F4270 UNIX, počítačové sítě (jaro 2022): zápočtové příklady

A1 Napište příkaz, který vypíše všechny názvy souborů ve vašem domovském adresáři, které začínají tečkou `.` a obsahují alespoň 2 znaky.

A2 Napište příkaz, který vypíše poslední řádku z každého souboru v adresáři `/usr/include`, jehož název končí `.h`.
nebo

A3 Napište příkaz, kterým okopírujete všechny soubory z adresáře `/usr/include/`, jejichž jména nezačínají číslicí, ale číslici v názvu obsahují, do vašeho domovského adresáře.

A4 Vypište počet skupin v systému (řádků v `/etc/group`).

A5 Napište příkaz, kterým nastavíte datum a čas poslední modifikace souboru na `11.3.1993 15:04`.

A6 Vypište jméno druhého největšího souboru v adresáři `/usr/bin`.

A7 Napište příkaz, kterým všem souborům v podstromu adresáře `/tmp/adresar` přidáte právo ke čtení všem členům skupiny a současně právo ke čtení zruší ostatním (tj. těm, kdo nejsou vlastníci ani členové skupiny).

A8 Vytvořte soubor a přidělte mu práva tak, aby jej mohl vlastník spustit, číst i zapisovat do něj, členové skupiny jej mohli spustit a číst, ostatní jej mohou jen číst. V příkazu `chmod` použijte jednak symbolický zápis práv a jednak octalový zápis práv

---

B1 V souboru `calories.csv` nahraďte každý název potraviny (tj. první sloupec) jediným znakem `-`. Ostatní sloupce zůstanou beze změny.

B2 Vypište dvojice `uid:login` za každého uživatele v `/etc/passwd`.

B3 Určete počet různých států, které se vyskytují v souboru `ip-by-country.csv`.

B4 Na základě obsahu souborů `countrycodes_en.csv` a `kodyzemi_cz.csv` vypište názvy států, které se jmenují stejně v angličtině jako v češtině.

B5 V podstromu aktuálního adresáře najděte soubory, které mají nulovou velikost.

B6 V podstromu adresáře `/usr/include` najděte soubory, jejichž jméno začíná řetězcem `std` a nekončí příponou `.h`.

B7 Vypište názvy souborů, které se nacházejí v adresáři `/usr/bin` (jen přímo, nikoli v podadresáři, takže stačí `ls`) a o kterých příkaz file vypíše, že jde o shellový skript (výpis `file` obsahuje „POSIX shell script“ jako popis souboru, `file` vypisuje ve tvaru *soubor: popis*).

B8 Vypište všechny uživatele (jen loginy), kteří mají v `/etc/passwd` uvedenou primární skupinu s číslem 200.

C1 Pomocí sedu ve vstupním textovém souboru nahraďte výskyty znaků '`&`', '`<`', '`>`' pomocí jejich HTML entit ('`&`' pomocí '`&`', '`<`' pomocí '`<`' a '`>`' pomocí '`>`').

C2 Pomocí sedu ve vstupním souboru spojte vždy dvě následující řádky do jedné (tj. odstraňte odřádkování mezi nimi). Ve výsledku tedy bude spojena 1. s 2. řádkou, 3. se 4. řádkou, 5. s 6. řádkou, atd.

C3 Předpokládejte, že řádky vstupního souboru obsahují jen znaky '`(`' a '`)`' (tj. na každé řádce je řetězec vyhovující regulárnímu výrazu "`[()]*`"), vytvořte skript pro `sed`, který zkontroluje, zda je řetězec na řádce správně uzávorkovaný. Pokud ano, nahradí řádku řetězcem '`ano`', v opačném případě řetězcem '`ne`'.

C4 Napište příkazy `edu`, které ve vstupním souboru přesune řádky začínající znakem `#` na konec souboru.

C5 Uvažte vstupní soubor, který obsahuje text, kde odstavce jsou odděleny prázdným řádkem. Pomocí `edu` proveďte spojení každého odstavce do jedné řádky. (Poznámka: Ke spojení dvou řádek do jedné lze použít v `edu` příkaz '`j`'.)

C6 Napište skript, který text ze souboru `apollo` zpracuje tak, že na výstupu vygeneruje zdrojový kód v LaTeXu, který sází abecední seznam použitých slov, každé slovo se ve výsledku objeví právě jednou, velká písmena převeďte na malá. Stanovte počet použitých slov.

C7 Napište skript, který text ze souboru `apollo` zpracuje tak, že pro dvacet nejčastěji použitých slov vytvoří histogram četnosti vykreslený gnuplotem (použijte terminál `pngcairo`).

