CHAPTER 09

Virtual-Memory Management

(61 Questions)

01: What does code need?

02: What rarely the entire program use?

03: What is ***Virtual Memory***?

04: What are the ***functions of Virtual Memory***?

05: How many ways for ***Virtual Memory Implementation***?

06: What is the difference between ***Demand Paging*** and ***Demand Segmentation***?

07: What are the ***benefits of Virtual Address Space***?

08: What is ***Demand Paging***?

09: What is ***Lazy Swapper***?

10: What is a ***Pager***?

11: What is ***Page Fault***? What is its Attributes?

12: What is ***Aspects of Demand Paging***?

13: Why ***Demand Paging*** needs Hardware support?

14: What is the ***performance of Demand Paging***?

15: What is ***Copy-on-Write*** (COW)?

16: What are the ***functions of Copy-on-Write*** (COW)?

17: What happens if there is ***no Free Frame***?

18: What is ***Page Replacement***? What is its performance?

19: Why we use ***Modify (dirty) Bit***?

20: What is the ***Basic Page Replacement***?

21: What would be happen to basic page replacement, ***if we found a Free Frame***? What if not?

22: What is the difference between ***Page*** and ***Frame Replacement Algorithm***?

23: How to ***Evaluate Algorithm***? What is ***String***?

24: What causes if we added more frames?

25: What does it mean "***3 frames***"?

26: How to track ages of pages?

27: What is ***Optimal Algorithm***?

28: What are ***functions of Least Recently Used (LRU)***?

29: What is ***Counter Implementation***?

30: What is ***Stack Implementation***?

31: What are ***LRU*** and ***OPT***?

32: Although ***LRU*** uses special hardware but still slow. Why?

33: What is ***Counting Algorithms***?

34: What are the ***schemes of Counting Algorithms***?

35: What is Page-Buffering Algorithms?

36: What is the possibility of ***Page-Buffering Algorithms***?

37: What is the relationship between ***Applications*** and ***Page Replacement***?

38: What are the ***major allocation schemes***? What is the difference between both?

39: What are the ***Fixed Allocation Schemes***?

40: What is ***Priority Allocation***?

41: How to classify the category of ***Page-Replacement Algorithms***?

42: What is the ***Non-Uniform Memory Access (NUMA)?***

43: What is ***Thrashing***?

44: What are ***the reasons that lead Thrashing***?

45: Why dose ***Demand Paging*** work?

46: Why does ***Thrashing*** occur?

47: What are the ***functions of Memory-Mapped Files***?

48: Can we use ***COW*** for read/write non-shared pages?

49: What are the attributes of ***Allocating Kernel Memory***?

50: What is ***Buddy System***?

51: Mention the ***Advantages and Disadvantages of Buddy System***?

52: What is a ***Slab*** and ***the relationship with Cache***?

53: Explain the ***Slab Allocator***?

54: What are the ***benefits of Slab Allocator***?

55: What are the ***functions of Prepaging***?

56: What are the ***considerations*** that must be taken for ***selection Page Size***?

57: What is ***TLB Reach***?

58: What will it happen if we ***increase*** or ***provide multiple Page Sizes***?

59: What is ***I/O interlock***?

60: Describe how ***Windows implements virtual memory***?

61: Describe how ***Solaris implement virtual memory***?

End of Questions (Chapter 9).

Chapter 09

Answer

1: **الكود** يحتاج بأن يكون في الذاكرة لتنفيذ البرامج.

2: البرنامج بالكامل نادرا ما يستخدم: (Error code, unusual routings, large data structures)

3: **الذاكرة الافتراضية (Virtual Memory):** هو فصل الذاكرة المنطقية (Logical Memory) للمستخدم من الذاكرة الفعلية (Physical Memory).

4: **خصائص الذاكرة الافتراضية:**

1) فقط جزء من البرنامج يحتاج لأن يكون في الذاكرة من أجل التنفيذ.
2) مساحة العنوان المنطقية يمكن ان يكون أكبر بكثير من مساحة العنوان الفعلية.
3) يسمح لمساحات العنوان بأن تكون مشتركة مع العديد من العمليات.
4) يسمح بإنشاء عملية أكثر كفاءة.
5) تشغيل العديد من البرامج في وقت واحد (concurrently).
6) محتاجة لمدخلات ومخرجات اقل من اجل تحميل أو مبادلة العمليات.

