

بِسْمِ اللَّهِ الرَّحْمَنِ الرَّحِيمِ

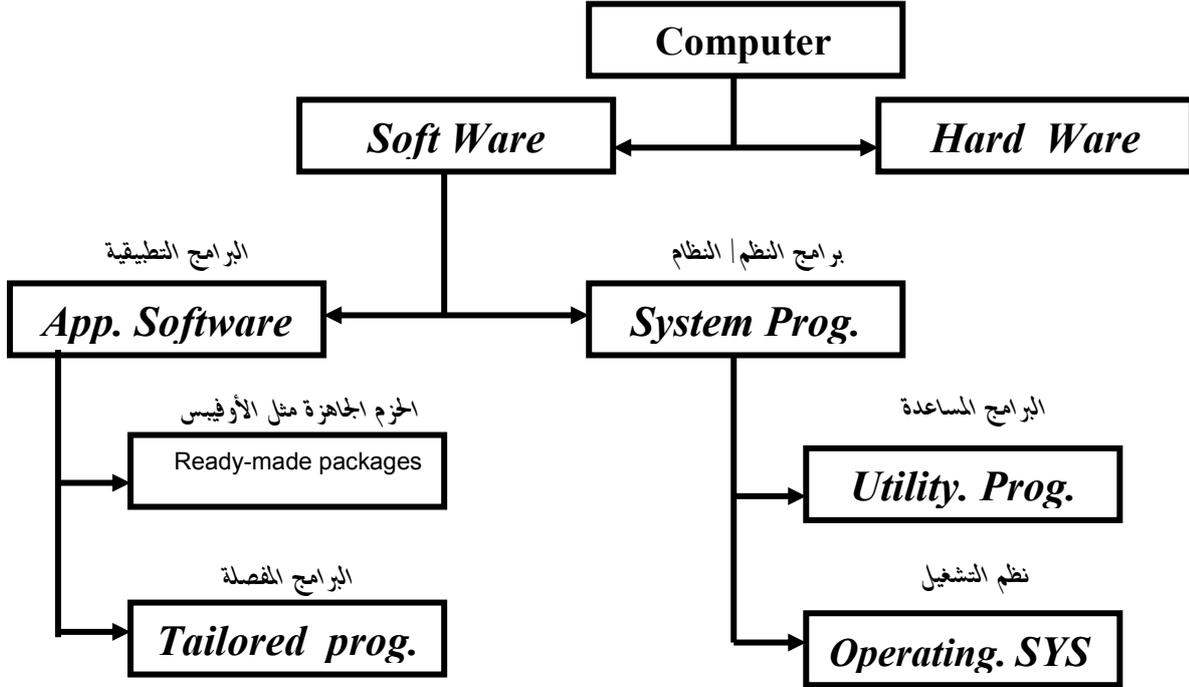
**جامعة القصارف**  
**كلية علوم الحاسوب ونقانة المعلومات**  
**الفصل الدراسي الخامس [علوم + نقانة]**

**محاضرات**

**فلسفة**

**نظم التشغيل**

- نظام التشغيل هو برنامج أو مجموعة برامج تقوم بإدارة معدات الحاسوب المختلفة .  
 أو : هو وسيط بين المستخدم وجهاز الحاسوب يمكن المستخدم من تنفيذ البرامج بكل سهولة وكفاءة .  
 تم تصميم نظم التشغيل مع الجيل الثالث للحواسيب عام ١٩٦٤م وذلك للإستفادة من الوقت الضائع من وحدة المعالجة المركزية عند إنتظار المشغل لتشغيل العملية التالية .  
 س | ما الهدف الأساسي من نظام التشغيل ؟ زيادة الإنتاجية أو رفع كفاءة تشغيل الحاسوب .



البرامج المكونة لنظام التشغيل :

- (١) البرنامج المشرف (Supervisor) ويهتم بالإشراف على برامج نظام التشغيل .
- (٢) برامج التحكم (Controller) ويهتم بالتحكم في سير العمليات أوهى التى يتعامل معها المشغل فى تنظيم وتشغيل العمليات وتنفيذ البرامج.
- (٣) برامج ترجمة اللغات (Compiler) ويهتم بعملية الترجمة إلى لغة الآلة (Machine Language) .
- (٤) برامج الصيانة (maintenance) ويهتم بصيانة نظم التشغيل .

وظائف أو أنشطة نظم التشغيل :

- (١) إدارة معدات الحاسوب المختلفة .
- (٢) حماية البرامج والبيانات من الأخطاء المقصودة وغير المقصودة من قبل المستخدمين .
- (٣) إختيار البرنامج المناسب فى كل عملية جديدة كمثال مكتبة البرنامج المقيم (Resident program Library)  
 توضيح : فلكل عملية برنامج مناسب للتشغيل وهذه البرامج موجودة بهذه المكتبة كمثال جمع عددين .
- (٤) الإنتقال من عملية تم تنفيذها إلى أخرى فى إنتظار التنفيذ . وهو الهدف من نظام التشغيل (زيادة الإنتاجية) .
- (٥) التحكم فى إدارة وتوزيع الذاكرة . (Memory Management)
- (٦) التفاعل بين المستخدم والحاسوب (تنظيم أسلوب المواجهة) كمثال الشاشة التالية :



### ويمكن تلخيص هذه الوظائف كالآتي :

(١) آلة ظاهرية (تخيلية | افتراضية) Virtual Machine كمثل قراءة أو كتابة فلايد من وجود عدد من المعاملات (البارامترات) لتحديد الأشياء. توضيح : من هذه المعاملات : أ- عنوان كتلة القرص المراد قراءتها . ب- عدد القطاعات في المسار ..... الخ هذه التعقيدات لا يحتاج إليها المبرمج ويحتوي القرص على مجموعة من الملفات فيحتاج المبرمج إلى طريقة يفتح بها الملف لإمكانية القراءة منه أو الكتابة عليه وإعادته بعد الإنتهاء من عمليتي القراءة أو الكتابة للقرص الصلب فلا يراها المستخدم إذ يمكن القول أن البرامج التي تخفي حقيقة المكونات المادية عن المبرمج أو المستخدم وتقدم له شكلاً بسيطاً وجميلاً في التعامل مع الملفات وإمكانية الكتابة أو القراءة منها هي نظام التشغيل وبالتالي يعتبر آلة ظاهرية ونظام التشغيل يقدم خدمات تستطيع البرامج الحصول عليها باستخدام تعليمات خاصة تسمى إستدعاءات أو نداءات النظام (System Calls) .

(٢) إدارة المصادر أو الموارد Resource Manager . مثل الطابعة وتنقسم إلى :  
 - برامج (Software) : مثل الـ (process) أو (files) أو (Records) .  
 - أجهزة (Hardware) : مثل الـ (Memory) أو (Printers)  
أمثلة لنظم تشغيل : ويندوز | يونكس | دوس | ماكنتوش وغيرها من النظم

### طبقات نظام الحاسوب (Structures of Computers) موقع نظام التشغيل وسط الحاسوب لإزالة التعقيد

App. Prog تسمى	Games (الألعاب)	Airline Reservation (مكاتب الحجز)	Banking System (نظام مصرفي)
SYS. Prog تسمى	Compiler (المترجم)	Editors (المحرر)	Interpreter (المفسر)
	Operating System (نظام التشغيل)		
	Machine Language (لغة الآلة)		
Primitive S.W تسمى	Micro Programming (البرامج   التعليمات) مثل Add		
	Physical Devices (الأجهزة الفيزيائية الدقيقة) مثل Chips و مزود الطاقة		

ورقة بحثية (١):

- س | كيف كان يتم التشغيل قبل تصميم نظم التشغيل ؟  
 س | ناقش : لا وجود للحاسوب دون وجود نظام تشغيل ؟  
 س | تحدث عن تطور نظم التشغيل ؟ نظام التشغيل مر بتاريخ تطوير طويل يلخص في الآتي :
- ١ | محتويات [أخف] Simple Batched Systems ٢ | أنظمة البرمجة المشتركة الخفيفة (Multiprogramming Batched Systems)
  - ٣ | أنظمة المشاركة الزمنية (Time-Sharing Systems) ٤ | أنظمة الحواسيب الشخصية (Personal Computer Systems)
  - ٥ | الأنظمة المتوازية (Parallel Systems)
  - ٦ | الأنظمة الموزعة (Distributed Systems)
  - ٧ | أنظمة الزمن الحقيقي (Real Time Systems)
- س | ماذا تعدد أنظمة التشغيل ؟ لوجود أنواع متعددة من المعالجات.

## أنواع نظم التشغيل :

تقسم تنظم التشغيل إلى أنواع حسب المهام (Tasks) والمستخدمين (Users) .

### حسب المهام :

- (١) مفرد المهام (Single Tasking) : وهو الذى يتعامل مع مهمة واحدة فى لوقت .  
(٢) متعدد المهام (Multi Tasking) : وهو الذى يتعامل مع أكثر من مهمة فى نفس الوقت .

