**1. (2 študenti)**

Vytvorte program, ktorý bude postupne vypisovať obsah súboru/súborov, zadaných na príkazovom riadku. Vypisovane riadky budú zobrazovane smerom nahor.

Popis príkazu:

**scrollfile [options] file(s)** - scrolls the content of file(s) on the screen

default size of scrolling window is 10 lines, default delay time is 0.5sec

Options:

**-l <lines>** changes number of scrolling window lines

**-d <number>** changes delay time

**-c** count lines

**-c** repeat listing

**-h** print this help

Pre analýzu argumentov použite príkaz **getopts**.

**2. (2 študenti)**

Vytvorte skript na hľadanie a manipuláciu s duplicitnými súbormi. Zvoľte vhodný spôsob detekcie duplicít (veľkosť súboru, md5,...). Pre každú skupinu duplicitných súborov umožníte odstrániť zvolene súbory. Aplikácia by mala vytvoriť jednoduché textové menu, kde by sa dalo určiť, ktoré súbory sa majú odstrániť a ktoré ponechať. Takisto musí byt možnosť kedykoľvek

ukončiť činnosť programu.

Parametrom je zoznam adresárov, v ktorých sa duplicity budú hľadať.

**3. (2 študenti)**

Vytvorte jednoduchý shell, ktorý bude prechádzanie súborovým systémom prezentovať

formou putovania hráča virtuálnym svetom. Jednotlivé adresáre predstavujú "dvere" miestnosti, do ktorých sa dá vstúpiť. Predmety (súbory) sa dajú ukladať do vaku a neskôr odkladať v iných

miestnostiach. Príkazy, ktoré shell nepozná spracuje bashom. Zoznam príkazov:

**chod <nazov\_miestnosti>** - vojde do danej miestnosti

**zober <predmet>** - zoberie predmet z danej miestnosti

**poloz <predmet>** - uloží predmet z vaku do aktuálnej miestnosti

**obsahvaku** - vypíše obsah vaku

**prezri**  - zobrazí zoznam predmetov v aktuálnej miestnosti

**cesty** - vypíše zoznam miestnosti, kam môžeme z aktuálnej miestnosti isť

**klonuj <predmet>** - vytvorí kópiu predmetu v danej miestnosti do vaku

**zahod <predmet>** - zahodí predmet z vaku

**znic <predmet>** - zničí predmet v danej miestnosti

**preskumaj <predmet>** - popise daný predmet

**koniec**  - ukončí shell

**4. (1 študent)**

Vytvorte v SHELLI kalkulačku. Kalkulačka umožní:

* sčítavanie, odčítavanie, násobenie a delenie pomocou operátorov: **+**, **-**, **x**, **/,**
* bude pracovať aj so zátvorkami () – operácie v zátvorke majú prednosť.

Operátory nie sú binárne, to znamená, že umožnia v rámci jedného príkazu vypočítať aj zložitejší výraz.

**5. (2 študenti)**

Vytvorte v SHELLI hru lodičky. Ide o klasickú hru pre dvoch hráčov, ktorí si na začiatku hry rozmiestnia flotilu v hernom poli a potom podľa súradníc jeden po druhom strieľajú. Vyhráva hráč, ktorý ako prvý zničí všetky protivníkove lode. Hra sa bude dať hrať z dvoch rôznych PCčiek a kedykoľvek dať ukončiť. Hracie pole bude mať rozmery 15x15 a bude vytvorene pomocou znakov. Každý pokus, kam hráč strieľal bude zaznačený. Susedné lode musia mať medzi sebou aspoň jedno voľné pole. Na začiatku bude flotila obsahovať:

1x **\* \* \* \*** 1x **\*** 2x **\*** 3x **\* \***

**\* \* \***

**\* \***

**\***

**6. (3 študenti)**

Vytvorte v SHELLI obdobu programu Norton Commander. Program umožní:

* prezerať adresárovú štruktúru spolu so súbormi a pohybovať sa v tejto štruktúre,
* kopírovať, premiestňovať, vytvárať a mazať súbory,
* vytvárať a mazať adresáre,
* prezerať a editovať obsah súborov.

Program sa bude dať kedykoľvek ukončiť, pričom zmeny sa udejú aj fyzicky v súborovom systéme.

**7. (2 až 3 študenti)**

Multi-user chat (história ako bonus alebo +1 študent v skupine).