5: **الذاكرة الافتراضية بإمكانها أن تكون منفذة عبر طريقتين:**

a) (Demand Paging)
b) (Demand Segmentation)

6: **الفرق بين (Demand Paging) و (Demand Segmentation):**

	Segmentation	Paging
Unit	وحدة منطقية	وحدة مادية (Physical)
Visibility	يمكن لبرامج المستخدم رؤيتها	لا تستطيع برامج المستخدم رؤيتها
Length	متغير (Variable)	ثابت (Fixed)
Scheme	مخطط إدارة الذاكرة (Memory Management Scm.)	مخطط الذاكرة الافتراضية
Address Space	عنوان ذي بعدين (Two Dimensional Address)	عنوان ذو بعد واحد (One Dimensional Address)
Data	القاعدة، محدود	رقم الصفحة، رقم الإطار
Protection	جيد	صعبه (Not easy)

7: **فوائد مساحة العنوان الافتراضية:**

1. (System Libraries) يمكن أن تكون مشتركة بواسطة العديد من العمليات وذلك من خلال خرائط الأشياء المشتركة (shared object) في مساحة العنوان الافتراضية.
2. بالمثل، العمليات بإمكانها مشاركة الذاكرة.
3. الصفحات بإمكانها أن تكون مشتركة خلال عملية الانشاء مع نظام الاستدعاء (()fork).

8: **(Demand Paging):** هو أن العملية لا توضع بالكامل في الذاكرة ولكن جزء منها ليقوم بإحضار الصفحات التي نحتاجها إلى الذاكرة عند الحاجة إليها.

9: **(Lazy Swapper)** لا يقوم أبدا بتبديل (swaps) الصفحة داخل الذاكرة باستثناء (unless) الصفحة التي نحتاجها.

10: **(Pager)** هو المقايض (Swapper) الذي يتعامل (deals) مع الصفحات.

11: **(Page Fault):** لو كان هناك مرجع (reference) إلى صفحة، فإن أول مرجع (reference) إلى تلك الصفحة ستعيق (will trap) نظام التشغيل. وأبرز صفاته:

1) نظام التشغيل يبحث عن جدول اخر ليقرر: (المرجع الغير صالح) أو (ليست في الذاكرة).
2) يحصل على إطار فارغ.
3) يبادل الصفحة في داخل الإطار عبر عمليات القرص المجدولة.
4) إعادة تعيين الجداول ليظهر (to indicate) الصفحة الان في الذاكرة.
5) إعادة تشغيل التعليمات التي سببت بـ (Page Fault).

12: **(Aspects of Demand Paging):**

a) الحالة القصوى (Extreme Case) المعالجة تبدأ بدون صفحات في الذاكرة.
b) التعليمات المعطاه قد تصل إلى العديد من الصفحات > (multiple page faults)
c) دعم أجهزة (Hardware) المحتاجة للـ (demand paging).

13: **دعم أجهزة الـ (Hardware) وحاجتها لـ (demand paging):**

a) جدول الصفحة مع (valid / invalid bit)
b) الذاكرة الثانوية (تبادل الأجهزة مع تبادل المساحة)
c) إعادة تشغيل التعليمات

14: **أداء (Performance or Stages in Demand Paging):**

1. فخ (Trap) لنظام التشغيل.
2. يحفظ تسجيلات المستخدم وحالة العملية.
3. يحدد بأن الـ (interrupt) كان (Page Fault).
4. يشيك مرجع الصفحة بأنها كانت نظامية وتحديد موقع الصفحة على القرص.
5. يصدر قراءة من القرص إلى إطار حر:
   a) ينتظر في الصف لأجل هذا الجهاز إلى ان يكون طلب القراءة تحت الخدمة.
   b) انتظار جهاز (Seek and/or latency time)
   c) يبدأ بنقل الصفحة إلى إطار حر.
6. بينما هو في حالة انتظار، يقوم بتخصيص وحدة المعالجة المركزية إلى بعض المستخدمين الاخرين.
7. يستقبل (interrupt) من القرص (I/O subsystem) (I/O completed).
8. يحفظ التسجيلات وحالة العملية للمستخدم الاخر.
9. يحدد تلك الـ (interrupt) كانت من القرص.
10. تصحيح جدول الصفحة وبعض الجداول لعرض الصفحة الان في الذاكرة.
11. ينتظر وحدة المعالجة المركزية لتكون متخصصة لذلك المعالج مرة أخرى.
12. إعادة تخزين تسجيلات المستخدم، وحالة المعالجة، وجدول صفحة جديدة وبعد ذلك يلخص تعليمات المقاطعة.