### حسب المستخدمين :

- (١) مفرد المستخدم (Single Users) : وهو الذى يتعامل مع مستخدم واحد فى الوقت . (دوس)  
(٢) متعدد المستخدم (Multi Users) : وهو الذى يتعامل مع أكثر من مستخدم فى نفس الوقت . (ويندوز)  
وبالتالى يمكن أن يكون لدينا أربعة أنواع من الأنظمة كالتالى :

- (أ) نظام : مستخدم واحد ومتعدد المهام  
(ب) نظام : مستخدم واحد ومفرد المهام  
(ج) نظام : متعدد المستخدمين ومفرد المهام  
(د) نظام : متعدد المستخدمين ومتعدد المهام

### نشغيل نظام التشغيل :

بما انه برنامج أو مجموعة برامج فإنه يخزن فى ملف ويتم نقله إلى الذاكرة ويبقى فيها ليشراف على بقية البرامج الأخرى وعلى وحدات التخزين ويتم ذلك عند التشغيل فبمجرد الضغط على زر تشغيل الحاسوب يعمل برنامج صغير يوجد فى ذاكرة القراءة فقط (Rom) يسمى IPL حيث يقوم بفحص معدات الحاسوب ويتأكد من سلامتها ومن ثم يقوم بتحميل نظام التشغيل من القرص وبعدها يقوم نظام التشغيل بإستلام أوامر المستخدم وتنفيذها .

### برنامج (IPL) : initial program load

- هو برنامج يقوم بتحميل نظام التشغيل من الـ HD إلى الذاكرة حيث يقوم بالوظائف التالية :  
(١) يقوم بتنفيذ المهام المحددة .  
(٢) البحث عن معدات الحاسوب والتأكد من سلامتها .



### محاضرة رقم (٢)

### مفاهيم خاصة بنظم التشغيل : O. S . Concepts

- يعتبر مصمى نظام تشغيل يونكس أن نظام التشغيل يتكون من ثلاثة مفاهيم هى :  
(١) النواة Kernel (٢) القشرية Shell (٣) نظام الملفات Files System  
- بينما يعتبر مصمى آخرون غير مجموعة اليونكس المفاهيم التالية إضافة للمفاهيم السابقة :  
\* العملية أو المهمة job / process  
\* نداء النظام System Call  
\* المقاطعة Interrupt

### النواة Kernel :

هى عبارة عن مجموعة الوظائف ذات المستوى الأدنى لنظام التشغيل والتي تحمل الى الذاكرة كلما قمت بتشغيل الجهاز وذلك مباشرة بعد أن تعمل بعض الوظائف الموجودة فى الـ (Bios) وهو نوع من أنواع برامج النظام .

User program
Sys call
Kernel
Operating system
hardware

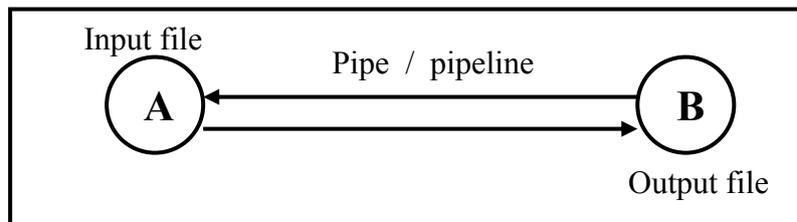
## القشرية shell

هي الواجهة المرئية ما بين المستخدم ونظام التشغيل وهي عبارة عن برنامج يعمل في الطبقة العليا منه ويسمح للمستخدمين بإصدار الأوامر إليه .  
تعريف آخر : لنظام دوس القشرية هي البرنامج الموجود في ملف Command . com وبالتالي هي :  
عبارة عن مفسر أوامر . shell is a command interpreter .

- ❖ هنالك نظم تشغيل بها عدد من القشريات مثلاً نظام يونكس كما أنه يقوم بإظهار الـ Shell . وكذلك الكثير من أنظمة التشغيل .
- ❖ هنالك أشخاص يعتبرون إنها scripting language لغة برمجة غير متاحة في مستوى اللغات العليا (HLL) ومفرداتها لا توجد في هذه اللغات عدا بعض المفردات في لغة السي مثل END والبرامج التي تكتب بلغة الـ Shell تسمى الـ Shell scripts . وهي التي تمكن المبرمج من كتابة System Programming .
- ❖ عندما يحتاج المبرمج تصميم System Programming يستخدم أوامر الـ Shell .
- ❖ في نظام دوس هنالك أوامر تكتب في الـ Shell .
- ❖ عندما نكتب الأمر توجد process في نظام التشغيل تسمى أيضاً Shell تأخذ الأمر وتقوم بتنفيذه . وهنالك أوامر تكتب في الـ Shell يقوم بتحميلها إلى الـ O.S وهنالك بعض الأوامر لا يستطيع الـ Shell تنفيذها وتنفذ بواسطة برامج نظام O.S الأخرى .
- ❖ اعتبرت الـ Shell جزء من الـ Kernel في نظم التشغيل القديمة أما النظم الحديثة فصلتها عن بعضهما تماماً .

## المهمة / العملية Process

- ❖ هي برنامج تحت التنفيذ & Program in execution
- ❖ البرنامج من الـ H D إلى الـ Memory لا يصبح Program ولكن Process .
- ❖ في كل عملية عندما ينفذ البرنامج يكون معه ما يسمى بالـ process address يوجد به عداد البرنامج (prog . counter) والمكدس (stack) ومؤشر المكدس (stack pointer) ..... إلخ كل هذا يسمى Process .
- ❖ يمكن أن يكون للعملية الواحدة مجموعة عمليات فرعية وبالتالي يكون لدينا شجرة عمليات (tree) وتسمى Process Hierarchical ويمكن أن تتصل مع بعضها وتسمى Inter process communication ويتم ذلك بوجود قناة اتصال تسمى الـ pipe أو الـ Pipe line وهو عبارة عن virtual file .
- مثال : إذا كان لدينا عملية (A) وعملية (B) وعند طلب معلومة معينة مثلاً طلبت (B) من (A) معلومة عبر قناة الإتصال فإن (A) تعتبر بمثابة output file و (B) تعتبر input file وهكذا وتتم عملية الإتصال هذه عبر الـ virtual file ويمكن توضيح ذلك بالرسم التالي :



توضيح بصورة موسعة :

يصبح الـ pipeline إفتراضى virtual ( pipe is a virtual file ) بمعنى وقتى يكون عند الحاجة ثم يحدث له ( terminate ) فتكون هنالك حالتان كما موضح أعلاه وهما :

- (1) process B wants info from process A
- (2) process A wants info from process B

ففي الحالة الأولى فإن (A) تنظر للـ pipe على أنه output file بالنسبة لها وتطرح المعلومة التي تريدها (B) في الـ pipe و (B) تنظر للـ pipe على أنه input file وبالتالي ستقرأ هذا الملف وستجد المعلومة التي طلبتها من (B) . والحالة الثانية العكس .

## بناء النظام : system call

- هي مجموعة من الأوامر (واجهة ما بين التطبيقات والنظام) يتم فهمها في نظام التشغيل .
- هي أمر واحد وهو غير متاح على مستوى اللغات العليا إلا لغة السي حيث تسمح باستخدام بعض مفردات الـ sys call .
- نداءات النظام مصممة خصيصاً للمبرمجين فالمستخدمين لا علاقة لهم بها .

## أنواع نداءات النظام: تضم خمسة مجموعات من الأوامر هي :

- (١) أوامر العمليات (control) process and job sys call كمثل end & abort & load & execute & create & delete
  - (٢) أوامر المعاملات | التعديل للملف file manipulation كمثل close & create & delete & open
  - (٣) أوامر الأجهزة devices manipulation كمثل إضافة جهاز جديد أو هي نفس الـ file manipulation .
  - (٤) أوامر الإتصال communication كمثل أوامر الضغط على أزرار لوحة المفاتيح . أو send & receive & delete links
  - (٥) أوامر معلومات الصيانة information maintenance وهي خاصة بالنظام مثل time & cpu usage & memory
- \* ملحوظة : تحدث المقاطعة عند تنفيذ أي عملية أو تنفيذ أى من هذه الأوامر .

المقاطعة interrupt عبارة عن حدث يغير من تسلسل تنفيذ الأوامر في المعالج كمثل إدخال الفلاش فجأة في الحاسوب .  
للمقاطعة عدة أسباب منها : إنتهاء الزمن المخصص للعملية .

