OS II.

Operációs rendszerek alapjai
II. gyakorlat

Shell script készítése

Shell script készítése

OS II.

- A shell script shell-parancsok sorozata
- A scripteket bármilyen szövegszerkesztővel (text editor) elő lehet állítani
  - nano
  - GNU Midnight Commander editora (mcedit)
  - Vi IMproved (vim)
  - gedit
  - kate
- Sokféle shell-ben írhatjuk a scriptjeinket (sh, bash, ksh, csh, ...)
  - #!/bin/sh
  - #!/bin/bash

Hello World!

OS II.

- echo
  - display a line of text
  - kiírja a megadott szöveget a terminálra
  - *echo Hello World!*
- chmod
  - change file mode bits
  - állományok hozzáférési jogai állíthatók vele, karakteres kapcsolókkal (r, w, x,...), vagy számjegyekkel (pl.: 755, 700)
  - *$ chmod +x script_file* - futtathatóvá teszi a script_file-t
- ./script_file
  - Alapból a $PATH globális változóban lévő útvonalakon keresi a parancsot (a futtatható fájlokat, programokat), ezért ha futtatni szeretnénk, adjuk meg, hogy az aktuális könyvtárba keresse, a "./" karaktereket elé írva
  - *$ ./script_file*
  - (de futtatható még például így is: *$ sh script_file* vagy *$ source script_file* sőt *$. script_file*)

Változók

OS II.

- A bash változói alapesetben szöveges változók
- A declare paranccsal meg lehet határozni:
  - integer típust
    - integer típusba ha szöveges adatot töltünk fel, akkor az nulla értéket ad vissza
    - *declare -i v* - integer típusú v nevű változó létrehozása
  - tömb típust
    - *declare -a v=(1 2 [4]=3)* - v nevű, 3 elemű tömb létrehozása, ahol az indexelés 0-tól kezdődik, de a 3. elem - jelen esetben a 3 - indexe a 4 lesz
    - *v[elemsorszam]="valami szoveg"* - értékadás az elemsorszam-adik elemnek
    - *echo ${v[elemsorszam]}* - az elemsorszam-adik elem értékének kiírása
    - *echo ${v[*]}* - a tömb összes elemének kiírása
    - *echo ${#v[*]}* - az értékkel feltöltött elemek száma
    - dinamikus tömb
  - de ekkor is minden karakteresen tárolódik
- A változót nem kell külön létrehozni, amikor értéket kap létrejön karakteres típusként

Változók

OS II.

- Idézőjelek:
  - ‘ ’ - egyszeres idézőjel
    - belsejében az összes speciális karakter jelentése ki van kapcsolva
    - *echo ‘$SHELL’* esetén a kimenet: $SHELL
  - “ “ - dupla idézőjel
    - a $, ', " és \ karakterek speciális jelentése megmarad a belsejében, megtörténik a változó helyettesítés
    - *echo “$SHELL”* esetén a kimenet: /bin/bash
  - ` ` - balra dőlő idézőjel (backtick)
    - a belsejében lévő utasítást kiértékeli (végrehajtja) a shell
    - mivel nem jól áttekinthető, ezért használata nem ajánlott, helyette inkább a $() használata javasolt (ami viszont nincs a Bourne Shell-ben, ott backtick kell)
- Backslash
  - \ - escape karakter
    - a speciális karakterek speciális jelentését megszünteti
    - *echo \$SHELL* esetén a kimenet: $SHELL

Adatfolyamok - standard input/output

OS II.

- echo - “Echo the STRING(s) to standard output”
  - a paraméterül kapott stringet kiírja a standard outputra
- read - read a line from the standard input
  - *$ read v* - a standard inputról beolvas egy sort a v változóba
  - ha argumentum nélkül indítjuk, vagyis nem határozunk meg változót, akkor alapértelmezetten a REPLY változóba kerül eltárolásra az adat
  - -p kapcsoló - “prompt_text”
    - hint írható ki a bekért adatra vonatkozóan
  - -s kapcsoló - secret/secure (do not echo input coming from a terminal)
    - beolvasás során a begépelt adat nem jelenik meg a terminálon
- cat - concatenate files and print on the standard output
  - ha argumentum nélkül indítjuk, a standard bemenetről (billentyűzet) fog olvasni és a standard kimenetre fog írni, azaz a standard bemenetet átmásolja a standard kimenetre

Adatfolyamok - standard error

OS II.