15: **(Copy on Write (COW)):** يسمح لعمليات (Parent) و(Child) معا ليتقاسمو من البداية (initially) نفس الصفحات في الذاكرة.

16: **وظائف الـ (COW):**

1) يسمح بإنشاء عملية أكثر كفاءة، كما يتم نسخ الملفات المعدلة (modified) فقط.
2) عامة، الصفحات الحرة تكون مخصصة من صفحات (a pool of zero fill on demand).

17: **عند عدم وجود (Free Frame) سيحدث الآتي:**

1. يستخدم من قبل الصفحات العملية.
2. وأيضا في الطلب (demand) من النواة، و (I/O buffers) ... الخ

18: **(Page Replacement):** هو إيجاد بعض الصفحات في الذاكرة، ولكن ليس حقا في حالة استعمال.

- **(Its performance):**
  a) منع الافراط في تخصيص الذاكرة عن طريق تعديل خدمة (page fault) الروتينية لتشمل (page routing).
  b) استخدام (modify bit) لتقليل أعالي الصفحة (overhead of page) الذي ينقل فقط الصفحات المعدلة المكتوبة إلى القرص.
  c) يكمل الفصل بين الذاكرة المنطقية والذاكرة الفيزيائية – الذاكرة الافتراضية الكبيرة يمكن توفرها في ذاكرة فيزيائية أصغر.

19: **يستخدم (modify bit)** لتقليل أعالي الصفحة (overhead of page) الذي ينقل فقط الصفحات المعدلة المكتوبة إلى القرص.

20: **(Basic Page Replacement):**

1. إيجاد موقع الصفحة المرغوب فيها في القرص.
2. إيجاد الإطار الحر:
   a) لو كان هناك (Free Frame)، يستخدمها.
   b) إذا لم يكن هناك (Free Frame)، يستخدم خوارزمية (page replacement) لاختيار (Victim frame). يكتب (victim) إلى القرص إذا كان متسخ أو كثير الأخطاء.
3. إحضار الصفحة المرغوب فيها إلى داخل الإطار الحر؛ وتحديث الصفحة وجداول الإطار.
4. استمرارية المعالجة بواسطة إعادة تشغيل التعليمات التي تسببت بالـ (Trap).

21: **عند توفر الإطار الحر في (Basic Page Replacement)** فسوف يستخدمه وأما إذا لم تتوفر فسوف يستخدم خوارزمية (Page Replacement) لاختيار (Victim frame).

22: **الفرق بين خوارزمية تخصيص الإطار وخوارزمية تبديل الصفحة:**

- (Frame-allocation algorithm): يحدد
  a) كم عدد الإطارات ليعطي كل عملية.
  b) أي الإطارات التي تحتاج إلى تغيير.
- (Page-replacement algorithm):

يحتاج إلى أدنى معدل للـ (page fault) على (first access) و (re-access) معا.

23: يتم تقييم الخوارزمية بواسطة تشغيلها على (Sting) معين من مراجع الذاكرة (reference string) وحوسبة ارقام (page fault) على ذلك الـ (String).

- (String): هو عبارة عن ارقام صفحات، وليست معنونه بالكامل.
- مع العلم بأن تكرار (access) في نفس الصفحة لا يسبب بالـ (page fault).

24: إضافة عدد أكثر من الإطارات قد يسبب بـ (page faults) أكثر.

25: **(3 frames)** يعني ثلاث صفحات باستطاعتها ان تكون داخل الذاكرة في نفس الوقت ولكل عملية.

26: نستطيع تعقب أعمار الصفحات عن طريق استخدام مصفوفة (FIFO: First Come First Out).

27: **(Optimal Algorithm):** يقوم باستبدال الصفحة التي لن تستخدم لفترة زمنية أطول.

