جامعة حماه
السنة الثالثة
قسم تقنيات الحاسوب
نظم التشغيل الشبكية

المحاضرة الثالثة

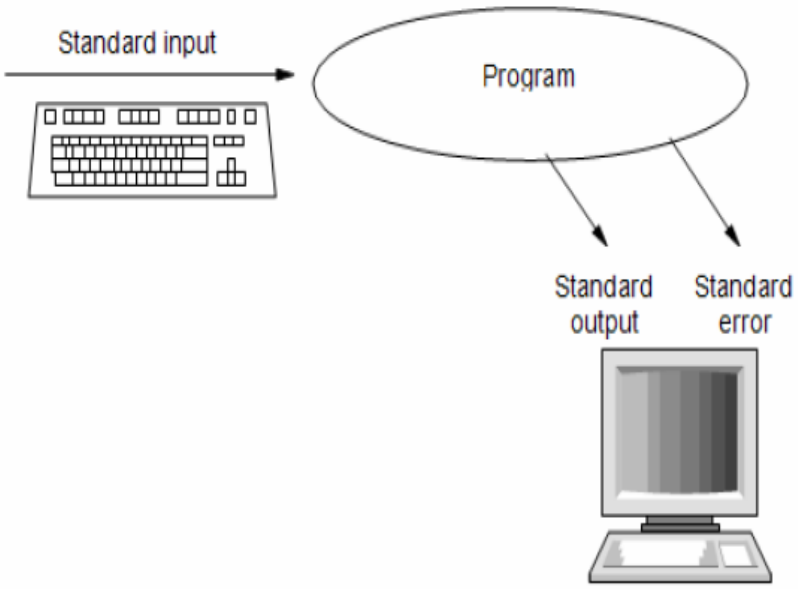
pipeline

اعداد وتقديم

م. مها رضوان جحا
م.مصطفى وهبة
م. فائزة حاتم

Use Unix Streams, Pipes and Redirects

أي برنامج عندما يعمل فإنه يفتح ثلاث ملفات:



Standard input : stdin

و هو ملف الدخل للبرنامج , نرمز له بالاختصار stdin ، و هو افتراضيا الكيبورد.

Standard output: stdout

و هو ملف الخرج للبرنامج ، نرمز له بالاختصار stdout ،و هو افتراضيا الشاشة.

Standard Erorr: stderr

و هو ملف خرج للبرنامج يشبه stdout لكنه يستخدم لطباعة رسائل الخطأ و رسائل التحذير ،نرمز له بالاختصار stderr ، و هو افتراضيا الشاشة.

لا تخلط بين stdout و stderr فكل منهما ملف مستقل ،و سيتوضح لنا ذلك لاحقا.

-عند فتح ملف فإنه يعيد رقم ، يسمى واصف الملف file descriptor و هو مؤشر إلى مصفوفة معلومات الملف في الذاكرة ، هذه الأرقام بالنسبة للملفات القياسية السابقة ثابتة دوما و هي كالتالي :

Name	Descriptor
Standard input	0
Standard output	1
Standard error	2

Pipe : |

و هو قناة لتحويل خرج برنامج إلى دخل برنامج آخر ، و بذلك نستطيع تنفيذ سلسلة برامج متتالية (نستخدمه من خلال |)

الفكرة هنا : من الأفضل أن لا نصمم برنامج يؤدي جميع الوظائف (لأنه لن يكون بقوة البرنامج المتخصص)،و بدلا من ذلك نقسم البرنامج إلى برامج جزئية خرج أحدها هو دخل للتالي.

أي بدلا من جعل خرج الملف هو ملف الخرج الأساسي stdout ،نجعل خرج البرنامج هو دخل للبرنامج التالي، و يفيد ذلك في :

- التخصص : تقسيم البرنامج إلى برامج جزئية ،يقوم بكتابتها مبرمجين كلٌ في مجال

خبرته و بالتالي المبرمج ليس بحاجة للإلمام بكل متطلبات البرنامج إنما بجزء بسيط ، مما يعطي قوة للبرنامج.

- فصل المهام التي تتكرر دائما :بحيث يمكن استخدام خرجها كدخل في أي برنامج.

-لا يمكن تخيل Unix بدون Pipe ،وهي القاعدة الثانية .

القاعدة الأولى : every thing is a file

القاعدة الثانية: Unix is unthinkable without a pipe

عملية إعادة التوجيه :redirection

- أنت قادر على توجيه أي خرج لأي مكان و يمكن توجيه الدخل من أي مكان.
- نستطيع إجراء عملية redirection لأي عملية كتابة (على stdout أو stderr)

مثلا : يمكن تغيير توجيه خرج برنامج من الشاشة إلى ملف لحفظه.

