SED Stream EDitor

Sed (stream editor) je mocný dávkový textový editor. Jeho dokumentaci je možné najít např. na stránkách GNU. Pro začátečníky ale není manuálová stránka příliš čitelná, respektive používání příkazů sedu není přímočaré. Vhodným začátkem může být SED FAQ nebo samozřejmě kniha sed & awk. Dalším užitečným zdrojem informací jsou různé skripty, kterých se na webu povaluje celá řada.

Sed se, jak je asi zřejmé, pouští z příkazové řádky. Typicky se zařazuje do roury, ale není problém ho použít na zpracování textového souboru. Tedy například

```
cat file | sed ....
sed .... < file
sed .... file
```

Poslední dva příkazy zpracovávají text ze souboru. Ve všech případech je výstup směřován na standardní výstup, který se dá podle potřeby přesměrovat jinam. Tečky, které byly v příkazech, musí být ve skutečnosti nahrazeny příkazy sedu. Tyto příkazy můžeme buď zadat přímo na do příkazové řádky (pak zapisujeme '`sed příkazy`' nebo '`sed -e příkazy`'), nebo je můžeme mít v textovém souboru (pak sed zavoláme příkazem '`sed -f skript`').

Standardně funguje sed tak, že načte řádku ze vstupu (do tzv. **pattern space**), vykoná na ní příkazy (oddělují se středníkem) a modifikovanou řádku vypíše. Chování lze ale mnoha způsoby ovlivňovat. Pokud zavoláme sed s parametrem `-n`, vypíšou se pouze ty řádky, u kterých je to explicitně skriptem nařízeno. Příkazy se nemusí vykonat na každé řádce; můžeme tedy vliv příkazů omezit tzv. adresováním. V sekvenci příkazů si můžeme definovat návěští a pak provádět podmíněné či nepodmíněné skoky. Do paměti **pattern space** si můžeme načíst několik řádek a pracovat s nimi najednou. Navíc máme k dispozici i pomocný paměťový prostor **hold space**, do kterého můžeme ukládat mezivýsledky.

Základem pro práci se sedem jsou regulární výrazy. Z jejich možností si uvedeme některé: Znaky \ [ ] ^ . * $ / mají speciální význam; ostatní znaky jsou "běžné". Potřebujeme-li některý z těchto znaků použít ve významu "běžného znaku", musí být uzavřený v uvozovkách nebo musí následovat za znakem \. V následující tabulce x, y, z symbolizují "běžné" znaky.

x	znak "x"
xyz	řetězec "xyz"
[xyz]	znak "x", nebo "y", nebo "z"
[x-z]	všechny znaky od "x" až k "z" (ve smyslu ASCII)
[^xyz]	jakýkoliv znak vyjma "x", "y", "z"
.	jakýkoliv znak, až na novou řádku
^x	znak "x", pokud se nachází na začátku řádky
x$	znak "x", pokud se nachází na konci řádky
x*	libovolný počet znaků "x"
x\{m,n\}	m až n výskytů znaku "x"
\(... \)	vyznačují v regulárním výrazu podřetězec, na který se lze odvolávat sekvencemi \1 až \9. Na celý řetězec, který odpovídá regulárnímu výrazu, se dá odvolávat znakem &.

Typický příkaz typu "najdi a nahraď " v sedu má strukturu `adresa(nepovinné) příkaz (1 písmeno)/popis vstupu (reg. výraz)/popis výstupu/další parametry`