28: **وظائف الـ (LRU: Least Recently Used):**

a) استخدام المعرفة المسبقة بدلا من المستقبل.
b) تغيير مكان الصفحة التي لم تستخدم في أكبر قدر من الوقت.
c) تعريف وقت آخر استخدام مع كل صفحة.
d) تنفيذ الحسابات (Counter Implementation)
e) (Stack Implementation).

29: **يتم عملية (Counter Implementation):**

- كل صفحة مدخلة تحتوي على (counter)، وكل وقت صفحة تكون مرجعيته خلال ذلك المدخل، ويقوم بنسخ الساعة على الـ (counter).
- عندما تحتاج الصفحة بأن تكون مغيره، انظر إلى (counters) لإيجاد القيمة الأصغر. والبحث يكون من خلال الجداول المحتاجة.

30: **تتم عملية (Stack Implementation):**

- الاحتفاظ بأرقام صفحات (Stack) في شكل رابط مزدوج.
- مرجعية الصفحة: (تحريكها إلى الأعلى / يتطلب 6 مؤشرات لتتغير)
- كل عملية تحديث تكون تكلفتها عالية.
- لا يوجد بحث في (replacement).

31: **(LRU) و (OPT)** هما حالتان من حالات (Stack Algorithm) التي لا تملك (Belady's Anomaly).

32: **بالرغم من أن (LRU) يحتاج إلى أجهزة خاصة إلى انه مازال بطيء بسبب:**

1. **(Reference bit):**
   - كل صفحة معرفة بـ Bit، (initially = 0)
   - عندما تكون الصفحة مرجعيتها bit تعين إلى 1.
   - استبدال أي صفحة مع مرجعية bit = 0 (لو وجد واحد)
2. **(Second change algorithm):**
   - عادة (FIFO)، مزودة بجهاز مرجعيته bit فإن: الساعة تستبدل ولو كانت الصفحة التي ستستبدل مرجعيتها (bit = 0) فسوف يستبدلها وأما إذا كان (bit = 1) فإنه سوف يتم:
     - تعيين مرجع bit 0 ، يترك الصفحة في الذاكرة.
     - يبدل الصفحة التالية، ويبقي الموضوع بنفس القواعد.

33: **(Counting Algorithm)** يقوم بالحفاظ على ارقام (counter) للمراجع التي تم اجراءها في كل صفحة.

34: **(Schemes of Counting Algorithm):**

- (LFU Algorithm): يقوم باستبدال الصفحة مع أصغر (count).
- (MFU Algorithm): مبني على البرهان (argument) لتلك الصفحة صاحب أصغر (count) والتي كانت من المحتمل ان تكون فقط مقدمة فيها ولم تكن مستخدمه بعد.

35: **(Page-Buffering Algorithm): الاحتفاظ دائما بـ (a pool) للإطارات الحرة:**

- يبقي الإطار متاح عند الحاجة، وغير موجودة عند (fault time).
- يقرأ الصفحة داخل الإطار ويختار (victim) للاسترداد والاضافة إلى (free pool).
- عندما تكون ملائمة، تسترد (victim).

36: **احتماليات عمل (Page-Buffering Algorithm):**

- من المحتمل، الاحتفاظ بقائمة الصفحات المعدلة.
- من المحتمل، الاحتفاظ بمحتويات الإطار الحر سليمة (intact) ويلاحظ يكون في داخلهم:
  - لو أعيدت مرجعيته قبل استخدامه مرة أخرى، لا يحتاج لتحميل المحتويات مرة أخرى من القرص.
  - عادة تكون ذو فائدة لتقليل العقوبة لو كان هناك اختيار إطار (victim) خاطئ.

37: **العلاقة بين (Applications) و (Page Replacement):**

- جميع هذه الخوارزميات تمتلك (OS guessing) عن مستقبل وصول الصفحة.
- بعض ال (applications) تمتلك معرفة أفضل.
- الذاكرة المكثفة للـ (applications) يمكن ان تسبب (buffering) مزدوج:
- نظام التشغيل بإمكانه ان يعطي إذن مباشر إلى القرص، والخروج عن طريق الـ (applications).