- يمكن إجراء عملية redirection لأي عملية قراءة من ملف الكيبـــورد stdin إلى ملف (أي يتم القراءة من ملف و ليس من الكيبورد).

مثال:

```
$ls  /etc
```

سيكون الخرج طويل جدا،لكن هناك برنامج less لتنظيم العرض على صفحات .

و بالتالي يمكن تحويل خرج الأمر السابق إلى دخل less ليقوم بتنظيم العرض و ذلك من خلال الـــ pipe | (shift+\)

$ls /etc | less

نحن نعمل الآن ضمن less (يمكنك الخروج بالضغط على Q).

Head :

وظيفته :نمرر له دخل input فيعرض عدد من الأسطر من أول الملف (افتراضيا10)

مثال :

لعرض محتويات الملف passwd على الشاشة .

$cat /etc/passwd

لعرض أول عشرة أسطر من الملف

$cat /etc/passwd | head

لعرض أول n سطر من الملف : **head –n**

مثلا : لعرض أول أربعة أسطر من الملف passwd .

$cat /etc/passwd | head -4

Tail :

وظيفته : عرض عدد من الأسطر من آخر الملف (افتراضيا 10)

$cat /etc/passwd | tial

لعرض آخر n سطر من الملف : **tial -n**

مثلا : لعرض آخر أربعة أسطر من الملف passwd .

$cat /etc/passwd | tial-4

مثال :

لعرض السطر الرابع من الملف

```
$cat  /etc/passwd | head -4 | tail -1
```

هذا يكافئ: نأخذ أول أربعة أسطر ثم نأخذ السطر الأخير منها .

أي أن خرج البرنامج head هو دخل للبرنامج tail .

: Cut

ملاحظة :

إن بنية الملفات في Unix على شكل جداول مما يسمح بأخذ عمود منها ، و هذا يفيد في الإحصائيات و سهولة الحصول على المعلومات ،مثلا : يمكننا الحصول علـــى أسمـــاء المستخدمين في النظام بأخذ العمود الأول من الملف /etc/passwd

-إن وظيفة برنامج cut هو اقتطاع حقل (أو حقول) من ملف معين حيث نحدد له الفاصل بين الحقول (لمعرفة نهاية الحقل) ،و نستخدم معه الخيارات التالية :

-f	(fileds)	تحديد رقم الحقل (أو الحقول) المراد اقتطاعها
-d	(delimiter)	تحديد الفاصل بين الحقول

مثال:

للحصول على أسماء المستخدمين (الحقل الأول من /etc/passwd)

```
$cat  /etc/passwd | cut –f  1  -d  :
```

Sort :

و هو برنامج متخصص بالفرز يمكن استخدامه مع أي برنامج آخر (فرز خرج برنامج)

مثلا : فرز أسماء المستخدمين أبجديا

```
$cat  /etc/passwd | sort
```

نستخدم مجموعة من الخيارات :

من أجل الفرز تنازليا	-r
جعل الفرز حساس لحالة الأحرف	-f
حذف الأسطر المكررة	-u

(يقوم بحذف الأسطر المكررة أثناء العرض و لكنه لا يؤثر على الملف الأصلي طبعا)

ملاحظة :

قم بإنشاء ملف .. كرر فيه بعض الأسطر، و جرب الخيار الأخير ..

إذا لم يقم الخيار الأخير بحذف بعض الأسطر المكررة في الملف فلا تشكك بصحة كلامي، لأنه غالبا السطر الأول يكون في نهايته فراغ ،أما السطر الثاني (المكرر) لا يكون في نهايته فراغ ، و بالتالي أنت المخطئ .. و السطر غير مكرر ..!!

Uniq :

يحذف الأسطر المكررة ، إذا كانت خلف بعضها(أي سطرين متتاليين)

مثلا:

```
$cat  roro | uniq
```

مثال:

$cat roro | sort | uniq

يحذف الأسطر المكررة ضمن الملف ، سواء في أي مكان (مشابه تماما لـــ sort –u)

نستخدم الخيار : uniq - d

يقوم بحذف الأسطر غير المكررة (أي يعرض الأسطر المكررة فقط).

نستفيد من ذلك في عدد من المجالات ،مثلا توليد أرقام بطاقات ائتمـــان يـــتم بشـــكل عشوائي ، و لكن يجب حذف الأرقام المكررة لضمان عدم حدوث مشاكل.و يمكننا ذلك من خلال تنفيذ الأمر (uniq -d) على ملف الأرقام ،لعرض الأسطر المكررة ثم نقـــوم بحذفها .

grep :

تقوم بالبحث عن كلمة في ملف(الدخل) و طباعة الأسطر الحاوية على هذه الكلمة ،وهي حساسة لحالة الأحرف .