## أنواع المقاطعات:

- (١) المقاطعة نتيجة إستدعاء المشرف: هي تقنية تساعد في الحفاظ على أمن نظام التشغيل من المستخدمين حيث أن المستخدم يجب أن يطلب الخدمة من خلال إستدعاء المشرف وبالتالي فإن نظام التشغيل يرفض طلب المستخدم إذا كان المستخدم لا يملك الصلاحية المناسبة كمثل حذف | تغيير كلمة السر .
- (٢) مقاطعات الإدخال والإخراج I/O interrupts : تحدث عندما تكتمل عملية إدخال أو إخراج بيانات من وحدات الإدخال والإخراج (الأجهزة) أو حدث خطأ في الإدخال والإخراج أو عند تجهيز جهاز معين .
- (٣) المقاطعات الخارجية External interrupts أو software interrupt وهي زمن معين لتنفيذ برنامج معين إذا لم ينتهي التنفيذ يتم إيقافه كمثل مقاطعة الطباعة . توضيح عندما تتم طباعة ملف معين وفجأة ينتهي الورق مثلاً . أو مثل إنتهاء الـ Quantum على الساعة وتسبب مقاطعة .  
\* ملحوظة : من الأسباب التي تجعل البرامج لا تنتهي حسب الزمن المعين دخولها في دوام لا نهائي وهو حدوث خطأ في برنامج معين .
- (٤) مقاطعة الإستئناف resume interrupts تقوم المقاطعة بإستئناف العمل عندما يصبح في الحالة ready . وهي تحدث عندما يضغط المشغل على زر الإستئناف للوحة التحكم أو عند وصول أمر إستئناف معالج بالإشارة إليه من معالج آخر في نظام المعالجات المتعددة .
- (٥) مقاطعات تدقيق البرامج : تحدث لأحد الأسباب الآتية :  
أ- مشاكل القسمة على الصفر .  
ب- محاولة تنفيذ شفرة لعملية غير صحيحة  
ج- محاولة الرجوع لموقع في ذاكرة غير موجودة
- (٦) مقاطعات حدوث الخلل (تدقيق الآلة) machine accuracy interrupts : تحدث نتيجة لحدوث خلل في جهاز الحاسوب .  
\* ملحوظة : توجد مع كل مقاطعة ما يسمى (ISR) .

## ال Interrupt Service Routine (ISR) : هي شفرة تحدد الحدث المطلوب مع كل نوع مقاطعة .

- عند وصول المقاطعة لوحدة المعالجة المركزية تقوم المقاطعة بتحويل التحكم من الـ cpu إلى الـ ISR .
  - أثناء تنفيذ الـ cpu للمقاطعة إذا حدثت مقاطعة أخرى تتوقف وتتعامل مع الأمر بطريقتين هما :  
الأولى priority scheme الأولوية للأهم والثانية Enable / disable السماح وعدم السماح .
  - تتمكن وحدة المعالجة المركزية من معرفة إنتهاء الـ ISR بإستعمال حالة الميرق Status flag
- ورقة بحثية (٢) أذكر خمسة من مقاطعات الأجهزة وخمسة مقاطعات للبرامج ؟  
س | هل يمكن إعادة المقاطعة (حدوث أكثر من مقاطعة في نفس الوقت) ؟  
يتم ذلك ب طريقتين هما : الأولى priority scheme الأولوية للأهم والثانية Enable / disable السماح وعدم السماح .

## مقاييس الحماية protection measures

س | كيفية حماية برامج نظم التشغيل من المستخدم أو نظام التشغيل نفسه ؟  
يتم ذلك بواسطة مقاييس الحماية وذلك لمنع أي تداخل بين برامج النظام والمستخدم . ويتم ذلك بأربعة طرق هي :

## Dual mode Operation (١)

يجب أن يكون لكل CPU طريقة لمعرفة نوع العملية هل هي (O.S أم User) ويتم ذلك بواسطة الـ flag وهو عبارة عن bit إضافية للتمييز وتسمى الـ mode bit وهي خانة يتم إضافتها للبرنامج لمعرفة هل هو O.S أم User وإذا كانت الخانة واحد هو User وإذا كانت صفر هو O.S فمثلاً برامج نظام التشغيل تسمى الـ Monitor mode والمستخدم تسمى الـ User mode والبرامج والتعليمات الخاصة بالنظام تسمى الـ Privilege instruction كمثل الـ timer .

## (٢) I/O protection

عند كتابة برنامج وهو يحتوي على I/O دائماً يوجد في الـ Monitor mode لذلك فإن كل أوامر الـ I/O موجودة فيها وذلك لأن المستخدم لا يستطيع أن يحدد مكان حفظ البرنامج أو الشفرة تحديداً في الذاكرة .

## (٣) Memory protection

هي الجهة المسئولة عن تحديد المواقع في الذاكرة والمسموح بتنفيذها فقط عن طريق نظام التشغيل مثلاً table of ISR في الذاكرة لا بد أن يكون نظام التشغيل على معرفة تامة به لمنع برامج المستخدمين من الدخول له في الذاكرة .

– تحدد مساحة الذاكرة من خلال الـ Register من خلال الـ Base & limit مساحة البرنامج داخل الذاكرة تساوى حاصل طرحهما كمثل : Base = 10 & limit = 20 إذا مساحة البرنامج تساوى 10 .

## (٤) CPU protection

يتم ذلك بجعل الـ cpu تقوم بتنفيذ العمليات الطبيعية فقط (العمل المخصص لها) وليس الغير مرغوب فيها كاللدار اللانهائي (infinite loop) .  
ولحل هذه المشكلة ظهر مفهوم الـ Timer . (privilege instruction)



## محاضرة رقم (٣)

### هيكلية نظم التشغيل : O . S Structure

هنالك عدة جوانب يجب على المصمم مراعاتها قبل تصميم النظام تتمثل في :

- (١) طبيعة نظام التشغيل : حسب نوعه multi أم Single أم غيره
- (٢) الهدف من النظام : هل يهدف للمستخدم أم للنظام .
- (٣) تصميم النظام: كمثل حسب طبيعة البيانات وحجمها وهذه الناحية مرتبطة بالـ Modules الموجودة و كل العمليات المتعلقة بها من حماية وغيره.

### أنواع بنائيات تصميم النظام :

(ب) Client - server system

(أ) Virtual machine

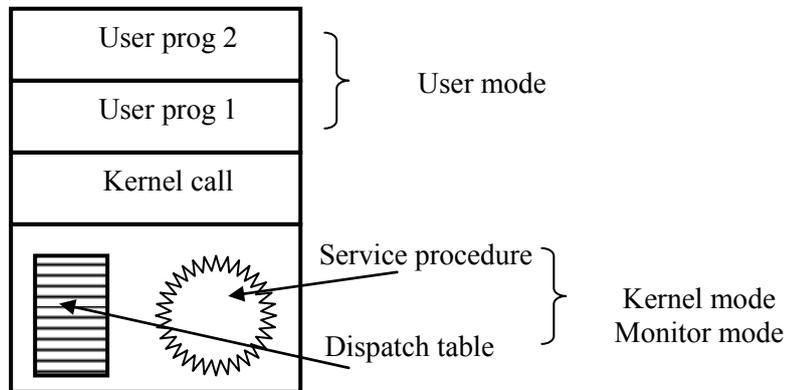
(د) Layered system

(ج) Monolithic system

وما يهمنا الآن هو النوع ( ج ، د ) فقط .

### الـ Monolithic system :

تعريف : هو عبارة عن مجموعة إجراءات مرتبطة ببعضها لتنفيذ العمل لوجود بنية منظمة لهذا النظام . وفي هذا النوع من نظم التشغيل يكتب نظام التشغيل كمجموعة من الإجراءات (Procedures) . (sub system) .  
بمعنى أنه ليس لديه هيكل وتتم كتابة الـ O . S في شكل عدد من الإجراءات كل إجراء يتصل بإجراءات أخرى ويتم تجميع كل الإجراءات في ملف واحد لتكون برنامج software يعرف بنظام التشغيل .



لكل نداء نظام وسائط وكل وسائط النظام تخزين في الـ Dispatch table كمثل ( ) delete .

\* كل الـ System call يشير لها الـ Programm بواسطة Parameter معين يتم تخزينها في وسط معين يعرف بالـ Dispatch table وبالتالي الـ O/S يقرأ الـ System call من خلال الـ Parameter . الموجودة في الـ Dispatch table .

User mode- CPU تنفيذ الـ Instruction التي تخص الـ User .

Kernel mode الـ CPU تنفيذ الـ Service المقدمة من الـ S.O .

- في الـ User mode تكون السيطرة للـ Program في التمييز وبما أن الـ User program يطلب خدمة إذا الكنتزول يتحول من الـ Program إلى الـ O. S وهذا يتم عبر الـ Kernel call أو Supervisor call .

\* الـ Dispatch table : هي منطقة تخزن فيها جميع وسائط النظام للنداء .

\* الـ Kernel call : هي منطقة تقوم بنقل التحكم من منطقة الـ User Mode إلى الـ Kernel call وتسمى أيضاً supervisor call

طريقة العمل : يقوم التحكم في البدء في التواجد في الـ User Mode وعند كتابة نداء مثلاً الأمر ( delete )

تكتب معه الوسائط وبعدها ينتقل التحكم إلى الـ Dispatch table لمعرفة الوسيط المطلوب ومنه ينتقل البرنامج من الـ User Mode إلى الـ

Monitor ويتم ذلك بواسطة الـ Kernel call وبعدها ينتقل إلى الـ Service وتبدأ الدورة من جديد هكذا .