38: **هناك مخططين تخصيص رئيسيتين (Major allocation schemes):**

1- تخصيص ثابت (fixed allocation)
2- تخصيص أولوية (priority allocation)

39: **التخصيص الثابت (Fixed allocation):**

- تخصيص متساوي (Equal allocation): مثال على ذلك، لو كان هناك 100 إطار (بعد تخصيص الإطارات لنظام التشغيل) و5 عمليات. سيعطي كل عملية 20 إطار.
- تخصيص نسبي (Proportional allocation): يقوم بالتخصيص وفقا لحجم العملية.

40: **تخصيص أولوية (Priority allocation):** يستخدم هنا الأولوية بدلا من الحجم.

41: **تصنف خوارزميات (Page replacement) إلى فنتين:**

1- عالمي (Global replacement):

العملية تختار إطار استبدال من مجموعة من كل الإطارات؛ كل عملية تستطيع ان تأخذ إطار من الآخر:

- لكن بعد تنفيذ العملية بإمكان الوقت ان يختلف بشكل كبير.
- ولكنه أكبر إنتاجية لذا هو أكبر شيوعا.

2- محلي (Local replacement):

كل عملية تختار فقط من مجموعته الخاصة من الإطارات المتخصصة:

- أكثر اتساقا في أداء كل عملية.
- ولكن ربما تكون الذاكرة غير مستغلة (underutilized memory).

42: **(NUMA: Non-Uniform Memory Access):**

- حتى الآن كل ذاكرة الوصول على قدم المساواة.
- أنظمة كثيرة تعتبر (NUMA) – سرعة الوصول إلى الذاكرة تختلف (Varies)
- الأداء الأمثل (Optimal performance) يأتي من تخصيص الذاكرة.
- حلها عن طريق (Solaris) وذلك بإنشاء (igroup):
  - هيكلة لتتبع CPU / انخفاض ذاكرة (latency groups)
  - استخدام (my schedule and pager)
  - عند إمكانية جدولة كل تشعبات العملية وتخصيص كل الذاكرة لتلك العملية ضمن Igroup

43: **(Thrashing):** هي عملية تكون مشغولة في دخول وخروج (swapping pages).

44: **الأسباب التي تقود إلى (Thrashing):**

1) استخدام وحدة المعالجة المركزية منخفض ().
2) نظام التشغيل يفكر في أنه يحتاج إلى زيادة درجة تعدد البرمجة.
3) عملية أخرى مضافة إلى النظام.

45: **يعمل الـ (demand paging) لأنه نموذج محلي (Locality model):**

- العملية تنزح أو تهاجر من (locality) إلى آخر
- (localities) قد تتداخل

46: **تجري عملية (Thrashing) لأن حجم الـ () هو مجموع حجم الذاكرة:**

- المحدودية تتأثر باستخدام محلية وأولية (page replacement).

47: **وظائف (Memory-Mapped Files):**

1- (Memory-mapped file I/O) يسمح لملف الادخال والإخراج بأن يعامل على انها ذاكرة وصول روتيني وذلك بواسطة (mapped a disk block) إلى صفحة في الذاكرة.
2- الملف في البداية يقرأ باستخدام الـ (demand page)
   - قراءة جزء بحجم الصفحة من الملف من نظام الملفات إلى صفحة فيزيائية.
   - يقرأ ويكتب لاحقا إلى ومن الملف المعالج كذاكرة وصول عادي.
3- يبسط ويسرع الوصول إلى الملفات بواسطة قيادة ملفات I/O من خلال الذاكرة بدلا من استدعاءات النظام (read () and write()).
4- يسمح أيضا بعدة عمليات لتعيين نفس الملف والسماح للصفحات في الذاكرة بأن تكون مشتركة.
5- بعض عمليات التشغيل تستخدم (memory mapped files) لأجل (standard I/O)
6- COW يمكن استخدامه لقراءة وكتابة الصفحات الغير مشتركة.
7- (Memory mapped files) يمكن استخدامه للذاكرة المشتركة.

48: يمكن استخدام (COW) لقراءة وكتابة الصفحات الغير المشتركة.

49: **من خصائص تخصيص Kernel Memory:**

- تتعالج بشكل مختلف من ذاكرة المستخدم
- غالبا يتم تخصيصه من (free user memory).

50: **(Buddy System)** يقوم بتخصيص الذاكرة من جزء الحجم الثابت المتألفة من (physically-contiguous pages).

