1.1. Začneme zlehka

Zadání:

Vypište jméno aktuálního pracovního adresáře.

Souborový systém v Unixu tvoří hierarchickou strukturu. Jednotlivá „patra“ tvoří adresáře (složky), které obsahují soubory a další podadresáře. Adresářová struktura má tak vlastně podobu *stromu* vyrůstajícího z tzv. *kořene*.

Chceme-li vyjádřit umístění nějakého souboru nebo adresáře, musíme zapsat tzv. *cestu*, která říká, přes jaké adresáře se dostaneme k cíli. Názvy jednotlivých adresářů se v cestě oddělují (obyčejným) lomítkem („/“).

Každý běžící program v systému má nějaký adresář nastaven jako svůj pracovní adresář. Podstatou úlohy je vypsat cestu k aktuálnímu pracovnímu adresáři shellu.

Rozbor:

Přesně pro tento účel slouží příkaz **`pwd`** (*print working directory*).

Řešení:

```
sinek:~> pwd
/home/forst
sinek:~>
```

Poznámky:

Po přihlášení do systému bude aktuálním adresářem váš *domovský* adresář – ten, který vám byl vytvořen při založení uživatelského účtu a ve kterém si můžete vytvářet soubory a podadresáře dle chuti. Změnit pracovní adresář můžete příkazem **`cd`** (*change directory*).

Pojem cesty budete možná v dnešní době znát z jiného prostředí, kam se vcelku intuitivně přenesl, a to jako cestu v internetových adresách stránek systému WWW:

```
http://www.yq.cz/trail-o/WTOC2008_results.pdf
```

I zde cesta (zjednodušeně řečeno) vyjadřuje umístění souboru s textem webové stránky.

Můj počítač se jmenuje „`sinek`“. Není to gramatická chyba, jméno má původ v anglickém slovese „think“. V současnosti bývá zvykem, že když vás unixový počítač vyzývá k zadání příkazu (neboli vypisuje tzv. *prompt*), představuje se svým jménem. Za jménem a dvojtečkou vidíte znak „`~`“ (vlnka), jenž znamená, že se právě nacházíme v mém domovském adresáři. Za vlnkou je ještě zobáček a mezera, které oddělují prompt od mnou napsaných příkazů, tzv. *příkazové řádky*. Přesný formát promptu je hodně závislý na nastavení prostředí. Pokud váš počítač vypisuje prompt v jiné podobě, nenechte se tím vůbec zmást.

1.2. Přidáme parametry

Zadání:

Vypište obsah adresáře `/etc`, včetně detailních informací o jednotlivých souborech.

Rozbor:

Pro tento úkol opět existuje správný příkaz (**`ls`**), ale tentokrát budeme muset přidat za název příkazu i nějaké parametry. Kdybychom zavolali příkaz bez nich, vypsal by pouze seznam jmen souborů a podadresářů, a to v aktuálním adresáři.

Přidáme tedy parametr „**`-l`**“ (*long*), který zařídí tzv. „dlouhý“ výpis. Každému souboru v něm přísluší jedna celá řádka s informacemi, jako jsou přístupová práva, jméno vlastníka, velikost, datum a čas poslední modifikace a jméno souboru.

A druhým parametrem příkazu sdělíme, že chceme vypisovat obsah adresáře `/etc`.

Řešení:

```
sinek:~> ls -l /etc
total 586
drwxr-xr-x  2 root  wheel       512 Oct  1 03:30 X11
... atd.
```

Poznámky:

U unixových příkazů se tradičně významově odlišují parametry, které začínají minusem (tzv. *přepínače* nebo optiony) – ty ovlivňují, *co* přesně příkaz dělá (v našem případě dlouhý výpis) – a ostatními parametry (tzv. *operandy*), které určují, *s čím* to dělá (v našem případě jméno adresáře). Všechny parametry se navzájem oddělují mezerami.

Popis parametrů a hlavně všech přepínačů člověk samozřejmě nenosí v hlavě. Návod (tzv. *manuálovou stránku*), jak s příkazem pracovat, zobrazíme příkazem **`man`**:

```
sinek:~> man ls
```

Pokud neznáme název potřebného příkazu, máme možnost název manuálové stránky vyhledat pomocí klíčového slova, pokud zadáme přepínač „**`-k`**“ (*keyword*):

```
sinek:~> man -k directory
```

Jednou ze základních jednotících myšlenek Unixu je zásada, že (skoro) všechno je soubor. Tento princip ulehčuje významně život tvůrcům programů, protože zjednodušuje programátorské rozhraní aplikací (API). Dokonce i na některé informace o běžícím programu se dá nahlížet jako na soubor. Pro nás je ale v této chvíli důležitý jiný důsledek – i adresář je soubor. Budeme tedy od této chvíle důsledně používat termín *obyčejný soubor* (text či data jako posloupnost bytů), pokud by hrozilo, že dojde k záměně s obecným pojmem „soubor“ (ve smyslu „obyčejný soubor i adresář“).