Nejjednodušší využití sedu je na operaci hledej/nahraď. Příklady:

sed 's/Ahoj/Nazdar/'	nahradí první výskyt řetězce Ahoj na řádce řetězcem Nazdar
sed 's/Ahoj/Nazdar/g'	nahradí všechny výskyty řetězce Ahoj na řádce řetězcem Nazdar
sed 's/[0-9][0-9]*/cislo/g'	nahradí všechna čísla řetězcem cislo
sed 's/[0-9][0-9]*/*** & ***/g'	obklopí všechna čísla třemi hvězdičkami, znak & symbolizuje celý nahrazovaný řetězec
sed 's/\(< img src="\)\ ([^"]*"\)/\1images\/\2/g'	v tagu < img > umístí před název souboru jméno adresáře, tedy například z řádky `< img src="obrazek.jpg" >` udělá řádku `< img src="images/obrazek.jpg" >`. Sekvence \1 symbolizuje podřetězec z nahrazovaného řetězce, který je obklopen první dvojicí \(a \), sekvence \2 podřetězec obklopený druhou dvojicí \(a \) atd.
s/\([0-9]\)\1/**/g	pokud se v souboru vyskytly dvě stejné číslice za sebou, nahradí je dvěma hvězdičkami
s/^/BAF!/	na začátek řádky vloží BAF!
s/^$/PRAZDNO/	prázdnou řádku nahradí slovem PRAZDNO

Pokud chceme hledání (či jinou činnost) omezit na některé řádky, použijeme adresování. U všech adresních způsobů platí, že připojením znaku "!" za adresu význam selekce znegujeme.

sed '1s/Ahoj/Nazdar/g'	provede záměnu jen na prvním řádku souboru
sed '1!s/Ahoj/Nazdar/g'	provede záměnu všude mimo prvního řádku souboru
sed '1,5s/Ahoj/Nazdar/g'	provede záměnu jen na prvním až pátém řádku souboru
sed '$s/Ahoj/Nazdar/g'	provede záměnu jen na posledním řádku souboru
sed '/Franto/s/Ahoj/Nazdar/g'	provede záměnu jen na řádcích, kde se vyskytuje slovo Franto
sed '/Franto/,/Karle/s/Ahoj/Nazdar/g'	provede záměnu jen od řádku, kde se poprvé vyskytuje slovo Franto, do řádku, kde se poprvé vyskytuje slovo Karle

Pokud chceme využívat skoky na návěští, musíme psát příkazy do příkazové řádky poněkud zvláštně, pomocí flagu -e. Příkazy pro návěští či skok totiž musí být uvedeny v sekvenci příkazů jako poslední; pokud ho potřebujeme někde uprostřed, musíme sekvenci rozdělit na dva kusy a zadat je odděleně. Příklad:

```
sed -n -e ':a' -e '1,10!{P;N;D;};N;ba'
```

Příkaz, který smaže posledních 10 řádek souboru, je poměrně zamotaný, proto si ho rozebereme. Jeho princip je následující: sed postupně načte prvních deset řádek ze vstupu, nebude s nimi nic dělat, jen si je bude skladovat v paměti. Jakmile přijde jedenáctá řádka, z paměti vytiskne první řádku a na konec paměti připojí právě načtenou jedenáctou řádku. Stejným způsobem pokračuje na řádce 12, 13 atd. V okamžiku načítání řádky *n* tedy bude vypisovat řádku *n-10*. Asi je zřejmé, že po načtení poslední řádky sed skončí svou činnost, a protože se dočasná paměť nevypíše, vznikne dojem, že sed smazal posledních deset řádek souboru (ty zůstaly ve vnitřní paměti sedu).

Flag -n říká, že sed nemá nic vypisovat vyjma toho, co je mu explicitně řečeno. Prvním flagem -e definujeme návěští :a na začátku sekvence

příkazů. Za něj nalepíme další příkazy.

Další tři příkazy ({P;N;D;}) nefungují na prvních deseti řádcích (to zařídí rozsah 1,10!), vysvětlíme si je tedy později. Proto na prvních deseti řádcích bude fungovat jen příkaz N, který do pattern space načte další řádek; řádky budou v pattern space odděleny znakem "\n". Tím, že po příkazu N skáčeme na začátek sekvence (nepodmínený skok na návěští :a příkazem ba), se do pattern space dostane prvních deset řádek. Od jedenácté řádky začnou fungovat i příkazy P;N;D; - P vypíše první řádek z pattern space, N načte další řádek (připojí ho na konec) a D první řádek smaže.