ملحوظة : في بعض الأحيان لا تنفذ كل الخدمات عبر الـ Service ماذا يفعل النظام عندها ؟

يقوم بإستدعاء البرامج المساعدة (Utility Program)

معلومات أكثر يرجى مراجعة المرجع المكتبة بالكتابة !!!!!!!!

الـ Layered system :

هو نظام في شكل طبقات أو مستويات

Level 5	Operators
Level 4	User program
Level 3	I/O management
Level 2	Operators process communication
Level 1	Memory management
Level 0	Process allocation & multi programming

level 0 : يتعامل مع الـ Process allocation & multi programming الموجودة في الذاكرة ومع الـ Cup في التحويل بين

العمليات عند حدوث المقاطعات أو عند إنتهاء الـ timer .

level 1 : مسئول من عملية تحويل البرامج من القرص الصلب إلى الذاكرة الرئيسية أو العكس .

level 2 : مسئول من عملية الإتصال بين الـ process و الـ operator console .

level 3 : مسئول عن إدارة أدوات الإدخال والإخراج .

level 4 : مسئول عن التحكم في برامج المستخدمين .

level 5 : مرتبط مع الـ System Operator حيث أن كل عمليات الـ System Operator موجودة في هذا المستوى .

\* ملحوظة : الفرق بين الـ Monolithic & Layered هو أن الأخير (نظام الطبقات) نظام واضح .

### إدارة العمليات : Process Management

هي الإنتقال من نشاط إلى آخر حسب المقاطعة . أو هو برنامج I/O data state

مفاهيم :

Women ----- processor

Recipe ----- program

Ingredients ----- input data

Activity ----- process

إذا أصيبت المرأة أثناء إعدادها (الكيك) تقوم بعمل إسعافات أولية (first aid) وذلك لأن الإسعافات الأولية هي نشاط بمعنى عملية لها الأولوية في

التنفيذ وبعد أن تنتهي تواصل عملها من نفس النقطة التي توقفت فيها وهكذا وهذا ما يعرف بالبرمجة المتعددة (المتوازية) Multi programming

حالات العملية : process states : لكل عملية إحدى خمسة حالات هي :

(1) الحالة New : وهذه الحالة التي تعني أن العملية قد تم إنشاؤها (Create)

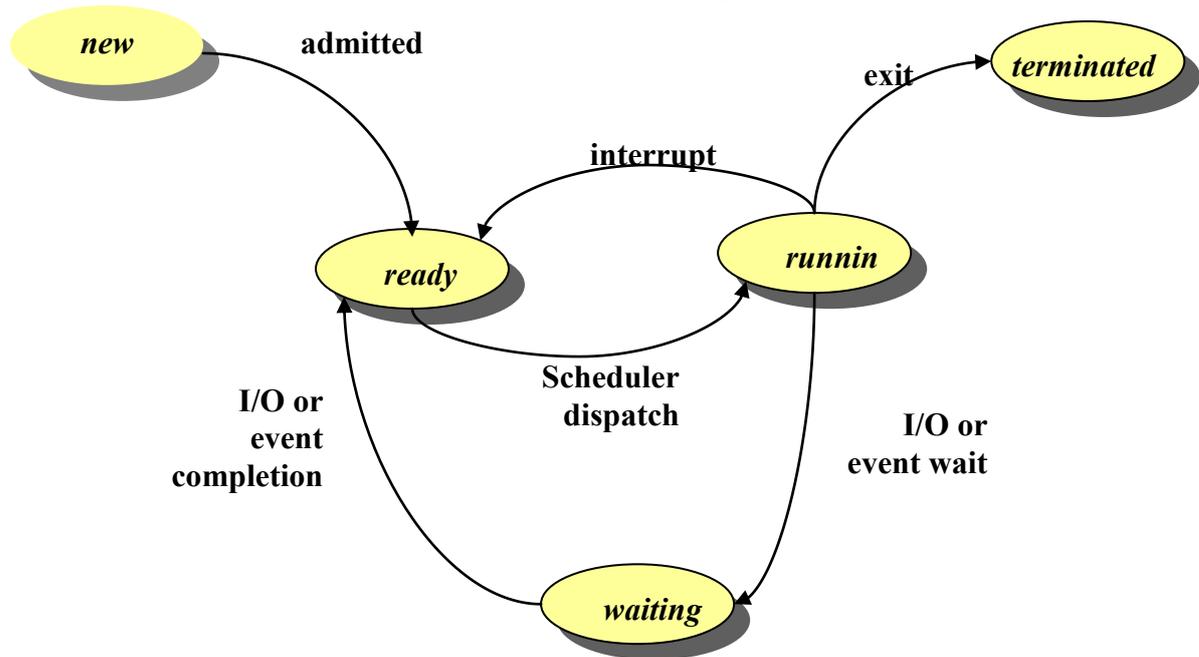
(2) الحالة Running : وتعني أن العملية في حالة تنفيذ وجودها في الـ CPU

(3) الحالة Waiting / Blocked : تعني أن العملية تنتظر حدوث حدث معين مثل إكمال بيانات أو إستقبال إشارة معينة أو غيرها .

(٤) الحالة Ready : تعني أن العملية مستعدة للإستدعاء من قبل المعالج

(٥) الحالة Terminated : تعني أن العملية تم تنفيذها تماماً (فرغت من التنفيذ).

\* تكون هنالك عملية واحدة في حالة تنفيذ في أي معالج وحيد بينما هنالك العديد من العمليات في حالة إستعداد وإنتظار .



من الـ Running إلى الـ Waiting تقوم بالنشاط العملية نفسها (process) أما في البقية يقوم بالنشاط الـ Scheduler dispatch



محاضرة رقم (٤)

### إدارة العمليات : Process Management

عبارة عن data structure module تمكن من تمييز كل process عن الأخرى (waiting , running) وغيرها ويستطيع تمييزها عن طريق أعداد صحيحة موجهة تعرف بـ process ID .

### كثلة تحكم العمليات / المهام : Process Control Block (PCB)

هي عبارة عن بنائيات أو هيكل بيانات (data structures) تخزن فيها كل المعلومات الخاصة بالعملية المعينة في شكل سجلات .

- أي عملية موجودة بالنظام بها (PCB) أو (process table) أو (task control block) . فكل عملية تدخل النظام تمثل بـ (PCB) وتخزن في الذاكرة في شكل سجلات .

- تخزن الـ PCB على سجلات سريعة جداً أسرع كثير من سرعة عناصر الذاكرة الأساسية حتى يتمكن نظام التشغيل من متابعة حالات التنفيذ . وتحتوي على كل المعلومات والبيانات اللازمة لنظام التشغيل لإدارة المهمة .

- أي عملية تدخل النظام وعند تنفيذها داخل الـ (PCB) يتم تحديد أي نوع تحتاجه من المصادر هل هو (S.W & H.W) ويكون موجود في ما يسمى الـ Resource S.W & H.W وتوضح العملية داخل نظام التشغيل بما يعرف بـ PCB

### مخطط كثلة تحكم المهام :

pointer	process state
process number	
program counter	
registers	
memory limits	
list of open files	
-	
Accounting information	

\* الـ pointer : هو المؤشر للـ pcb التالية .

\* الـ process state : حالة العملية هي ready & running & new .

\* الـ process no : هو رقم العملية .

\* الـ program counter : عداد برنامج التعليمات وبدل على عنوان التعليمات التالية التي ستنفذ للعملية .

\* الـ Register : المسجلات تتنوع المسجلات في الأرقام والأنواع حسب معمارية الحاسوب وتحتوي المسجلات على مخازن التجميع (accumulators) وفهارس المسجلات (register index) ومؤشرات المكذسات (stack pointer) وكل هذه المعلومات يجب حفظها عند حدوث أي مقاطعة

حتى تواصل العملية بصورة صحيحة . ( الـ register تسجل فيه حالة العملية والزمن الذي توقفت فيه ) .

\* معلومات عن جدولة وحدة المعالجة المركزية cpu scheduling information تحتوي على process priority ومؤشرات لصفوف الجدولة .  
\* الـ Memory limits : هي تحتوي على معلومات توضح حد الذاكرة (قيمة موقع العملية بالذاكرة) وقيمة الـ limit والـ base الخاصة بالمسجلات وعلى الـ page tables .

\* الـ List of open file : هي الملفات التي يتم فتحها في لحظة الـ I / O التي يتم تخصيصها للعملية .

\* الـ Accounting information : وتعني بها ساعة الـ CPU للبرامج في نفس اللحظة وكذلك عدد العمليات والزمن المستخدم .

### جدولة العمليات : Process Scheduling

توضيح : إن مفهوم البرمجة المتعددة Multi programming وإشراك الزمن Time sharing يعني إدخال عدة برامج بالذاكرة ومعالجتها معاً في نفس الوقت وتعني بالـ Time مشاركة الوقت بأن تتحول الـ CPU من عملية لأخرى في نفس الوقت .  
والهدف من البرمجة المشتركة (المتعددة) هو الاستفادة من وحدة المعالجة المركزية للحد الأقصى وذلك بجعلها في حالة تنفيذ دائم حتى تمكن المستخدم من التعامل مع أى برنامج أثناء تنفيذه .