- Standard error az összes parancssori utasítás hibaüzenetének standard kimenete
- Alapesetben a hibaüzenet a kijelzőre íródik
- Hibaüzenetek:
  - errno - number of last error
    - EACCES - Permission denied
    - ENOENT - No such file or directory
    - ENOTDIR - Not a directory
    - ENOTEMPTY - Directory not empty
    - ...

Adatfolyamok átirányítása

OS II.

- A stream-ek számokban:
  - stdin - 0
  - stdout - 1
  - stderr - 2
- Minden stream-nek saját átirányítási parancsa van, mellyel a standard adatfolyamot fájlba lehet irányítani, ami ha korábban nem létezett automatikusan létrejön
- Átirányítási parancsok:
  - felülírja a már létező fájl tartalmát:
    - > - standard output
    - < - standard input
    - 2> - standard error
  - hozzáfűzi a már létező fájl tartalmához:
    - >> - standard output
    - << - standard input
    - 2>> - standard error

Adatfolyamok szerkesztése

OS II.

- Pipe
  - egy program standard kimenetét egy másik program bemenetére küldjük csővezetéken keresztül
  - az adat továbbítását egy FIFO oldja meg, amit váltakozva az első program tölt, a második kiolvas, és ez addig folytatódik, amíg az első programnak van mit tölteni (ez a cső szerű működés, innen a pipe...)
  - az első program kimenete nem jelenik meg a terminálon, csak a második program kimenete
  - | - a pipe jele
    - ls | head -3
      - ls - kilistázza a könyvtár tartalmát
      - head -3 - a fájl (lista) első 3 elemét kilistázza
      - vagyis csak a mappa első 3 elemét listázzuk ki
  - A pipe-ok többszörözhetőek, láncszerűen adhatják tovább az adatokat egymásnak
    - ls | head -3 | wc -L
      - a mappa első 3 eleme közül a leghosszabb nevű fájl nevének hosszát írja ki

Adatfolyamok szerkesztése

OS II.

- grep
  - global regular expression print
  - egy adott minta előfordulásait keresi a paraméterként megadott állományban
  - a minta egy reguláris kifejezés
  - ls | grep D
    - csak azokat a fájlokat listázza ki, melyek tartalmaznak D-t (a kis d betűvel nem foglalkozik - case-sensitive)
    - -i kapcsoló - kikapcsolja a case-sensitive működést
    - -r kapcsoló - a keresés rekurzívvá tehető
  - ls | grep “\.txt”
    - txt kiterjesztésű fájlok listázása
  - ls | grep [A-Z]
    - nagy betűt tartalmazó fájlokat listázza
- fgrep
  - fixed grep
  - a minta csak karakterlánc lehet, reguláris kifejezés nem (azaz pl. az fgrep [A-Z] nem működik)

Szöveges adatok feldolgozása

OS II.

- sort
  - sort lines of text files
  - a rendezés alapértelmezett szabályai:
    - a számok előbb vannak mint a betűk
    - a betűket ABC sorrendben rendezi
    - azonos betűk esetén a kisbetű előbb van mint a nagybetű
  - kapcsolókkal ezek a szabályok módosíthatóak:
    - -r, --reverse - megfordítja a sorrendet
    - -R, --random-sort - random rendezi az elemeket
  - sort data.txt > output.txt - data.txt fájl sorait rendezi és az output.txt fájlba írja az eredményt
    - sort -o output.txt data.txt - ez is ugyanazt csinálja…
  - sort -c data.txt - ellenőrzi, hogy rendezve van-e már az adat
    - elkezdi vizsgálni az elejétől és az első hibával visszatér (ha rendezve van már, akkor nincs hiba, nem tér vissza semmivel sem)

Szöveges adatok feldolgozása

OS II.

- uniq
  - report or omit repeated lines
  - kiszűri az egymás utáni egyező sorokat az inputban
  - ha az input és output nincs meghatározva akkor a standard inputról olvas és a standard outputra ír
  - alapesetben az egyező sorokat az első kivételével elhagyja
  - *uniq input.txt output.txt* - input fájlban kiszűri az egyező sorokat és a redukált tartalmat az output.txt fájlba írja
- cut
  - remove sections from each line of files
  - -c - karaktereket vág ki, melyek megadhatók intervallummal vagy karakterek listájával
  - -f - minden sor adott mezőjét vágja ki, a mezők között a tab az alapértelmezett elválasztó karakter
  - *cut -c 2-3 input.txt* - az input fájl minden sorának 2. és 3. karakterét írja ki

Szöveges adatok feldolgozása

OS II.

