad una pagina virtuale diversa da quella caricata nella frame genera un page fault e ciò indipendentemente dall'algoritmo di rimpiazzamento utilizzato. Infatti, con una sola frame ogni nuova pagina riferita genera un un page fault che deve necessariamente rimpiazzare l'unica frame esistente. Quindi, in assenza di prepaging ciascuno dei 12 riferimenti della page-reference string genera un page fault.

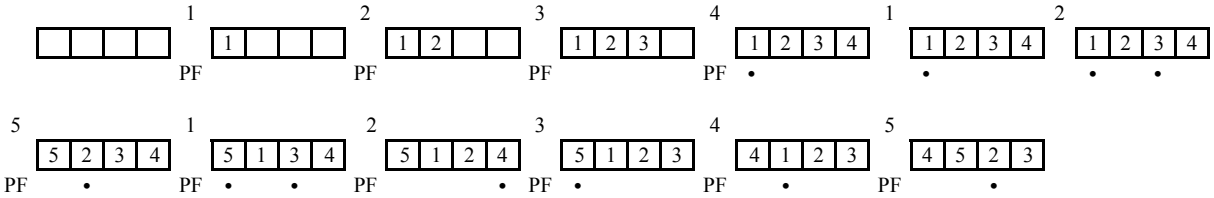
Anche nel caso di 2 frame (m=2) si ha lo stesso numero di page fault poiché ogni nuovo riferimento genera un page fault come viene messo in evidenza nella figura seguente dove, nella prima riga vengono riportate le pagine logiche riferite (nello stesso ordine in cui le pagine compaiono nella page reference string,) nella riga intermedia sono indicate le due frame di memoria prima e dopo ogni riferimento (e dove i rimpiazzamenti vengono effettuati mediante l'algoritmo FIFO) e nella terza riga viene indicato, mediante l'acronimo PF, se il corrispondente riferimento genera un page fault. Il puntino nero sotto una casella denota la frame che in base all'algoritmo di rimpiazzamento sarà scelta come vittima al prossimo rimpiazzamento di pagina. Quindi, anche in questo caso si verificano 12 page fault.



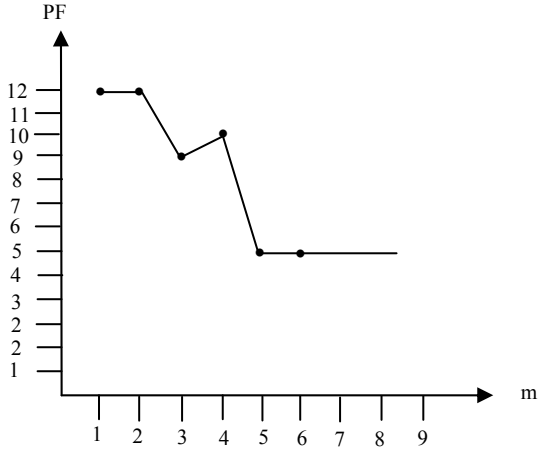
Nel caso di tre frame (m=3) il numero di page fault è 9 come evidenziato dalla figura successiva.



Nel caso di 4 frame (m=4) si hanno 10 page fault come evidenziato nella figura seguente.



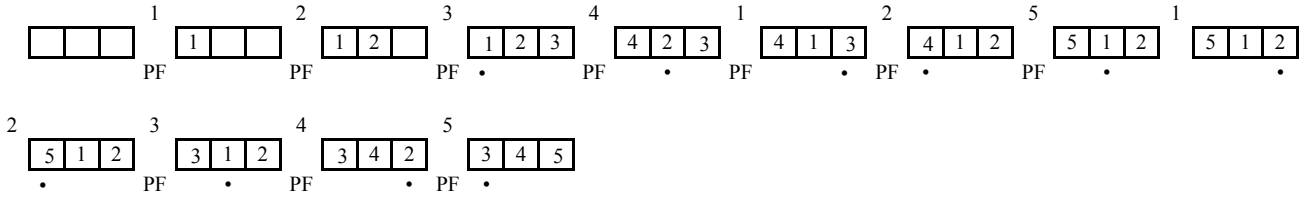
Nel caso di 5 frame (m=5), come già indicato nella soluzione del precedente problema, essendo il numero totale di pagine virtuali riferite uguale al numero di frame tale numero coincide anche col numero di page fault generati. Inoltre, poiché nessun page fault in questo caso può causare un rimpiazzamento di pagine, tale numero è indipendente dall'algoritmo di rimpiazzamento.
Infine, nel caso di 6 frame (m=6), il numero di page fault è a maggior ragione ancora uguale a 5, numero di pagine virtuali riferite. Come nel caso precedente, nessun page fault può causare rimpiazzamenti avendo disponibili più frame che pagine da caricare.
Nella figura che segue viene riportato il grafico del numero di page fault generati in funzione del numero m di fame disponibili:



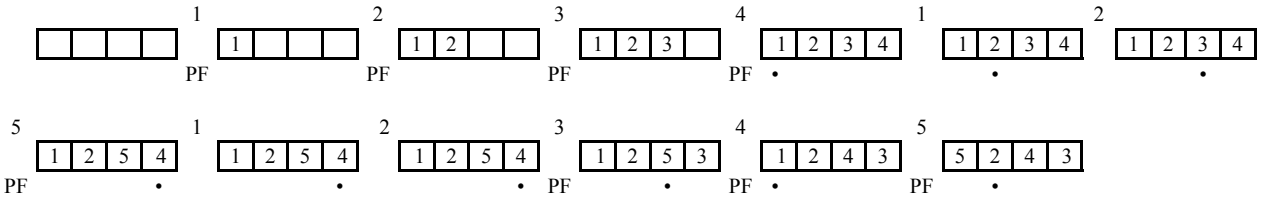
Come si può notare dal grafico, si evidenzia un'anomalia, nota come anomalia di Belady, corrispondente al fatto che all'aumentare del numero di frame disponibili, in particolare nel passare da 3 a 4 frame disponibili, il numero di page fault aumenta contrariamente a quanto ci dovremmo aspettare. Tale anomalia si può manifestare in presenza di particolari page-reference string quando l'algoritmo di rimpiazzamento utilizzato, come ad esempio l'algoritmo FIFO, non

appartiene alla cosiddetta categoria degli algoritmi a stack. Un algoritmo appartiene a tale categoria se gode della seguente proprietà: "indichiamo con $S_n(t)$ l'insieme delle pagine virtuali che, durante l'esecuzione di un programma e in base all'algoritmo di rimpiazzamento, sono presenti in memoria all'istante t avendo disponibili n frame; allora durante la stessa esecuzione e allo stesso istante t in presenza di un numero di frame disponibili m<n, l'algoritmo di rimpiazzamento usato è a stack se garantisce che $S_m(t)$ sia un sottoinsieme di $S_n(t)$".
Dalle figure precedenti si può facilmente verificare che nel caso di m=4, dopo che viene riferita per la prima volta la pagina virtuale 5, le pagine virtuali allocate in memoria sono: 5, 2, 3 e 4. Mentre nel caso di m=3, allo stesso istante sono presenti in memoria le pagine virtuali 5, 1, e 2. Tale insieme, contenendo la pagina 1 non è un sottoinsieme del precedente. Ciò conferma che l'algoritmo FIFO non è a stack.

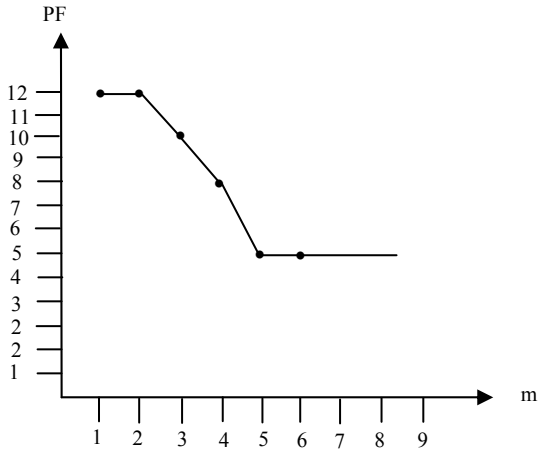
b):
Come già messo in evidenza precedentemente, se m=1 il numero di page fault non dipende dall'algoritmo di rimpiazzamento per cui tale numero resta uguale a 12 come nel caso precedente.
Anche nel caso di m=2 il numero di page fault è uguale a 12 come nell'esempio precedente. Infatti, la precedente figura vista per l'algoritmo FIFO che, nel caso di due sole frame, illustra come le pagine virtuali vengono caricate e quando la loro richiesta genera page fault, resta del tutto inalterata anche per l'algoritmo LRU.
Nel caso di m=3 abbiamo adesso 10 page fault, come posiamo verificare dalla seguente figura



Infine, nel caso di m=4 si verificano 8 page fault come indicato nella figura seguente.



