OS III.

Operációs rendszerek alapjai III. gyakorlat

Vezérlési szerkezetek

test

- Check file types and compare values
- A kifejezésnek megfelelően igaz (0) vagy hamis (1) értékkel tér vissza
  - fájlok vizsgálata:
    - -d : *test -d Documents* - a Documents mappa-e vagy sem
    - -e : *test -e Documents* - a Documents létezik-e
  - numerikus értékek vizsgálata:
    - -eq: *test 5 -eq 5* - egyenlőség vizsgálata
    - -gt, -ge: *test 2 -gt 1* - 2 nagyobb mint 1, illetve -ge esetén nagyobb egyenlő
    - -lt, -le: *test 2 -lt 3* - 2 kisebb mint 3, illetve -le esetén kisebb egyenlő
  - szöveges kifejezések vizsgálata:
    - *test “alma” = “alma”* - ugyanaz a két string, tehát egyenlőek
- A test visszatérési értékét a “$?” változó tartalmazza
  - *echo $?* - kiírja a kifejezés értékét (0 vagy 1)
- A test parancs helyettesíthető a “[ ]” -lel
  - *[“madár” = “ász”]; echo $?* - a madár és az ász stringek nem azonosak, 1 a visszatérési érték
    - a “[” után és a “]” előtt, illetve az “=” előtt és után szóköz van
- A bash esetén modernebb verzió a “[[ ]]” és integerekre “(( ))”
  - Új funkció a bash-ben: [[string1 =~ regex]] - a stringek vizsgálata reguláris kifejezések használatával

test

- A kifejezések
  - negálhatók
    - test ! 3 -eq 3 - A 3=3 kifejezés igaz értékét negálja
      - a “!“ előtt és után szóköz van
    - [! 3 -eq 3]
    - ((! 3 == 3))
      - -eq helyett a “==” szolgál az összehasonlításra, illetve a <, >, <=, >= is működik
  - összekapcsolhatók
    - test és “[ ]” esetén:
      - -a : and
        - test 3 -eq 3 -a 5 -gt 4
      - -o: or
        - test “alma ” = “alma” -o “alma” = “alma”
    - “[[ ]]” és “(( ))” esetén:
      - && : and
        - [[3 = 3 && “ez” = “ez”]]
      - || : or
        - [[3 = 3 || “ez” = “az”]]

Feltételes vezérlési szerkezet - if

OS III.

if <condition>; then

<commands>

elif <condition>; then

<commands>

else

<commands>

fi

Feltételes vezérlési szerkezet - case

OS III.

```
case  $variable-name  in
        pattern1)
          command1
          ;;
        pattern2)
          command2
          ;;
        patternN)
          commandN
          ;;
        *)
           Default condition to be executed
    esac
```

Ciklusok - while

OS III.

- A while utáni feltétel vizsgálatra kerül, amíg igaz, a do és a done közötti utasítások végrehajtódnak.

```
while [ condition ]
        do
            command
        done
```

Ciklusok - until

OS III.

- Az until addig folytatja a ciklust, míg a feltétel hamis és amikor igazzá válik, akkor lép ki a ciklusból.

```
until [ condition ]
        do
            command
        done
```

Ciklusok - for

OS III.

- A for ciklus a ciklusnak átadott értékeken egyenként végig megy, majd kilép a ciklusból.

```
for variable in [ condition ]
        do
            command
        done
```

Ciklusok - for

OS III.

- A for ciklus a ciklusnak átadott értékeken egyenként végig megy, majd kilép a ciklusból.

```
for i in 1 2 3
        do
            echo $i
        done
```

Kimenet:

```
1
2
3
```

Ciklusok - for

- Újabb verziójú (v4.0+) bash (verzió lekérdezése: *echo ${BASH_VERSION}*) esetén a for listája megadható az intervallum két szélének értékével és a lépésközzel (*for i in {int_kezdete..int_vége..lépésköz}*)

```
for i in {0..50..20}
        do
            echo $i
        done
```

Kimenet:

```
0
20
40
```

Ciklusok - for

- Létezik C szerű for ciklus is.

```
for ((i=0; i<3; i++))
        {
            echo $i
        }
```

Kimenet:

```
0
1
2
```

Ciklusok - break és continue

OS III.

- break
  - kilép a ciklusból
  - *break n* - egymásba ágyazott ciklusok esetén az *n* paraméterrel megadhatjuk, hogy hány ciklusból lépjen ki
- continue
  - az adott iterációt megszünteti és átlép a következőbe
  - *continue n* - ugyanúgy használható az n paraméter, mint a break esetében

Feladatok - G03F01

OS III.

- Készítsünk egy faktoriálist számoló programot!
  - A program szólítsa a felhasználót a user nevén és írja ki, hogy a felhasználótól vár egy számot
  - A program 0-tól 20-ig terjedő egész bemenetre írja ki a szám faktoriálisát
  - A negatív bemenetre írja ki, hogy nem értelmezett a faktoriális
  - 20-nál nagyobb egészre írja ki, hogy túl nagy és nem tudja kiszámolni
  - A program addig kérjen új bemenetet minden kiírt kimenet után, míg a felhasználó exit utasítást nem ad

Feladatok - G03F02

OS III.

- Készítsünk ügyfélszolgálathoz interaktív automata menürendszert!
  - A program olvassa be fájlból a felhasználónevet (id.txt), a jelszót (pw.txt) és a számlán található összeget (money.txt)
  - Kérje meg a felhasználót, hogy azonosítsa magát a felhasználónév és jelszó (ne legyen látható mikor beírja a felhasználó) párossal
    - addig kérje újra ezeket, míg jót nem ad meg
  - Sikeres belépés után az alábbi menük közül választhat a felhasználó (amit írjunk is ki):
    - SIM-kártya letiltása
      - kérjen megerősítést, i - igen biztosan, n - nem, mégsem akarja letiltani
    - számlainformáció
      - írja ki a money.txt fájlból beolvasott összeget
    - panasz bejelentése
      - kérjen be egy egysoros szöveget és mentse ki a panasz.txt fájlba
    - kilépés
      - a program leáll
  - A végrehajtott menüpont után újból választhat közülük a felhasználó, amíg ki nem lép

Feladatok - G03F03

- Számoljuk ki 1-től N-ig a számok összegét!
  - Írjunk egy scriptet ami bekér egy pozitív egész N-t
    - Számolja ki 1-től N-ig a számok összegét ciklus nélkül
    - Írja ki az eredményt és a számolás futásidejét milliszekundum pontossággal
    - Számolja ki ciklussal is és szintén írja ki az eredményt és a futásidőt

Hint:

N*(N+1)/2 - 1-től N-ig a számok összege

date +%s%N - az idő nanoszekundum pontossággal