– ملحوظة : لكل صف من صفوف الجدولة مؤشر يشير للعملية التالية وأى عملية لها PCB خاص بها .

### صفوف الجدولة (جدولة الصفوف) : Scheduling Queues

الصفوف تخزن كل الـ linkedlist وأى عملية لا بد من وجودها في صف معين . و لصفوف الجدولة أنواع منها :

(1) صف المهمة : Job Queue :

وهو صف توجد به كل عمليات النظام بمختلف حالاتها وهو عبارة عن هيكل بيانات .

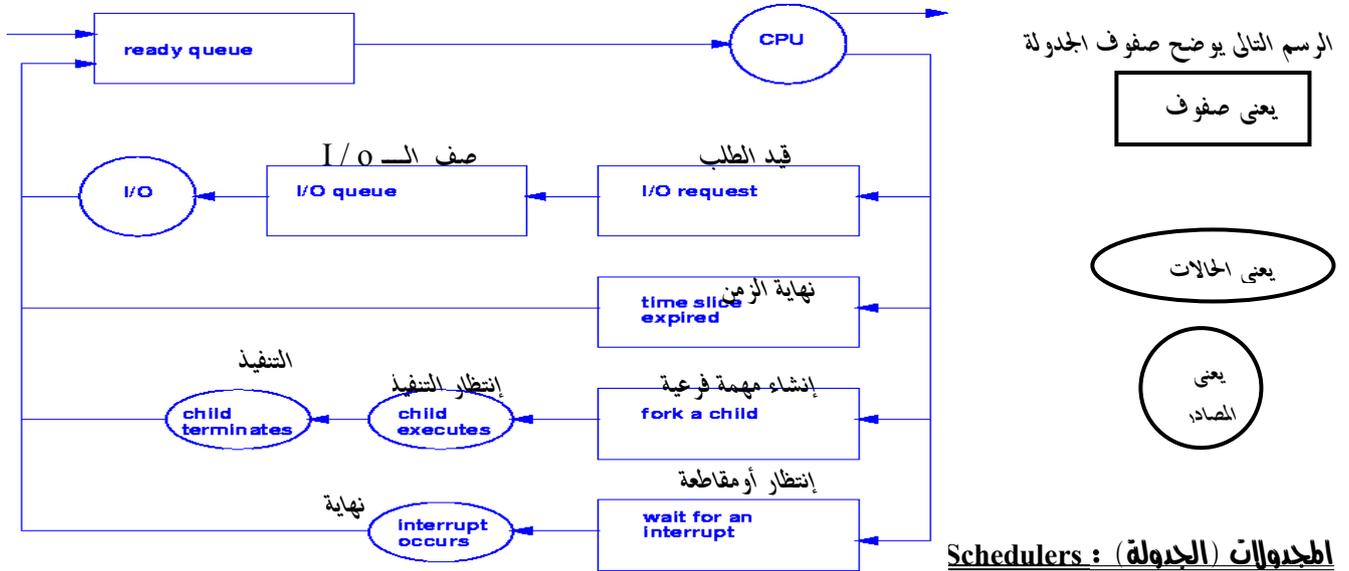
(2) صف الإستعداد : Ready Queue :

وهو صف توجد به العمليات الجاهزة للتنفيذ والتي تكون موجودة بالذاكرة الرئيسية في حالة إنتظار أو إستعداد.

(3) صف الأجهزة : Device Queue :

وهو صف توجد به كل عمليات I / O device الخاصة بالنظام .

توضيح : أثناء تنفيذ العملية في الـ cpu عند توقفها فجأة يكون هذا نتيجة مقاطعة أو حدوث طلب ( I / O Request ) فإذا كان المصدر غير متوفر أى تم طلبه من عملية أخرى فإن العملية يجب أن تنتظر حتى يتم تحرير ذلك المورد من العملية الأخرى والعمليات التي تكون في حالة إنتظار I / O device معين تعرف بـ device Request



يوجد نوعان منها في كل نظم التشغيل هما الأول Long Term Scheduler والثاني Short Term Scheduler أو CPU Scheduler وفي بعض نظم التشغيل يوجد نوع ثالث يسمى Medium Term Scheduler وبه ما يسمى الـ Swapping وتعني به حركة العمليات من وإلى الذاكرة .

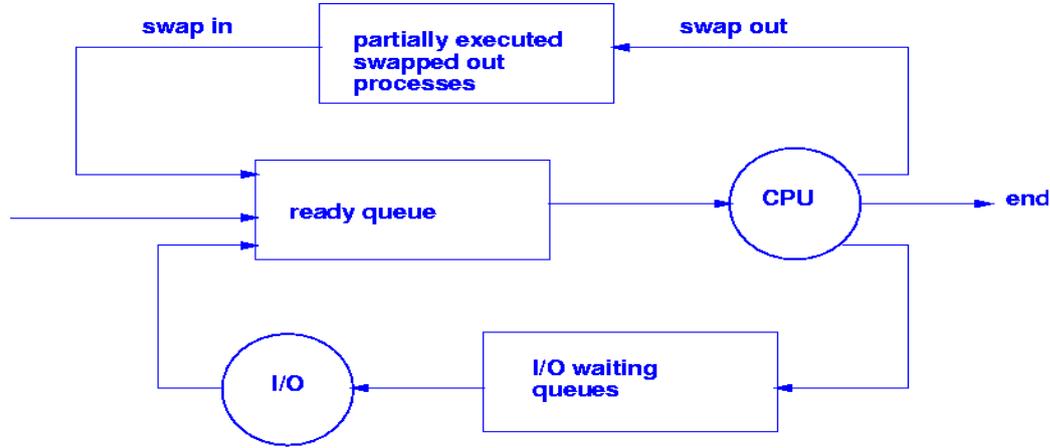
\* في الـ batch system لا يتم تنفيذ كل العمليات حالاً والتي لم يتم تنفيذها يتم تجميعها وتوضع في معدة تخزين مثلاً القرص حتى تنفذ لاحقاً وذلك بواسطة الـ long ثم بعد ذلك بواسطة الـ short .  
ملحوظة : أى عملية تدخل النظام للتنفيذ موجودة في صف المهمات (job queue) . ولكنها فعلياً موجودة بالذاكرة .

\* الـ Long Term Scheduler : هي الآلية التي تقوم بإختيار نقل العمليات التي يجب تنفيذها من الذاكرة إلى الـ cpu .

\* الـ Short Term Scheduler : هي الآلية التي تقوم بإختيار العمليات التي يتم تنفيذها من حالة الـ Ready إلى الـ Running .

\* الـ Medium Term Scheduler : تقوم بالـ Swapping وتعنى تحريك العملية من الذاكرة وإعادتها مرة أخرى وإعادتها وهكذا وذلك للإستفادة القصوى من الذاكرة .

الرسم التالي يوضح الـ Medium Term (Time-sharing) Scheduler



\* الـ context switch : هو عملية تحويل وحدة المعالجة المركزية من عملية لأخرى بعد حفظ العملية القديمة وتحميلها لأجل العملية الجديدة .

(١) سرعة الذاكرة

(٢) عدد المسجلات التي يجب نسخها

(٣) وجود تعليمات خاصة مثل تعليمة تخزين المسجلات ويتراوح زمن العملية context switch بين ١ إلى ١٠٠٠ ميكرو ثانية .

يوجد نوعان من المجدولات للـ process هما :

(١) CPU poundprode وسميت بهذا الإسم لأنها تستخدم وحدة المعالجة المركزية (تعمل بطريقة حسابية) .

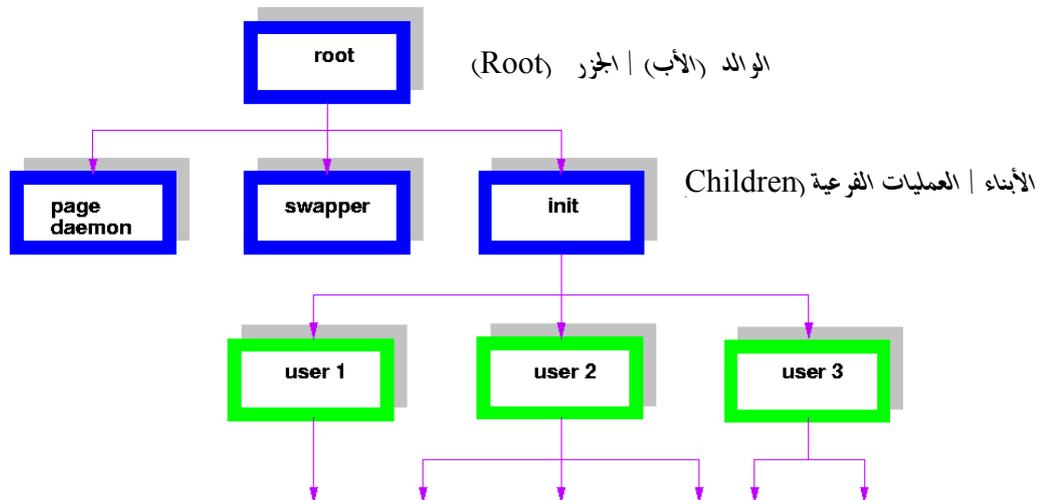
(٢) I / O boundprode سميت بذلك لأنها تستخدم I / O في أغلب الأحيان .