Pro podrobnější informace je samozřejmě nutné přečíst si manuálovou stránku sedu či jiné doporučené zdroje. Nejlépe se sed naučíte samozřejmě tak, že v něm připravujete skripty. Pokuste se proto o vyřešení následujících úloh; jsou to většinou jen hříčky bez většího uplatnění, zato si ale u nich procvičíte méně běžné schopnosti sedu.

Pod zadáním je vždy napsáno řešení, ale bílou barvou. Pro zviditelnění ho zvýrazněte, případně si tuto stránku stáhněte a barvu předefinujte. Jak jinak, než sedem! Mohli byste potřebovat tyto příkazy sedu:

- x - prohodí obsah pattern a hold space
- s - operace hledej/nahraď, syntaxe je s/hledej/nahraď/modifikátor, kde modifikátor může být prázdný (pak s nahrazuje pouze první výskyt na řádku) nebo g (pak se výměna udělá na všech výskytech na řádku)
- G - k pattern space připojí \n (konec řádky) a obsah hold space
- h - nahradí hold space obsahem pattern space
- p - vypíše obsah pattern space
- P - vypíše obsah pattern space od začátku do prvního znaku \n (konec řádky)
- = - vypíše číslo řádky
- N - na konec pattern space přidá \n (konec řádky) a do pattern space přidá další řádku ze vstupu
- b - nepodmíněný skok na návěští, např. ba skočí na návěští definované znaky :a
- t - podmíněný skok na návěští, provede se v případě, že poslední příkaz hledej/nahdaď uspěl a něco nahradil; syntakticky ta skočí na návěští definované znaky :a

Úlohy k něčemu dobré

1. Textový soubor budeme organizovat tak, že na řádce bude jméno, příjmení a řekněme plat. Vytvořte pomocí sedu (a příkazu sort či jiných elementárních příkazu shellu) skript, který uvedený textový soubor převede do podoby html stránky, kde položky budou uspořádány v tabulce a jména budou seřazena. Jako zesložitění můžete implementovat podbarvení každé sudé řádky tabulky.
2. Průzkumem fotobanky zjistíte, že V adresáři `puvodni` máme obrázky s původním názvem z fotoaparátu. V adresáři `vyber` máme výběr z těchto obrázků. Ty jsou ale kvůli pořadí prohlížení přejmenovány na 01, 02, 03 atd. Zajistěte jejich přejmenování na 01_, 02_ atd. Coby jednoznačný identifikátor můžete využít délku souboru. V našem konkrétním případě budou názvy `01__MG2383.jpg`, `02__MG1288.jpg`, `03__MG2198.jpg` atd.
3. Potřebujeme pozměnit automaticky generovanou fotogalerii. Zajistěte, aby šedý pruh na horním okraji všech stránek byl vysoký jen jednu řádku (tj. ne jako teď, kdy má výšku několika řádek). Dále ze všech stránek k jednotlivým obrázkům vymažte třistašedesátišestipoložkový seznam obrázků.

Úlohy k procvičení syntaxe

1. Najděte ve vstupním textu řádky s textem 'nazdar' a vložte za ně řádku se třemi hvězdičkami.
   Využijete příkazy s, x a G. Tip: v hold space si vytvořte řádek se třemi hvězdičkami a ten dle potřeby připojujte k načtené řádce v pattern space.

   ```
   sed '1{x;s/^/***/;x};/nazdar/G' file
   ```

   Protože jde o první náraz na kryptický jazyk sedu, podíváme se na řešení podrobněji. Tak za prvé: ne že by to nešlo jednodušeji. Například sekvence

   ```
   sed '/nazdar/s/^.*$/&\n***/' file
   ```

   udělá požadovanou činnost také. Jak? Na řádcích, kde je slovo nazdar, "najdeme" celou řádku od začátku do konce (část `^.*$`) a nahradíme ji za totéž (znak `&`) a připojíme konec řádku a hvězdičky. Kvůli dalším příkladům (e efektivitě) to ale uděláme jinak, pomocí pattern a hold space.