Nyní už si můžeme zpřesnit popis příkazu **ls** – vypisuje informaci o souborech, jejichž jména dostal jako parametry, a to tak, že v případě adresáře vypíše jeho obsah (seznam souborů a informace o nich), v případě obyčejného souboru vypíše jen jeho jméno (resp. informace o něm). Vyzkoušejte si třeba příkaz:

```
sinek:~> ls -l /etc/passwd
```

Kdybyste chtěli vidět pouze informace o adresáři `/etc` samotném a nikoliv jeho obsah, museli byste použít další přepínač příkazu **ls**, a to „**-d**“ (*directory*):

```
sinek:~> ls -l -d /etc
```

Většina unixových příkazů ovšem umožňuje přepínače *slučovat*, tj. napsat jich více za sebou s jediným znakem „**-**“. Předchozí ukázku by tedy bylo možné napsat jako:

```
sinek:~> ls -ld /etc
```

Stejně tak je obvyklé, že příkazy dokážou akceptovat více operandů. Pokud bychom chtěli vypisovat současně ještě obsah dalšího adresáře, například `/tmp`, můžeme napsat:

```
sinek:~> ls -l /etc /tmp
```

Všechny doposud uvedené cesty v parametrech, byly cesty *absolutní* – začínaly lomítkem a popisovaly cestu od kořene. Jinou možností, jak popsat umístění souboru, je *relativní* cesta od aktuálního adresáře. Poslední ukázku bychom mohli změnit např. na:

```
sinek:~> cd /
sinek:/> ls -l etc tmp
```

Všimněte si, že po změně adresáře se změnil obsah mého promptu.

Pokud se při psaní relativní cesty potřebujeme odkázat na *aktuální* adresář, můžeme použít soubor se zvláštním jménem „**.**“ (tečka). Volání **ls** bez parametrů by tedy bylo možné nahradit třeba voláním:

```
sinek:~> ls .
```

A v každém adresáři je ještě jedno zvláštní jméno, které představuje *rodičovský* nebo *nadřazený* adresář, a to „**..**“ (dvě tečky). Používá se pro pohyb „vzhůru“ po adresářové struktuře. Můj domovský adresář by bylo možné (zbytečně složitě) vypsat pomocí:

```
sinek:~> ls -l ../../home/forst
```

(neboli: „vyskoč o dvě patra výš a pak sestup do podadresářů `home` a `forst`“).

Pokud v parametru uvedené jméno (cesta) není jménem existujícího souboru, příkaz skončí chybou. Toho se dá například využít pro jednoduchý test existence souboru.

Názvy příkazů v Unixu bývají většinou „logické“ (jen je občas nutné si na tu logiku poněkud zvyknout). Název příkazu **ls** patří mezi výjimky potvrzující toto pravidlo. Zkratka pochází ze slov „list segments“ a je reliktem, jakousi zkamenělinou pamatující předchůdce Unixu, systém Multics. Tenhle název si prostě musíte zapamatovat.

1.3. Poprvé kombinujeme

Zadání:

Vypište počet uživatelů systému.

Seznam uživatelů se nachází v souboru **`/etc/passwd`**. Soubor je textový a každému uživateli přísluší jedna řádka (její obsah si probereme v poznámkách):

```
root:*:0:0:Super User:/root:/bin/bash
... atd.
forst:*:1004:1004:Libor Forst:/home/forst:/bin/bash
```

Rozbor:

Tento úkol už nás donutí přemýšlet v souvislostech. Počet uživatelů se rovná počtu řádek souboru `/etc/passwd`, takže by nám stačilo je jen spočítat.

Počítat řádky umí příkaz **`wc`** s přepínačem „**`-l`**“. A abychom k němu řádky souboru dostali, použijeme operátor přesměrování „**`<`**“ s operandem `/etc/passwd`. Shell potom otevře soubor (`/etc/passwd`) a předá ho na vstup příkazu **`wc`**.