### العمليات على المهام / العمليات : Operators On Process

تنفيذ العمليات يتم بصورة متزامنة ويتم أنشاؤها وتدميرها بصورة ديناميكية .

(1) إنشاء العملية : Process Creation :

عند إنشاء أى عملية نستخدم Sys Call تسمى Create و كما هو معلوم أن أى عملية يمكن أن تقوم بمجموعة عمليات أخرى تسمى العملية الرئيسية بالأب Parent process والعمليات المولدة منها تسمى الأبناء Children process وبذلك يكون لدينا شجرة من العمليات كمثال عملية الطباعة عملية رئيسية أنشأت عمليات أخرى كمثال إدخال الورق للطابعة



\* الموارد :

لفرض أن العملية الأب تسمى (A) والعمليات المولدة (C, D, E) وأن العملية (C) ولنكن في الرسم (init) وهكذا ولكل عملية إنشاء عمليات فرعية (مولدة) تحتاج مجموعة موارد أخرى قد تكون S.W & H.W وهناك ثلاثة احتمالات لهذه الموارد هي :

- (١) أن العملية (C) لا تتشارك مع (A) في الموارد .
- (٢) أن العملية (C) تتشارك مع (A) في بعض الموارد .
- (٣) أن العملية (C) تتشارك مع (A) في كل الموارد .

#### \* التنفيذ :

\* يتم التنفيذ بصورة متزامنة ولكن هنالك احتمال آخر أن تنفذ العملية الـ (Sub) وهي الفرعية أولاً ثم تتم مواصلة العمل .

\* تنفيذ العملية المولدة بجزئية من موارد العملية الأب يمنع أى عملية أخرى من إنشاء العديد من العمليات الأخرى منها وذلك بالطبع يؤدي إلى تحميل زائد للنظام .

#### \* التخزين :

من الممكن أن يتشارك الأب والأبناء نفس المساحة في الذاكرة أو أن يكون للأب مساحة خاصة به في الذاكرة أما في مساحة العنوان يكون هنالك احتمالان هما :

(١) تكون العملية المولدة عبارة عن نسخة من العملية الأب . (٢) تكون العملية المولدة لديها برنامج خاص محمل بها .

\* في نظام يونكس يتم إنشاء العملية الجديدة باستخدام نداء نظام يسمى **Fork system call** .

ملحوظة : إذا إستمرت عملية توليد العمليات سوف يحدث **Over load** لذا يتم إيقاف ذلك بشرط محدد مرتبط بالموارد .

أثناء التنفيذ : يكون هنالك احتمالان هما :

الأول : تواصل العملية الأب التنفيذ بصورة متزامنة مع العملية المولدة . الثاني : تنتظر العملية الأب حتى يتم تدمير بعض أو كل العمليات المولدة .

#### (2) تدمير العملية : **Process Termination** :

\* يتم تدمير العملية بعد أن تنتهي من تنفيذ كل التعليمات الخاصة بها عن طريق **Sys call** لإنهاء العمليات يسمى **Exit** (إنهاء) ويتم تحرير كل

المراد الخاصة بها عن طريق نظام التشغيل

\* في أحيان أخرى نستخدم نداء نظام يسمى **Abort** (قطع) ولكن القطع لا تستخدم إلا عن طريق العملية الأب **Parent** وذلك لمنع المستخدمين

من تدمير وظائف بعضهم البعض .

#### أسباب تدمير العمليات :

(١) إنتهاء المهمة للإستفادة من مساحة الذاكرة (تنفيذ العملية) .

(٢) إنتهاء الوقت المخصص للمورد .

(٣) تدمير العملية الأب بواسطة **Abort** .

\* العديد من نظم التشغيل تقوم بتدمير العمليات المولدة بصورة مباشرة بعد تدمير العملية الأب



#### محاضرة رقم (٥)

العمليات المتعاونة Co operating process : نظام التشغيل يقوم بتقسيم الـ process إلى نوعين هما :

(١) عمليات مستقلة Independent process (٢) عمليات متعاونة Co operating process

العمليات المستقلة هي عملية لا تؤثر ولا تتأثر بالعمليات الأخرى أي إنها لا تتشارك معها في أي نوع من البيانات .

العمليات المتعاونة هي عملية تؤثر وتتأثر بالعمليات الأخرى لأنها تتشارك معها في البيانات أو المصادر .

#### ما الفائدة من المتعاونة ؟

(١) المشاركة في المعلومة : أحياناً يكون هنالك العديد من المستخدمين بحاجة إلى المعلومة نفسها ( ملف مثلاً )

(٢) السرعة في إنجاز المهام Speed up : إذا كنا نريد القيام بمهمة بسرعة أكبر فإنه يمكن تجزئتها إلى عدة مهام أصغر وسيتم تنفيذ هذه المهام بصورة

متوازية مع بعضها البعض وهذا يحدث إذا كان هنالك عناصر معالجة متعددة .

(٣) التوافقية Convenience : أحياناً يحتاج المستخدم إلى أكثر من مهمة في وقت واحد مثلاً قد يحتاج أن يطبع أو يترجم أو يجرر في وقت واحد .

يقسم البرنامج إلى مجموعة عمليات | مهام وذلك لوجود مجموعة معالجات متعددة بالجهاز

\* لتوضيح مفهوم التعاون بين العمليات مثلاً مشكلة المنتج والمستهلك .

## مشكلة المنتج والمستهلك Producer & consumer Problem

هنالك عملية منتجة تنتج معلومات يتم إستهلاكها بواسطة المستهلك ويتم تخزينها في **Buffer** بسعة معينة وعندما يمتلئ توقف عملية الإنتاج حتى يتم الإستهلاك وإذا كان الـ **Buffer** فارغ لا يتم إستهلاك .

إذاً:

يتم الإنتاج أولاً ثم الإستهلاك بعد ذلك . مثلاً برنامج الطباعة ينتج حروف تستهلك بواسطة الطباعة والمترجم ينتج شفرات تجميعية يتم إستهلاكها بواسطة الحاسوب بالـ **Assembler** وهكذا .

وهناك مشكلتان لهذه العملية في الـ **Buffer** هما :

(١) قد يكون الـ **Buffer** غير محدود **Unbound Buffer** لذلك يكون في حالة إنتاج دائم .

(٢) قد يكون الـ **Buffer** محدود **Bound Buffer** وفي هذه الحالة تكون العملية قد توقفت لمحدودية الـ **Buffer** .

ملحوظة في الحالة الثانية إذا لم يتم إنتاج فإن المستهلك تكون في وضع الـ **waiting** وكذلك إذا لم يتم إستهلاك فالمنتج ينتظر .

\* المخزن يتم دعمه عن طريق نظام التشغيل او عن طريق تشفير واضح من قبل مبرمجي البرامج التطبيقية .

\* تقوم العملية المنتجة بإنتاج مفردة واحدة في أثناء إستهلاك العملية المستهلكة لمفردة أخرى كما يجب أن تعمل العملية المنتجة والمستهلكة بصورة

متزامنة **synchronized** حتى لا تحاول العملية المستهلكة إستهلاك مفردة لم يتم إنتاجها بعد .

### ورقة بحثية رقم (٣)

ما هي الخيوط Threads ؟

## جدولة وحدة المعالجة المركزية: CPU Scheduling

\* تعريف :

هي إختيار أحد العمليات من صف الـ **ready Queue** إلى مرحلة التنفيذ وذلك بواسطة إحدى خوارزميات الجدولة .

الهدف من ذلك : جعل الـ **CPU** مشغولة دوماً .

توضيح : تعتبر جدولة وحدة المعالجة المركزية من أهم الأشياء بالنسبة لنظم التشغيل متعددة البرمجة حيث أنه عن طريق تحويل وحدة المعالجة المركزية من عملية لأخرى تزيد إنتاجية الحاسوب والهدف الأساسي من البرمجة المتعددة هو الإستفادة القصوى من وحدة المعالجة المركزية وذلك بجعلها في حالة تنفيذ دائم للعمليات ،،،،،،،،،،،، في نظم الحاسوب البسيطة عندما تكون العملية في حالة تنفيذ تكون وحدة المعالجة في وضع مستقر ولا تنفذ أى عمل آخر طيلة فترة الإنتظار .

### خوارزميات الجدولة Scheduling Algorithms:

مفاهيم أساسية : قبل التعرف على خوارزميات الجدولة لا بد أن نتعرف على هذه المفاهيم الأساسية وهي :

\* جدولة عدم الإيقاف **Non-preemptive scheduling** وهي جدولة عدم إيقاف المهمة .

\* جدولة الإيقاف (الجدولة الشفعية) **preemptive scheduling** وهي جدولة الإيقاف المهمة.