**8. (2 študenti – každý ďalší študent znamená + 3 vlastne príkazy)**

Vlastný interpreter príkazov pričom je potrebne spraviť aspoň 4 vlastne implementácie z nasledujúcich príkazov:   
   a) **cat** vratane **-n**   
   b) **wc (-l -w -c)**   
   c) **du**   
   d) **head**   e) **tail**   f) **cut**   g) **finger**   h) **ps**   
   i) **vmstat**

**9. (3 študenti)**

Jednoduchý http www server, ktorý bude zobrazovať nasledujúce informácie o systéme:   
    a) hw info: CPU, pamäť,...   
    b) pripojene súborové systémy a ich voľna kapacita   
    c) prihlásení používatelia   
    d) momentálne vyťaženie   
    e) uptime   
    f) spustene procesy   
Server musí vedieť zanalyzovať HTTP request od prehliadača   a poslať naspäť prehliadaču platný HTML kód, ktorý zobrazí požadované údaje v prehľadnej forme.   
  
**10. (2 študenti)**

Http server ktorý umožní spúšťať bashovske (príkazy pod používateľským kontom príslušného študenta) cez prehliadač a zobrazí výsledok (bonus voliteľný interpreter príkazov, nielen bash). Server musí vedieť zanalyzovať HTTP request od prehliadača a poslať naspäť prehliadaču validný HTML kód, ktorý zobrazí vstupný formulár, umožňujúci odoslanie príkazu pre bash. Po odoslaní údajov z formulára, musí vedieť server spracovať zadaný príkaz a poslať prehliadaču HTML kód ktorý zobrazí výstup zadaného príkazu a vstupný formulár na zadanie ďalšieho prikazu.   
  
**11. (2 študenti)**

Http server, ktorý do internetového prehliadača vypíše prihlásených používateľov ktorí majú  povolene mesg, a umožní poslať vybraným z nich zadanú správu cez write. Server musí vedieť zanalyzovať HTTP request od prehliadača a na výstup vypíše formulár so zoznamom užívateľov, ktorým je možné poslať správu na konzolu cez príkaz write. Tento formulár tiež musí obsahovať vstupný komponent pre zadanie správy, ktorá sa ma poslať vybraným používateľom. Používatelia, ktorým sa ma poslať sprava sa budú vyberať pomocou checkboxov. Po odoslaní formulára server musí vedieť spracovať request od prehliadača a pomocou príkazu **write** poslať vybratým používateľom zadanú správu. Prehliadaču musí odoslať HTML kód, ktorý zobrazí vstupný formulár pre zadanie novej správy.

**12. (1 študent, graficky pre 2 študentov)**

**Slučka pre vývoj programu**

Syntax: ladenie <nazov zdrojoveho suboru>

* Ak zadaný súbor neexistuje, treba zistiť od používateľa, či chce ladiť skript alebo program v jazyku C. Ak existuje, tak sa typ súboru (skript alebo C-program) určí automaticky (ak to nie je program v jazyku C, považujte ho za skript). Spracovanie sa vetví podľa typu súboru.
* Ak je to skript - otvorí editorom súbor, používateľ zapíše príkazy a uzatvorí súbor. Následne scriptu nastaví príznak vykonateľnosti a spustí ho v novom shell-i s voľbou -x (aby bolo možné sledovať ako sa vykonávajú príkazy). Ak súbor prebehne bez chýb, ukončí ladenie, ináč sa vráti do editora (poskytnúť voľbu pokračovať v ladení, alebo ukončiť).
* Ak je to program v C jazyku, cyklus je podobný. Po uzatvorení editora sa spustí prekladač, ak ten vyhodí chyby, vráti používateľa do editora (s možnosťou aj vystúpiť zo slučky), ináč spustí program a po ukončení programu sa ukončí aj ladenie.

Každý z vyššie uvedených príkazov musí mať korektne ošetrené vstupy a pri zadaní neznámeho prepínača, alebo zlého počtu parametrov vypíše informáciu o správnej syntaxi použitia (prepínače a ich popis a argumenty a ich popis).

**13. (1 študent)**

Naprogramujte svoj príkaz find, ktorý prehľadá zadaný podstrom adresárovej štruktúry, nájde súbory, ktoré budú zadané a vykoná na nich uvedenú sériu príkazov. Zadanie vypracujte nasledovne: script musí obsahovať Usage, t.j. po nesprávnom zadaní parametrov výpise ich správne poradie a ukončí sa. Jednotlivé úseky programu okomentujte. Chybové stavy spracujte a vypíšte chybové hlásenie

**14. (1 študent)**

Vypracujte script, ktorý bude vypisovať z databázy popisov príkazov ( man -k):

* všetky príkazy shellu, ktoré sa tykajú zadaného tématu (napr. ktoré pracujú so súbormi),
* všetky systémové volania na zadane téma
* všetky interne príkazy shellu (man sh)
* obecne, bez udania manuálu, ktorý sa ma prehliadať