Řešení:

```
sinek:~> wc -l < /etc/passwd
      27
```

Poznámky:

Pokud jste zvyklí na jiné operační systémy, možná vás překvapí, že seznam uživatelů je uložen v *textovém* a nikoliv v *binárním* souboru. Textový soubor se člení se na řádky obsahující pouze tisknutelné znaky a ukončené znakem konce řádky (*line feed*, LF). Takový soubor lze číst obyčejným editorem a zpracovávat (namísto jednoúčelových speciálních nástrojů) zcela obecnými programy (*utilitami*). A takových programů má UNIX velké množství. Je to důsledek historického vývoje a dle mého nejhlubšího přesvědčení je to důsledek dobrý.

Jednotlivé sloupce (nebo také pole) řádek v souboru `/etc/passwd` jsou odděleny dvojtečkami a obsahují pro každého uživatele následující atributy:

- jméno nebo též *přihlašovací jméno* (*login*);
- zakódované heslo (dnes už tam obvykle nebývá, ale pole samo v souboru kvůli kompatibilitě zůstalo a bývá vyplněno třeba hvězdičkou);
- číslo uživatele (*UID*);
- číslo (primární) skupiny uživatele (*GID*);
- *plné jméno* uživatele;
- domovský adresář uživatele;
- *login-shell* (název, resp. cestu k shellu, který se uživateli spouští při přihlášení).

Soubor může ještě obsahovat komentářové řádky (začínající znakem „#“). Ty se budeme muset časem naučit ze zpracování vyloučit. Prozatím je budeme ignorovat.

K tomu, aby programy v UNIXu mohly komunikovat se svým okolím, slouží vstupní a výstupní *proudy dat*. Každý program má implicitně otevřen tzv. *standardní výstup* a má ho přiřazen na obrazovku terminálu, na kterém běží. Díky tomu jsme ostatně mohli vidět výsledky všech doposud spouštěných příkazů. Stejně tak má každý program přiřazen i *standardní vstup*. Tím je implicitně klávesnice terminálu, kde program běží. My jsme toto přiřazení zrušili operátorem přesměrování. Kdybychom to neudělali, program **wc** by čekal na to, co naťukáme na klávesnici terminálu, a nakonec by nám to spočítal (tedy, spočítal by nám napsané řádky). Vyzkoušejte si to. Ale nejprve dočtěte odstavec! Až naťukáte celý text, budete totiž muset nějak vyjádřit, že už nic dalšího psát nehodláte. K tomu slouží kombinace kláves Ctrl+D, kterou zmáčknete po odřádkování poslední řádky.

Zastavme se ještě u pojmu *terminál*. Pochází rovněž z prehistorie. Kdysi označoval hardwarové zařízení, které mělo klávesnici a obrazovku (a ještě před tím třeba psací stroj...) a sloužilo pro komunikaci s počítačem. Dnes je jeho pozůstatkem softwarová komponenta, která uživateli poskytuje přesně totéž – klávesnici a obrazovku, ze které budete moci svůj UNIX ovládat.

Nabízela se ještě jedna možnost, jak soubor příkazu předat. Když napíšeme název souboru jako operand na příkazové řádce, nebude **wc** číst vstup, ale zadaný soubor. Tak se ostatně chová většina unixových utilit:

```
sinek:~> wc -l /etc/passwd
      27 /etc/passwd
```

Výsledek ale nesplňuje zadání, protože tam překáží jméno souboru.

Kdo se vydá studovat manuálovou stránku (příkazem „**man wc**“), aby našel přepínač, kterým potlačí výpis jména souboru, jde na věc zcela správně, systémově a prokázal velkou míru pochopení problematiky. Bohužel neuspěje. Příkaz takový přepínač nemá.

Opravdoví programátoři pravděpodobně začnou dumat nad tím, jak z výstupu jméno odfiltrovat nějakým dalším programem. To, pochopitelně, jde. Ale my to zatím neumíme a dokonce i poté, co to umět budeme, nebude to nejlepší řešení.

Nejjednodušší řešení, které jsme také nakonec použili, nám nabídl „selský rozum“. Když program nebude jméno souboru znát, neboť nečte soubor, ale svůj standardní vstup, nemůže žádné jméno vypsat.

Poněkud podezřelé jméno příkazu **wc** pochází z anglických slov *word count* (počítej slova), přestože se příkaz většinou používá pro počítání řádek či znaků. Proto jsme také museli použít přepínač „**-l**“ (*lines*), abychom dali najevo, že nás zajímají jenom řádky.