\* البرمجة المتعددة : **Multi Programming** : يعني إدخال عدة برامج بالذاكرة ومعالجتها معاً في نفس الوقت

\* الـ **Job / batch** : برامج يتم تنفيذها من غير المستخدم مثال برامج النظام .

\* الـ **User** : برامج تحتاج في تنفيذها للمستخدم مثال البرامج التطبيقية .

\* الـ **CPU burst cycle** : الفترة التي تقضيها العملية داخل وحدة المعالجة وهو عادة أقل بكثير من زمن الـ **I / O** .

\* الـ **I / O burst cycle** : الفترة التي تقضيها العملية في وحدات الـ **I / O** .

\* مفهوم المنجز **Dispatcher** هي آلية تعطي وحدة المعالجة المركزية التحكم في العمليات المختارة بواسطة الـ **Short Term Scheduler**

- يجب أن يكون المنجز سريعاً جداً عند التحول من عملية لأخرى .

- يعرف الزمن الذي يستغرقه المنجز لإيقاف عملية وبدء أخرى بـ **Dispatch latency** .

\* تعريف :

جدولة الإيقاف هي: مقاطعة تحدث للعملية أثناء تنفيذها تتسبب في وقفها وتخريبها من وحدة المعالجة ولذا تنزع الـ **CPU** منها وتعطى لعملية أخرى .

ملحوظة: لا يعني وجود العملية في أول الصف تنفيذها أولاً وربما نعامل هذا العنصر على إنه **LIFO** أو نظام **Linkedlist**

\* الجدولة الناجحة لوحدة المعالجة المركزية تعتمد على :

(١) تنفيذ عملية تحوى دورة لتنفيذ **cpu** و **I / O wait**

(٢) عملية تناوب بين الحالتين

أما تعريف جدولة عدم الإيقاف الـ **Non** عكس ما سبق ذكره تماماً .

ومقابل جدولة الإيقاف و عدمها يوجد لدينا ما يسمى بال preemptive process & Non preemptive process

\* عندما تكون وحدة المعالجة المركزية في وضع مستقر (idle) يقوم نظام التشغيل باختيار إحدى العمليات من ال ready queue للتنفيذ ويخصص لها وحدة المعالجة وال ready queue ليس fifo بالضرورة ويمكن أن يكون fifo أو priority أو unordered linked list tree / متى يتم اتخاذ قرار جدولة وحدة المعالجة المركزية في إحدى أربع حالات هي :

(١) في حالة تحويل العملية من Running إلى Waiting (تحتاج إلى I/O)

(٢) في حالة تحويل العملية من Running إلى Ready (في حالة مقاطعة)

(٣) في حالة تحويل العملية من Waiting إلى Ready (توفر لها I/O)

(٤) في حالة تدمير العملية Terminate (إنهاء المهمة)

\* يلاحظ أن الجدولة في الحالة الأولى والرابعة هي Non preemptive أما الحالة الثانية والثالثة هي preemptive . تستخدم جدولة عدم الإيقاف (Non preemptive) في بيئة ويندوز حيث أنها الطريقة الوحيدة التي يتم إستخدامها على برامج عتاد معينة وذلك لأنها لا تتطلب عتاد معين مثل ال timer .

مصطلحات مهمة :

\* CPU utilization استخدام المعالج CPU utilization وتعني به إبقاء المعالج مشغولاً بقدر الإمكان بنسبة معينة مثلاً ٤٠٪ والزمن الفعلي له (٤٠ - ١٠٠٪).

\* Waiting time زمن الانتظار هو متوسط الفترة الزمنية لبقاء العملية في حالة ال Waiting حتى تتوفر لها ال CPU .  
خوارزمية الجدولة لا تؤثر في الزمن المستغرق بالنسبة للعملية في تنفيذ أدنى عمل I/O ولكنها تؤثر في الزمن الذي تستغرقه في صف ال ready أي زمن الانتظار .

\* Through Put الطاقة الإنتاجية تعني به عدد العمليات المنفذة في وحدة زمنية معينة . (الطاقة الإنتاجية محددة أو معدل العمليات المنجزة)  
مثال: إذا كان لدينا عمليات موضحة من A إلى D حسب الزمن المخصص لكل عملية فإن الطاقة الإنتاجية هي مجموع العمليات

العملية Process	الزمن المخصص Burst time
A	5
B	2
C	5
D	3

العمليات في ال 15 وحدة زمنية عددها أربعة

الإنتاجية = مجموع زمن الانتظار ÷ عدد العمليات =  $4 \div 15 = 3.75$  وحدة زمنية

\* الزمن الدوري Turnaround time: هي الفترة الزمنية التي تستغرقها العملية حتى نهاية التنفيذ .

تعريف آخر : هي مجموع الفترات الزمنية التي تمر بها العملية كمثل: فترة الانتظار حتى تصل ال CPU وفترة ال Running وفترة ال Ready Queue وفترة ال I/O Request أو : هو يعني زمن الانتظار + زمن التشغيل .  
وبالرجوع للمثال السابق فإن الزمن الدوري هو :

$$A = 0 + 5 = 5 \quad B = 5 + 2 = 7 \quad C = 7 + 5 = 12 \quad D = 12 + 3 = 15$$

وبالرجوع للمثال السابق فإن زمن الانتظار هو :  $A = 0 \quad B = 5 \quad C = 7 \quad D = 12$

\* زمن الاستجابة Response time : هو الفترة الزمنية من بداية الطلب حتى حدوث أول إستجابة للطلب وهنا يجب الإشارة إلى أنه ليس الزمن الذي يستغرق لإخراج تلك الإستجابة وبالتالي يجب زيادة (إستخدام المعالج والطاقة الإنتاجية) وتقليل ( الزمن الدوري وزمن الانتظار وزمن الإستجابة)  
\* الأولويات priorities: كل عملية تحمل عنوان به إشارة تحدد أفضلية المهمة وأولويتها في التنفيذ وتقوم خوارزمية الجدولة باختيار العمليات ذات الأولوية العالية .

\* الزمن المخصص Burst time : هو الزمن الذي يستغرقه المعالج لإنهاء من المهمة ويسمى أيضاً (service time)

خوارزميات الجدولة :

هنالك ست خوارزميات للجدولة هي :

(١) الأول في الوصول الأول في المعالجة (FCFS) First Come First Serve Scheduling

(٢) أقصر المهمات الأول في المعالجة (SJFS) Shortest Job First Scheduling

(٣) خوارزمية الأولوية Priority Scheduling

(٤) التخصيص الدوري Round Robin Scheduling

## (٥) الصفوف ذات المستويات المتعددة Multi Level Queues Scheduling

### (٦) صفوف التغذية الراجعة ذات المستويات المتعددة Multi Level Feedback Queues Scheduling

الأهداف :

(١) العدل Fairness : وهو توفير الـ CPU لكل العمليات بغض النظر عن إتمام المهمة أو عدمه (إعطاء فرصة لأي عملية).

(٢) التوازن Balance : ونعني به توازن النظام عموماً (إبقاء كل أجزاء النظام مشغولة تؤدي عمل ما).

(٣) إنفاذ سياسة نظام التشغيل Policy enforcement : ونعني به تنفيذ سياسة معينة للتعامل مع نظام التشغيل.

### (٧) الأول في الوصول الأول في المعالجة (FCFS) First Come First Serve Scheduling

العملية التي تطلب الـ cpu أولاً يتم تخصيصها لها أولاً.

مثال : إذا كان لدينا العمليات التالية والتي وصلت على الترتيب الموضح p1 أولاً وهكذا

process	Burst time
P1	24
P2	3
P3	3

يمكن تمثيلها بمخطط جانث (Gant Chart) كما يلي :

P1	P2	P3	
0	24	27	30

زمن الإنتظار بالنسبة لـ p1 هو 0 و بالنسبة لـ p2 هو 24 و بالنسبة لـ p3 هو 27

ومتوسط زمن الإنتظار (average waiting time) =  $(27 + 24 + 0) \div 3 = 17$  مللي ثانية.

– مثال آخر : إذا وصلت العمليات على الترتيب p2 ثم p3 ثم p1 يكون زمن الإنتظار أقل =  $3 \text{ mlsc} = (3 + 6) \div 3$

P2	P3	P1	
0	3	6	30

\* هذه الطريقة جيدة في حالات المهمات الطويلة ولكنها غير عادلة بالنسبة للمهام الصغيرة كما في المثال الأول.

\* تعتبر هذه الخوارزمية من خوارزميات إيقاف الجدولة (Non preemptive) فعندما يتم تخصيص الـ cpu لعملية معينة فإنها تحتفظ بالـ cpu حتى تقوم بتحريرها وذلك بتدميرها أو بطليها لـ I/O.

\* إنتظار العمليات الصغيرة في الـ ready queue حتى يتم الإنتهاء من الكبيرة وحتى تحرر وحدة المعالجة

\* غير مناسبة لأنظمة الـ time sharing system حيث أن المهم هو أن يتشارك كل مستخدم والـ cpu مع المستخدمين الآخرين وبالتالي ستكون هنالك مشكلة إذا احتفظت العملية بوحدة المعالجة لفترة طويلة.