   Sledujte prosím řešení. Nejprve uděláme přípravu, vygenerujeme si pro pozdější využití onen řádek s hvězdičkami. Na prvním řádku (znak `1`) provedeme sekvenci příkazů mezi složenými závorkami. Nejprve si tento načtený řádek schováme do hold space, abychom si ho nepoškodili. Přesněji řečeno, prohodíme obsah pattern space (tam je řádek uložený) a hold space (tam na začátku není nic) příkazem `x`. Pak v pattern space vygenerujeme tři hvězdičky (příkaz `s` pracuje vždy v pattern space) a opět zaměníme hold a pattern space. Jakmile tedy sed dojde ve skriptu na ukončovací složenou závorku, bude mít v pattern space načtenou řádku (tu by tam měl tak jak tak), navíc bude ale hold space obsahovat řádku s hvězdičkami.

   Následující část skriptu říká, že za všechny řádky, které obsahují slovo nazdar (sekvence `/nazdar/`) se má připojit obsah hold space (příkaz `G`), což přesně potřebujeme.

2. Najděte ve vstupním textu řádky s textem 'nazdar'; v dalším řádku toto slovo podtrhněte znaky '-'.
   Využijete k tomu příkazy h, x, s, G. Tip: řádku, kde najdete "nazdar", si schovejte do hold space. Tam nahraďte slovo nazdar minusy, ostatní znaky mezerami a tuto modifikovanou řádku připojte k původní.

   ```
   sed '/nazdar/{h;s/nazdar/\n\n\n\n\n\n/g;s/[^\n]/ /g;s/\n/-/g;x;G}' file
   ```

3. Najděte ve vstupním textu řádky s textem 'nazdar'; v řádku pod i nad toto slovo vyznačte znaky '-'.
   K tomu budete potřebovat příkazy h, x, s, G a možná p. Tip: modifikujte předchozí úlohu, řádek s minusy potřebujete tisknout dvakrát.

   ```
   sed '/nazdar/{h;s/nazdar/\n\n\n\n\n\n/g;s/[^\n]/ /g;s/\n/-/g;p;x;G}' file
   ```

4. Očíslujte neprázdné řádky vstupního souboru, mezi číslem řádku a prvním znakem nechť jsou dvě mezery.
   K tomu využijete příkaz =, N, s. Tip: není třeba všechno dělat sedem na jeden průchod. V prvním průchodu vložte příkazem = před neprázdné řádky jejich čísla, v druhém průchodu (např. unixovou rourou) neprázdné řádky slepte k sobě a znak konce řádku nahraďte dvěma mezerami.

   ```
   sed '/./=' file | sed '/./N;s/\n/  /'
   ```

5. Očíslujte řádky vstupního souboru, mezi číslem řádku a prvním znakem nechť jsou dvě mezery a čísla řádků nechť jsou zarovnaná doprava na 5 znaků.
   Využijete příkazy =, s, N. Tip: v prvním průchodu vložte příkazem = před každou řádku její číslo, v druhém průchodu načtěte řádku (číslo), připojte na začátek mezery, některé smažte tak, aby zůstalo číslo zarovnané na 5 znaků doprava a tento výsledek připojte k následujícímu řádku (řádek původního vstupu).

   ```
   sed = file | sed 's/^/     /;s/^ *\(.....\)$/\1/;N;s/\n/  /'
   ```

6. Vycentrujte text na šířku sloupce 80 znaků.
   Budeme využívat skok příkazem t a příkaz s. Tento příklad je dost zamotaný. Vtip je v tom, že řádek, má-li méně než 80 znaků, můžeme z obou stran obklopit jednou mezerou; toto opakujeme tak dlouho, dokud se záměny daří (tj. řádek má méně než 80 znaků).