D1 Napište skript, který dělá totéž, co `cp`, ale s opačným pořadím parametrů (cíl je jako první, nezapomeňte, že obecně `cp` přijímá $n+1$ parametrů s tím, že je-li $n > 1$, pak poslední parametr `cp` musí být adresář.

D2 Napište skript, který očekává tři parametry, soubor, číslo `n` a znak `c`. Pokud třetí parametr chybí, za znak `c` je považován tabelátor (`c=tab`), pokud chybí i druhý parametr, uvažuje se navíc `n=1`. Skript potom provede cyklický posun doleva řádek podle položek oddělených znakem `c` s krokem `n`. Tj. například při volání `skript /etc/passwd 2`: by každou řádku tvaru
`login:*:uid:gid:jméno:domovský adresář:shell`
upravil na řádku tvaru
`uid:gid:jméno:domovský adresář:shell:login:*`
Upravený soubor vypisuje skript na standardní výstup, soubor zadaný v parametru neupravuje.

D3 Napište skript, který očekává dva parametry, oba obsahují cestu k adresářům, zdrojovému a cílovému, přičemž cílový dosud neexistuje. Skript zduplikuje adresářovou strukturu do cílového adresáře. Soubory v adresářích nekopíruje a ignoruje je, vytváří jen kopii adresářové struktury. Například volání `skript /etc /etc` by v domovském adresáři vytvořilo podadresář `etc`, pod ním by byl stejný podstrom adresářů jako je v `/etc` (bez kopií souborů v nich obsažených).

D4 Napište `awk` skript, který formátuje soubor po odstavcích do zadaného počtu sloupců (=znaků na řádek). Jednotlivé odstavce jsou od sebe odděleny volným řádkem, tj. několik řádek za sebou tvoří týž odstavec, pakliže mezi nimi není žádná prázdná řádka. Program vždy vezme odstavec a napíše ho na co nejmenší počet řádků tak, aby k přechodu na nový řádek nedocházelo uprostřed slova. Počet sloupců bude zadaný jako parametr předaný pomocí `-v`.

D5 Napište skript `rmexcept`, který očekává $n+1$ parametrů, kde první parametr je adresář a další parametry jsou řetězce popisující názvy souborů (může jít o shellové masky) v tomto adresáři. Skript v zadaném adresáři vymaže všechny soubory, jejichž názvy nevyhovují maskám zadaným v parametrech. Soubory, které vyhovují zadaným maskám a názvům, skript nechá beze změny, přičemž se jich ani netkne (tedy speciálně nebude je přesouvat mimo a pak zpět). Například volání `rmexcept . '*.jpg' '*.png'` by v aktuálním adresáři vymazalo všechny soubory, které nemají příponu `jpg` ani `png`.

D6 Vytvořte skript pro `sed`, který na řádce očekává číslo n zadané v desítkové soustavě. Na výstup vypíše `sed` čísla 0, 1, ..., n, každé na nové řádce.

D7 Napište skript, který ze souborů jejichž názvy jsou zadané v parametrech, vybere n nejdelších řádek, u každé řádky je uvedeno pořadí ve vstupním souboru (ve formátu `číslo řádky: řádka`), řádky jsou na výstupu uspořádány podle pořadí ve vstupním souboru (tj. podle toho čísla před dvojtečkou na výstupu). Parametr n je implicitně 5, nebo může být zadán jako první jeden až dva parametry pomocí `-n10`, nebo `-n 10` (to je pro příklad $n = 10$, tj. syntaxe je stejná jako třeba u `head`). Pokud chybí parametry s názvy souborů, čte se standardní vstup. Pokud jsou zadány alespoň dva soubory, pak je výpis nejdelších řádek z každého souboru uvozen názvem souboru.

D8 Napište skript, který bude ve vstupním textovém souboru provádět indentaci řádků podle hloubky zanoření daném závorkami. Parametry skriptu bude název znak indentace c, počet znaků na úroveň n a seznam souborů (pokud chybí soubory, čte se standardní vstup). Skript před každou řádku vloží $k * n$ znaků c, kde k je hloubka zanoření počátku řádku v závorkách, uvažují se kulaté, hranaté a složené závorky. Pokud jsou před začátkem textu na řádce již prázdné znaky, tak jsou nejprve odstraněny. Například vstupní soubor

```
a ( b
    c d [ e ] f [
  g h { j (
          k ) } l m
    n ] o ) p
q r
```

by volání skriptu s parametrem `c='.'` a počtem $n = 1$ upravilo do následující podoby:

```
a ( b
.c d [ e ] f [
..g h { j (
....k ) } l m
..n ] o ) p
q r
```

Můžete předpokládat, že vstup je správně uzávorkovaný (pokud není, je chování nedefinované). Dá se očekávat, že znakem může být mezera nebo tabelátor.