**Sistemi di Calcolo (A.A. 2014-2015)**
Corso di Laurea in Ingegneria Informatica e Automatica
Sapienza Università di Roma

**Esercizi riepilogativi sulla seconda parte del Modulo I – Memoria virtuale**

---

**Domanda 1**

La memoria virtuale è una tecnica di gestione della memoria basata sulla separazione tra:

| | | | |
|---|---|---|---|
| **A** | lo spazio di indirizzi logico accessibile al sistema operativo e all'hardware e quello fisico gestito dai processi | **B** | lo spazio di indirizzi logico accessibile ai processi e quello fisico gestito dal sistema operativo e dall'hardware |

---

**Domanda 2**

In un sistema di memoria virtuale lo spazio logico dei processi non può essere più grande di quello fisico:

| | | | |
|---|---|---|---|
| **A** | Vero | **B** | Falso |

---

**Domanda 3**

In un sistema di memoria virtuale:

| | | | |
|---|---|---|---|
| **A** | gli indirizzi fisici vengono mappati su quelli logici | **B** | gli indirizzi logici vengono mappati su quelli fisici |

---

**Domanda 4**

In un sistema di memoria virtuale, la tabella delle pagine fornisce:

| | | | |
|---|---|---|---|
| **A** | un meccanismo per realizzare in modo efficiente il mapping tra indirizzi dello spazio virtuale e quelli dello spazio fisico | **B** | un meccanismo efficiente per ricercare dati in memoria |

---

**Domanda 5**

Una sola delle seguenti affermazioni sulla memoria virtuale è falsa. Quale?

| | | | |
|---|---|---|---|
| **A** | Tenendo distinti gli spazi virtuali (logici) di processi diversi, si impedisce che un processo possa interferire con le attività dell'altro in modo errato o malizioso, realizzando un meccanismo di protezione | **B** | Separando lo spazio virtuale da quello fisico, si può fare in modo che un processo usi più memoria di quella disponibile mappando pagine su disco invece che in RAM |
| **C** | Essendo basato su pagine tutte della stessa dimensione, consente di portare a zero il livello di frammentazione interna della memoria | **D** | Consentendo di mappare pagine di indirizzi virtuali di processi distinti sullo stesso frame fisico, la memoria virtuale consente la comunicazione e la cooperazione tra processi diversi |

**Domanda 6**

Qual è la dimensione tipica di una pagina in un sistema di memoria virtuale?

| | | | |
|---|---|---|---|
| **A** | 4 MB | **B** | 4 KB |
| **C** | 64 byte | **D** | 64 KB |

**Domanda 7**

Quanto grande è lo spazio di memoria virtuale che è possibile indirizzare usando un puntatore a 16 bit?

| | | | |
|---|---|---|---|
| **A** | 64 GB | **B** | 64 TB |
| **C** | 64 MB | **D** | 64 KB |

**Domanda 8**

Quanti bit deve avere un puntatore per indirizzare uno spazio di memoria virtuale di 1 GB?

| | | | |
|---|---|---|---|
| **A** | 32 | **B** | 64 |
| **C** | 24 | **D** | 30 |

**Domanda 9**

Quanto dovrebbe essere grande una tabella delle pagine per mappare uno spazio di memoria virtuale di 1 TB su uno spazio di memoria fisico di 4 GB con pagine di 1 KB?

| | | | |
|---|---|---|---|
| **A** | 128 MB | **B** | 1 GB |
| **C** | 4 GB | **D** | 512 MB |

**Domanda 10**

Una sola delle seguenti affermazioni è corretta:

| | | | |
|---|---|---|---|
| **A** | 1 KB = $2^{16}$ byte, 1 MB = $2^{24}$ byte, 1 GB = $2^{32}$ byte, | **B** | 1 KB = $2^{10}$ byte, 1 MB = $2^{20}$ byte, 1 GB = $2^{30}$ byte |
| **C** | 1 KB = $2^{12}$ byte, 1 MB = $2^{22}$ byte, 1 GB = $2^{32}$ byte | **D** | 1 KB = $2^{20}$ byte, 1 MB = $2^{30}$ byte, 1 GB = $2^{40}$ byte |

**Domanda 11**

Una tecnica implementativa comune nei sistemi di memoria virtuale consiste nel partizionare ogni indirizzo virtuale in una parte $p$ che specifica il numero di pagina in cui ricade l'indirizzo e una parte $d$ che specifica l'offset dell'indirizzo all'interno della pagina. Qual è la dimensione in bit di $p$ e $d$ per uno spazio virtuale di 1 GB suddiviso in pagine di 8 KB?