   ```
   sed -e ':a' -e 's/^.\{1,78\}$/ & /;ta' file | sed 's/ *$//'
   ```

7. Končí-li řádek znakem '\', připojte k němu řádek následující (znak '\' při tom zmizí).
   Využijeme k tomu opět podmíněný skok příkazem t a příkazy s, N. Jak na to: pokud řádek obsahuje na konci lomítko, připojíme následující řádek, smažeme sekvenci "zpětné lomítko" "konec řádku" a toto děláme do omrzení.

   ```
   sed -e ':a' -e '/\\$/N;s/\\\n//;ta' file
   ```

8. Vypište soubor. Pokud se v něm vyskytuje několik stejných řádek za sebou, vypište ji jen jednou.
   Využijeme k tomu příkazy N, P, D. Jde to vyřešit dvojím způsobem, buď (varianta 1) při tom vynecháme prázdné řádky, nebo (složitější varianta 2) je nevynecháme.

   V první variantě to můžeme udělat tak, že na každé vyjma poslední řádky načteme do pattern space dvě po sobě jdoucí řádky, pokud nejsou stejné, první z nich vytisknout a v každém případě první z nich smazat.

   ```
   sed '$!N; /^\(.*\)\n\1/!P; D' file
   ```

   V druhé variantě to uděláme podobně, ale dáme si pozor na to, aby v pattern space byl na řádce alespoň jeden znak.

   ```
   sed '$!N; s/^\n/ \n/;/^\(.*\)\n\1/!P; D' file
   ```

9. Odstraňte z html kódu všechny tagy. Předpokládejte, že znaky '<' a '>' jsou použity jen pro označení tagů.
   Opět budeme využívat skoky, tentokrát nepodmíněné příkazem b. Tip: nejprve odstraníme všechny tagy, které jsou pouze na jedné řádce. Pokud nám na řádce zbyl osamocený znak "<", musíme připojit další řádek a náhradu opakovat.

   ```
   sed -e ':a' -e 's/<[^>]*>//g;/
   ```

Domácí úkol:
K dipozici máte soubor s titulky k filmu ve formátu sub. Na každé řádce je ve složených závorkách číslo snímku, kde se má titulek zapnout, kde vypnout a text titulku, např. takto:

```
{0}{20}First subtitle
{30}{50}Second subtitle|New line is made this way.
{70}{100}Third.
{1010}{1033}Fourth etc.
```

Překladatel soubor přeložil takto:

```
{0}{20}First subtitle
První titulek.
{30}{50}Second subtitle|New line is made this way.
Druhý titulek|Nová řádka se dělá takto.
{70}{100}Third.
Třetí.
{1010}{1033}Fourth etc.
Čtvrtý atd.
```

Navrhněte pomocí sedu a případně jiných příkazů shellu tři postupy:

- Jak z překladového souboru vytvořit korektní soubor jen s českými titulky, tedy:

  ```
  {0}{20}První titulek.
  {30}{50}Druhý titulek|Nová řádka se dělá takto.
  {70}{100}Třetí.
  {1010}{1033}Čtvrtý atd.
  ```

- Jak z překladového souboru vytvořit korektní soubor jen s anglickými titulky, tedy:

  ```
  {0}{20}First subtitle
  {30}{50}Second subtitle|New line is made this way.
  {70}{100}Third.
  {1010}{1033}Fourth etc.
  ```

- Jak zkombinovat soubor s českými a soubor s anglickými titulky zpátky do překladové podoby, tedy:

  ```
  {0}{20}First subtitle
  První titulek.
  {30}{50}Second subtitle|New line is made this way.
  Druhý titulek|Nová řádka se dělá takto.
  {70}{100}Third.
  Třetí.
  {1010}{1033}Fourth etc.
  Čtvrtý atd.
  ```

Poslední změna: 02.11.2011