مثلا :

- عرض معلومات المستخدم MySQL :

```
$cat  /etc/passwd | grep MySQL
```

الدخل الكلمة

- عرض معلومات المستخدمين الذين يستخدمون الـ shell : bash

```
$cat  /etc/passwd | grep bash
```

نستخدم مجموعة من الخيارات :

- جعل البحث غير حساس لحالة الأحرف -i
- البحث (عوديا) ضمن جميع الملفات المحتواة في مجلد -r
- طباعة أسماء الملفات(فقط) التي تتضمن أسطر تحوي الكلمة -l
- عدد مرات ورود الكلمة في كل ملف -c
- ترقم الأسطر الحاوية على الكلمة في الـoutput -n
- تعرض الأسطر التي لا تحوي الكلمة (عكس الفحص reverse) -v

أمثلة :

- عرض معلومات المستخدم mysql علما أنني لا أتذكر طريقة كتابة اسم المستخدم (أي حالة الأحرف):

```
$cat  /etc/passwd | grep –i   mysql
```

- معرفة جميع المستخدمين الذين لا يستخدمون الـــ shell : bash .

```
$cat  /etc/passwd | grep -v bash
```

- البحث عن كلمة Network ضمن الاعدادات :

```
$grep –r  Network   /etc/*
```

أي البحث عن الكلمة Network ضمن جميع الملفات المحتواة في المجلد etc

لاحظ أننا لو نفذنا التعليمة بالشكل :

```
$grep   Network   /etc
```

لن يظهر معنا أي نتيجة لأن المجلد etc لا يحوي ملف باسم Network .

الفرق بين find و grep :

find : البحث عن ملفات .

grep : البحث عن كلمة ضمن محتويات ملف.

ذكرنا في المحاضرة الماضية أنه يمكن تحديد عملية لتنفيذها على ناتج الـــ find ،و فيمايلي بعض الأمثلة:

- البحث عن الملف Network ضمن مجلد الإعدادات و عرض محتوياته على الشاشة:

```
$find  /etc   -type f   -name  Network  - exec  cat  {}  \;
```

/etc : مكان بدء البحث .

cat : الأمر المراد تطبيقه على نتائج البحث.

نقصد بقوسي المجموعة الفارغين أن دخل التعليمة هو خرج الـــ find .

\; :علامة الانتهاء.

مثال:

البحث عن جميع الملفات التي اسمها Network تحت etc/ و إجراء نسخ لها إلى البيت home directory بتعليمة واحدة.

```
$find /etc -type f -name Network - exec cp {} ~ \;
```

wc : word count

عدد الأحرف أو الأسطر أو الكلمات في ملف (الدخل).

نستخدم الخيارات التالية :

wc -l ⟶ line
-w ⟶ word
-c ⟶ character

أمثلة:

- عدد المستخدمين المعرفين ضمن النظام :

عدد المستخدمين هو عدد أسطر الملف passwd

```
$cat  /etc/passwd  | wc –l
```

- عدد المستخدمين الذين يستخدمون bash :

```
$cat  /etc/passwd  | grep bash | wc –l
```

- طول اسم المستخدم الرابع في النظام :

أولا نحصل عى السطر الرابع ثم نقص منه الحقل الأول (نكون قد حصلنا على اسم المستخدم الرابع)،و اخيرا نقوم بحساب عدد أحرف الاسم .

```
$cat  /etc/passwd  | head -4 | tail -1 | cut –f 1 –d : | wc –c
```

ملاحظة :

لاحظ أن الرقم الناتج سيكون هو طول اسم المستخدم الرابع + 1 و نفسر هذه الظاهرة بأن عملية cut تقوم بإضافة /n إلى الاسم حتى تظهر الأسماء على أسطر متتالية (على شكل عمود) و هذا هو المحرف الإضافي .

بعض الأمثلة عن مفهوم pipeline :

```
Ls-R /   |   tr ‘a-z’  ‘A-Z’ | less
```

حيث يتم عرض اسماء المجلدات المرتبة بعد التحويل في صفحات ..

```
Sort number |  tee sorted.txt | uniq –c
```

هنا يتم الاحتفاظ بنسخة عن الخرج باستخدام المعامل tee بملف وسيطي مرحلي غير مرئي باسم sorted.txt بعدها يتم تمرير الخرج للتعليمة التي تحذف التكرارات وتعدها

```
Cat< tom | xargs rm
```

باستخدام المعامل xargs يتم اخذ البارامتر لتعليمة ال rm من التعليمة السابقة يعني هنا أن الخرج أصبح جزء من التعليمة الممرر لها المعامل xargs