Come già messo in evidenza precedentemente nei casi in cui m≥5 il numero di page fault resta uguale a 5 indipendentemente dall'algoritmo di rimpiazzamento.
Possiamo quindi riportare nel grafico seguente l'andamento dei page fault in funzione di m anche per l'algoritmo LRU. Da tale grafico risulta evidente che, in questo caso, non si manifesta l'anomalia di Belady.



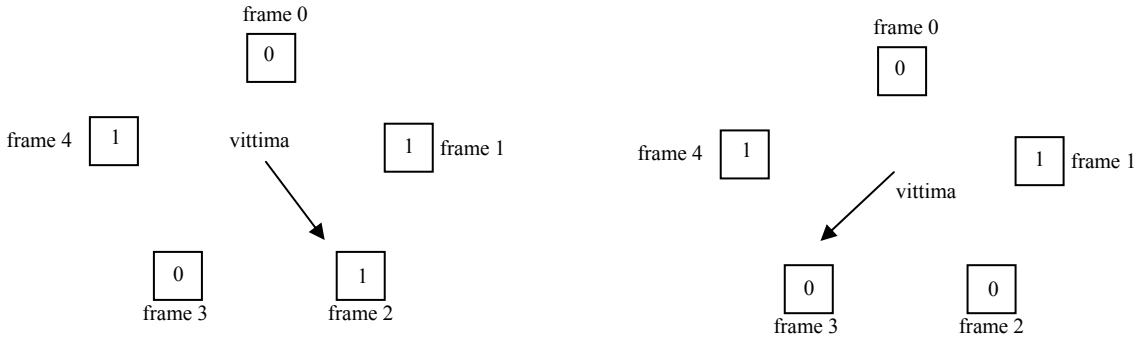
Infatti, l'algoritmo LRU gode della proprietà degli algoritmi a stack come risulta evidente sia dalle pecedenti figure sia perché, per definizione di LRU, nel caso di n frame in ogni istante sono presenti in memoria le n pagine virtuali più recentemente usate. Se invece di n frame fossero disponibili soltanto m<n frame, in esse sarebbero presenti le m pagine più recentemente usate che, ovviamente, sono un sottoinsieme delle n più recentemente usate.

6. <u>Soluzione</u>:

a):
Nel caso di algoritmo FIFO verrà rimpiazzata la pagina 3 cha da più tempo è in memoria, come si può desumere dai dati riportati nella seconda colonna della tabella.

b):
Nel caso di algoritmo LRU verrà rimpiazzata la pagina 1 che è quella meno recentemente usata, come si può desumere dai dati riportati nella terza colonna della tabella.

<u>Risposta c)</u>:
Nel caso di algoritmo *second-chance* verrà rimpiazzata la pagina 3 come si può desumere dai valori dei bit di uso riportati nella quarta colonna della tabella. Infatti, facendo riferimento alla figura 4.32 del libro che descrive visivamente il comportamento dell'algoritmo *second-chance*, possiamo riportare di seguito un'analoga figura relativa ai dati del problema e, indicando anche adesso nella parte a) della figura la configurazione dei bit di uso all'inizio dell'esecuzione dell'algoritmo (configurazione specificata nella quarta colonna della precedente tabella) e nella parte b) della figura la configurazione che sarà presente alla fine dell'esecuzione dell'algoritmo.



a) all'inizio dell'esecuzione dell'algoritmo b) alla fine dell'esecuzione dell'algoritmo

Come sappiamo, l'algoritmo inizia a verificare i bit di uso associati alle varie frame di memoria iniziando dalla frame successiva a quella che per ultima è stata rimpiazzata. Poiché l'ultima frame che è stata rimpiazzata è quella di indice 1, come risulta dai dati presenti nella seconda colonna della tabella, l'algoritmo comincia a scandire le frame a partire da quella di indice 2 azzerando ogni bit di uso che trova al valore 1 e fermandosi alla prima frame che trova col bit di uso a zero. Quindi, come indicato nella parte b) della figura precedente, la vittima da rimpiazzare è la frame di indice 3.