| | | | |
|---|---|---|---|
| **A** | $size(p) = 16\ bit, size(d) = 8\ bit$ | **B** | $size(p) = 20\ bit,\ size(d) = 10\ bit$ |
| **C** | $size(p) = 18\ bit, size(d) = 13\ bit$ | **D** | $size(p) = 17\ bit,\ size(d) = 13\ bit$ |

**Domanda 12**

Si consideri uno spazio di memoria virtuale di 4 GB suddiviso in pagine di 4 KB. Qual è il

numero di pagina p e l'offset d dell'indirizzo 0xABADBABE?

| | | | |
|---|---|---|---|
| **A** | p=0xABAD, d=0xBABE | **B** | p=0xABADBA, d=0xBE |
| **C** | p=0xABADB, d=0xABE | **D** | p=0xABA, d=0xABE |

---

**Domanda 13**

In un sistema operativo Linux a 32 bit ogni processo ha associato uno spazio virtuale di 4 GB che ospita i vari segmenti CODE, DATA, HEAP, STACK. Aprendo una shell e digitando il comando `ps aux` si noterà che vi sono centinaia di processi nel sistema anche ove la memoria fisica sia di pochi GB. Ne consegue che non tutta la memoria virtuale associata a ogni processo può essere effettivamente mappata su frame della memoria centrale. Come fa il sistema operativo a distinguere efficientemente quali pagine sono mappate su frame della memoria centrale e quali no?

| | | | |
|---|---|---|---|
| **A** | Il bit più significativo di un indirizzo vale 1 se la pagina in cui ricade e valida, e zero altrimenti | **B** | Il sistema operativo mantiene un array contenente i numeri delle pagine che sono mappate su frame fisici |
| **C** | Il sistema operativo mantiene una lista collegata che contiene i numeri delle pagine che sono mappate su frame fisici | **D** | A ogni entry della tabella delle pagine è associato un bit di validità della pagina che stabilisce se la pagina è in memoria centrale o meno |

---

**Domanda 14**

Il segmento HEAP di un processo mantiene:

| | | | |
|---|---|---|---|
| **A** | Il codice eseguibile su cui è basato il processo | **B** | Le stringhe contenute nel testo del programma |
| **C** | I blocchi allocati dinamicamente dal programma | **D** | Le variabili con durata statica (es. variabili globali) di un programma |

---

**Domanda 15**

Il segmento DATA di un processo mantiene:

| | | | |
|---|---|---|---|
| **A** | Il codice eseguibile su cui è basato il processo | **B** | I record di attivazione delle funzioni |
| **C** | I blocchi allocati dinamicamente dal programma | **D** | Le variabili con durata statica (es. variabili globali) di un programma |

---

**Domanda 16**

Il segmento TEXT di un processo mantiene:

| | | | |
|---|---|---|---|
| **A** | Il codice eseguibile su cui è basato il processo | **B** | Le stringhe contenute nel testo del programma |
| **C** | I blocchi allocati dinamicamente dal programma | **D** | Le variabili con durata statica (es. variabili globali) di un programma |

---

**Domanda 17**

Il segmento STACK di un processo mantiene:

| | | | |
|---|---|---|---|
| **A** | Il codice eseguibile su cui è basato il processo | **B** | I record di attivazione delle funzioni |
| **C** | I blocchi allocati dinamicamente dal programma | **D** | Le variabili con durata statica (es. variabili globali) di un programma |

---

**Domanda 18**

Le variabili locali alle funzioni sono memorizzate nel segmento:

| | | | |
|---|---|---|---|
| **A** | CODE | **B** | DATA |
| **C** | HEAP | **D** | STACK |

---

**Domanda 19**

Le variabili statiche sono memorizzate nel segmento:

| | | | |
|---|---|---|---|
| **A** | CODE | **B** | DATA |
| **C** | HEAP | **D** | STACK |

---

**Domanda 20**

I parametri delle funzioni sono memorizzati nel segmento:

| | | | |
|---|---|---|---|
| **A** | CODE | **B** | DATA |
| **C** | HEAP | **D** | STACK |

---

**Domanda 21**

Un blocco allocato dinamicamente con `malloc`/`calloc` è memorizzato nel segmento:

| | | | |
|---|---|---|---|
| **A** | CODE | **B** | DATA |
| **C** | HEAP | **D** | STACK |

---

**Domanda 22**

Parte del codice eseguibile delle funzioni è memorizzato nel segmento:

| | | | |
|---|---|---|---|
| **A** | CODE | **B** | DATA |
| **C** | HEAP | **D** | STACK |

---

**Domanda 23**

Le stringhe che appaiono in un programma C (es. `"hello world"`) sono memorizzate nel segmento:

| | | | |
|---|---|---|---|
| **A** | CODE | **B** | DATA |
| **C** | HEAP | **D** | STACK |