### (٨) أقصر المهمات الأول في المعالجة (SJFS) Shortest Job First serve Scheduling

تعتمد على طول الزمن المخصص (cpu burst) حيث نختار العملية التي يكون لها أقصر طول زمن مخصص وإذا كانت هنالك عمليتان لهما نفس طول

الزمن فإننا نستخدم في هذه الحالة خوارزمية الأول في الوصول (FCFS) لحل هذا التعقيد.

مثال : مثال : إذا كان لدينا العمليات التالية والتي وصلت على الترتيب الموضح p1 أولاً وهكذا

process	Burst time
P1	6
P2	8
P3	7
P4	3

يمكن تمثيلها بمخطط جانث (Gant Chart) كما يلي :

P4	P1	P3	P2	
0	3	9	16	24

نحسب متوسط زمن الانتظار =  $(3 + 6 + 9 + 0) \div 4 = 7$  مللي ثانية.



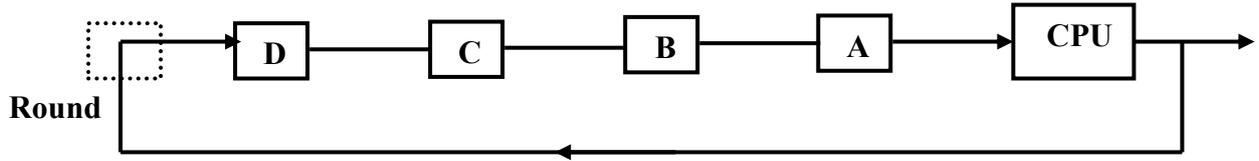
\* تقنية الـ ging : هي تقنية تعمل على زيادة الأولوية بالنسبة لتلك العمليات التي إنتظرت في النظام لفترة طويلة بعد كل فترة زمنية معينة .

### (. ) التخصيص الدوري Round Robin Scheduling

\* تم تصميمها خصيصاً لأنظمة إشتراك الزمن (time sharing systems) وهي تشبه خوارزمية الأول في الوصول ولكن بها وحدة زمنية صغيرة تسمى الـ **time quantum** أو الـ **time slice** .

\* تتم معالجة الـ **ready queues** كـ **Circler queues** حيث يقوم الـ **cpu scheduling** بالذهاب إلى الـ **ready queues** ويقوم بتخصيص وحدة المعالجة المركزية لكل العمليات لفترة زمنية **quantum** معينة .

\* تتم معاملة الـ **ready queues** كـ **Fifo queues** حيث تتم إضافة العمليات الجديدة في مؤخرة الصف ويقوم بوضع مؤقت يعمل على المقاطعة بعد وحدة زمنية واحدة (**1 time quantum**) وقد يحدث أحد أمرين هما : الأول : إما أن يكون الزمن المخصص الخاص بالعملية أقل من الـ **quantum** وفي هذه الحالة فإن العملية نفسها تقوم بتحرير وحدة المعالجة المركزية ويقوم الجدول باختيار العملية التالية من الصف والأمر الثاني هو: أن يكون زمن التخصيص أكبر من الـ **quantum** فتحدث مقاطعة وتحدث عملية **Context Switch** وسوف توضع العملية في مؤخرة الصف وبالتالي يقوم الجدول باختيار العملية التالية من الصف .



process	Burst time
P1	24
P2	3
P3	3

**time quantum = 4 milliseconds** & **Average waiting time = 17/3 = 5.6 mlsc**

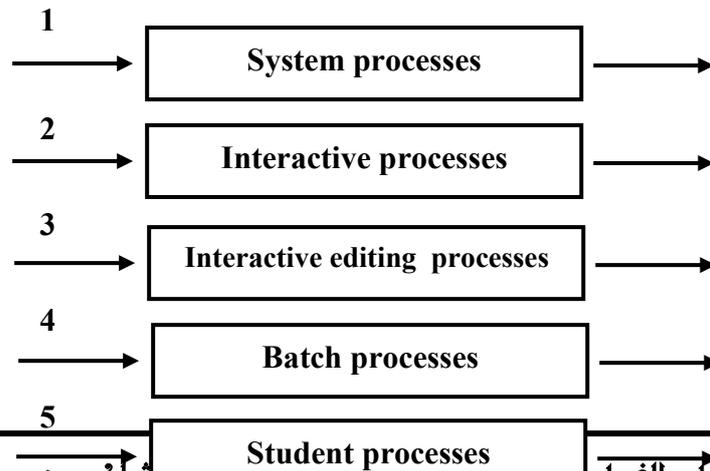
P1	P2	P3	P1	P1	P1	P1	P1	
0	4	7	10	14	18	22	26	30

\* يجب أن تكون الفترة المنوحة لكل مهمة مناسبة لأنه إذا كانت كبيرة لكل المهمات فإن خوارزمية الأولوية تصبح خوارزمية الأول في الوصول في لحظة من اللحظات . وإذا كانت الفترة قصيرة تكثر عمليات التبديل بين المهمات وهذا يؤثر على كفاءة النظام (إذا كانت صغيرة فإن منهجية RR تعرف بـ **Processor sharing** .

(. ) الصفوف ذات المستويات المتعددة Multi Level Queues Scheduling يتم تقسيم العمليات إلى :

(1) foreground (interactive) processes (2) background (batch) processes

يختلف كل من هذين القسمين في إحتياجاته من ناحية الجدولة مثلاً تقوم خوارزمية الصفوف ذات المستويات المتعددة بتقسيم الصف (ready queue) إلى العديد من الصفوف الأخرى والتي تحتوى على العديد من العمليات ويكون لكل صف خوارزمية جدولة خاصة به حسب نوع العمليات وحسب أولويتها ... ألخ فمثلاً **background queue** قد تتم جدولتها بـ **RR** بينما تتم جدولة **foreground queue** بخوارزمية **FCFS** مثلاً بالإضافة أنه لا بد من وجود جدولة بين الصفوف وتنفذ عادة كـ **fixed priority preemptive scheduling**



\* كل صف تكون له أولوية أعلى من أولوية الصفوف التي تقع بالأسفل مثلاً لا تستطيع أى عملية موجودة في الصف *Batch processes* أن تبدأ في التنفيذ إذا لم يتم تنفيذ كل العمليات التي توجد في الصفوف ( ١ ، ٢ ، ٣ ) وأصبحت فارغة تماماً .  
 \* إذا دخلت *Interactive processes* إلى الـ *Ready queue* أثناء تنفيذ الـ *Batch processes* في هذه الحالة ستوقف الـ *Batch processes* عن التنفيذ .  
 \* يوجد *time slice* بين الصفوف حيث أن كل صف يكون له وقت محدد في وحدة المعالجة .

### ( ٦ ) صفوف التغذية الراجعة ذات المستويات المتعددة Multi Level Feedback Queues Scheduling

\* في الخوارزمية السابقة نلاحظ أن الخوارزمية إذا كانت *background* أو *foreground* فإنها لا تستطيع التنقل بين الصفوف بينما نجد أن هذه الخوارزمية تسمح بذلك .  
 \* تلخص الفكرة في تقسيم العمليات اعتماداً على خصائص الـ *cpu burst* لكل عملية فإذا كانت ذات زمن تخصيص كبير جداً فإنه يتم تحريكها إلى صف أقل أولوية وهذه الطريقة تجعل الـ *I / O bound* والـ *Interactive* موضوعة في صف أكبر أولوية والعملية التي تكون في حالة إنتظار طويلة في صف أقل أولوية يتم نقلها إلى صف أكبر أولوية مما يمنع حدوث المجاعة (*Starvation*) .  
 توضيح : إنتقال العمليات من صف إلى آخر حسب الـ *quantum* وإذا لم تكتمل العملية تنتقل إلى مؤخرة الصف الذي يليه .  
 - لا تنفذ الصفوف السفلى بخوارزمية الأول في الوصول إلا إذا كانت الصفوف العليا (التي قبلها) فاعية تماماً



اللهم صلى على النبي وآله ما إهتزت الأثلاث من نفس الصبا  
 اللهم صلى على النبي وآله ما كوكب في الجوف قابل كوكبا  
 اللهم صلى على النبي وآله ما قال ذو كرم لضيف مرحبا

\*\*\*\*\*

إذا ركلك الناس من الخلف فأعلم أنك في المقدمة  
 وأزد على علمك أن ناج الفيصر لا يفهم من الصراع  
 فالفشل في التخطيط يفود إلى التخطيط للفشل

صدى - يونيو ٢٠١١م - أسألكم خاص الدعاء لى ولوالداى



لا تتجاهل المحاضرات أثناء العام الدراسى  
 //// وتتسارع بالإحتهاد لحظة قدوم  
 الإمتحان //// فهما أمران لا يستقيمان  
 أبد //// تمنع بيسر وإقرأ بهدوء البال  
 وراحة النفس وإستمع بفكر دواخلك ولا  
 تنحصر فى بوتقة اللجوء إلى الأسباب أو  
 ما شابه ذلك //