51: **أبرز مميزات (Buddy System):** تتجمع الأجزاء الغير مستخدمة من القطعة الصغيرة بسرعة إلى قطعة أكبر.

وأبرز عيوب (Buddy System): التجزئة أو (fragmentation)

52: **(Slap)** هي صفحة أو أكثر من صفحة متجاورة فيزيائيا (physically contiguous pages).

(Cache) تتألف من slap أو أكثر من slaps.

53: **من سمات (Slab Allocator):**

- استراتيجية بديلة
- كل (cache) مملوء بـ (objects) وهي عبارة عن تعليمات لهيكلة البيانات.
- عندما يتم انشاء (cache) تعبأ بـ (objects) وتوضع كـ (free).
- عندما الهيكلة يتم تخزينه (objects) توضع كـ (used).
- لو كان slap ممتلئ بـ (objects) فإن الـ (object) التالي يتم تخصيصه في ال slap الفارغ وعند عدم توفر slap فارغ فإنه يتم تخصيص slap جديد.
- الفوائد تشمل عدم التجزئة والذاكرة السريعة تتطلب الاكتفاء.

54: **فوائد (slap allocator)** تشمل عدم التجزئة والذاكرة السريعة تطلب الاكتفاء.

55: **خصائص الـ (Prepaging):**

1- لتقليل الاعداد الكبيرة من (page faults) التي تحدث عن عملية البدء.
2- Prepage كل أو بعض الصفحات العملية المطلوبة قبل أن يتم عملية الرجوع إليه.
3- ولكن لو كانت صفحات الـ (prepaged) غير مستخدمه؛ I/O والذاكرة تكون مهملة.

56: **هناك اعتبارات (considerations) يجب اتخاذها لاختيار حجم الصفحة:**

1) التجزئة (Fragmentation)
2) حجم جدول الصفحة.
3) الدقة
4) I/O overhead
5) عدد (page faults)
6) Locality
7) حجم TLB وكفاءته

57: **(TLB Reach)** هو مقدار الذاكرة التي يمكن الوصول إليها من (TLB).

58: **زيادة حجم الصفحة:**

- بإمكانه بأن يقود إلى الزيادة في التجزئة كما أنها ليست كل التطبيقات تتطلب حجم صفحة كبير.

توفير احجام صفحات متعددة:

- يسمح للتطبيقات التي تتطلب احجام صفحات أكير الفرصة لاستخدامهم من غير زيادة في التجزئة.

59: **(I/O Interlock):** هي صفحات يجب ان تكون مقفلة في بعض المرات في داخل الذاكرة.

60: **كيفية تنفيذ الويندوز للذاكرة الافتراضية:**

- يستخدم demand page مع المجموعات (clustering). Clustering يحضر في الصفحات المحاطة بـ (faulting page)
- العمليات يتم تعيينها على (working set minimum) و (working set maximum).
- Working set minimum هو العدد الأدنى من الصفحات للعملية التي تكون مضمونه لتكون في الذاكرة.
- العملية يمكن ان يتم تعيينها في العديد من الصفحات إلى (working set maximum).
- عندما ينخفض مقدار الذاكرة الحرة في النظام تحت (threshold)، (**automatic working set trimming**) ويتم تنفيذه لاستعادة كمية الذاكرة الحرة.
- (Working set trimming) يقوم بحذف الصفحات من العمليات التي تملك صفحات في تجاوز (their working set minimum).

61: **كيفية تنفيذ Solaris للذاكرة الافتراضية:**

- يقوم بصيانة قائمة من الصفحات الحرة لتعيين عمليات (faulting).
- *Lotsfree* – threshold parameter (amount of free memory) to begin paging
- *Desfree* – threshold parameter to increasing paging
- *Minfree* – threshold parameter to being swapping
- Paging is performed by *pageout* process
- **Pageout** تقوم بتفحص الصفحات باستخدام تعديل خوارزمية الساعة.
- ***Scanrate*** هو معدل الصفحات التي فحصها. *This ranges from slowscan to fastscan*
- **Pageout** يستدعى في كثير من الأحيان لاعتماده على مقدار الذاكرة الحرة المتاحة.
- **Priority paging** يعطي الأولوية لمعالجة كود الصفحات.

END of chapter 09