- paste
  - merge lines of files
  - *paste input1.txt input2.txt*
    - input1 n. sorához hozzáfűzi az input2.txt n. sorát
- join
  - join lines of two files on a common field
  - *join input1.txt input2.txt*
    - közös field alapján fűzi össze a két inputot, a fieldeket whitespace választja el egymástól alapértelmezetten
- tr
  - translate or delete characters
  - *tr “a” “A” < input.txt*
    - az input.txt fájlban kicseréli a kis a-t nagy A-ra
- aspell
  - interactive spell checker
  - *aspell -c input.txt*
    - interaktívan ellenőrizhetjük az input.txt szavainak helyességét

Szövegek összehasonlítása

OS II.

- comm
  - compare two sorted files line by line
  - *comm list1.txt list2.txt*
    - 3 oszlopban listázza ki az eredményt
      - első oszlop: azok amik csak a list1-ben szerepeltek
      - második oszlop: azok amik csak a list2-ben szerepeltek
      - harmadik oszlop: azok amik a list1-ben és list2-ben is szerepeltek
- diff
  - compare files line by line
  - *diff list1.txt list2.txt*
    - kiírja azokat amik különböznek a két list-ben, illetve jelzi, hogy milyen változtatás szükséges, hogy a két list megegyezzen:
      - változtatásra példa kimenet: "2,4c2,4": 2-től 4. sorig meg kell változtatni az első fájl tartalmát, a második fájl 2-től 4. sorára.
      - vagyis baloldalt első fájl sorai, jobb oldalt második fájl sorai, középen utasítás
      - utasítások: a - add, c - change, d - delete
      - < - az első fájlból származó sor
      - > - a második fájlból származó sor

Szövegek összehasonlítása

OS II.

- patch
  - apply a diff file to an original
  - a diff kimenetét felhasználva azonossá teszi a diff két különböző bemenetét
    - *diff -u file1.txt file2.txt >file.patch*
    - *patch < file.patch*
  - -b kapcsoló: csinál az eredeti fájlokról egy backupot, melynek neve az eredeti név .orig végződéssel
  - --dry-run kapcsoló: ellenőrizhetjük a patch működését anélkül, hogy valóban végrehajtanánk a módosítást a fájlokon
  - -R kapcsoló: a már végrehajtott patch parancs visszacsinálása
    - *$ patch -R < file.patch*

Feladatok - G02F01

OS II.

- Az eddigiek alapján a *vezeteknev.txt* illetve a *keresztnev.txt* fájlokban szereplő vezeték és keresztnevekből hozzunk létre random módon teljes neveket!
- A létrehozott random neveket rendezzük ABC szerint sorrendbe!
- Ellenőrizzük, hogy a generálás során minden vezetéknév fel lett-e használva a diff vagy a comm parancsok segítségével!

Feladatok - G02F02

OS II.

- A palacsintasütéshez az alapanyagokat a palacsinta.txt tartalmazza, míg az otthon lévő ételeket a konyha.txt
- Készítsünk bevásárlólistát az eddigiek alapján, ami tartalmazza azokat az élelmiszereket, melyek a palacsintához kellenek de nincsenek a konyhában!
- Feltételezve, hogy mindent sikerült megvásárolni ami hiányzott, listázzuk ki a vásárlás utáni, de még a sütés előtti konyha tartalmát!
- A sütés során minden ami a palacsinta alapanyaga volt elfogyott, listázzuk ki a sütés utáni konyha tartalmát!

Feladatok - G02F03

OS II.

- Az abc.txt fájlban hibásan szerepel az angol abc egy része, a helyes megoldást az abc_res.txt fájl tartalmazza
- Írjuk ki egymás mellé a két fájl tartalmát!
- A diff és patch parancsok és az abc_res.txt fájl segítségével javítsuk ki az abc.txt fájlt és a javítottat írjuk is ki!
- Vonjuk vissza a javítást!