7. Gyakorlat

Ciklusok:

- Gyakran előfordul, hogy a programunkban egy kódrészlet végrehajtását többször meg akarjuk ismételni, ilyenkor célszerű ciklusokat alkalmazni.
- A bash-ben három ciklus típust különböztethetünk meg, a for, a while, és az until ciklusokat.
- Általános alakjuk:

```
ciklusfej
do
     utasítások;
done
```

A for ciklus:

- Általános alakja:

```
for ciklusváltozó in lista
do
     utasítások;
done
```

- A ciklusváltozó minden iterációban felveszi értékül a megadott lista következő elemét.
- A lista megadása történhet:
  - Egyszerű felsorolással. (pl.: `for i in 1 „2 3” Föld Hold`)
  - A seq paranccsal. (pl.: `for i in $(seq 1 10)`)
  - A {n..m} kifejezéssel. (pl.: `for i in {1..10}`)
  - Parancs(lista) végrehajtásával. (pl.: `for i in \`cat vmi.txt\``)
- Lehetőségünk van a C és egyéb programozási nyelvekből ismert for ciklus megadására is, a következőképpen:

```
for ((i=1; i<=10; i++)); do utasítások; done
```

- Példa (fájl sorainak beolvasása):

```
IFS=$'\n'
for sor in $(cat /etc/passwd); do
     echo $sor
done
```

A while ciklus:

- Általános alakja:

```
while kifejezés
do
     utasítások;
done
```

- Elől tesztelő ciklus, kiértékeli a megadott kifejezést, majd, ha igaz, végrehajtja a ciklusmagot.
- Amíg a megadott kifejezés igaz, addig újra és újra végrehajtja a ciklusmagban megadott utasításokat.
- A kifejezés megadása történhet:
  - A test paranccsal. (pl.: `while test –n „hello”`)
  - A […] kifejezéssel. (pl.: `while [ –n „hello” ]`)
  - Parancs(lista) végrehajtásával. (pl.: `while echo „hello”`)

- Példa (fájl sorainak beolvasása):

```
while IFS=$'\n' read -r sor || [[ -n $sor ]]
do
     echo $sor
done < „vmi.txt”
```

Az until ciklus:

- Általános alakja:

```
until kifejezés
do
     utasítások;
done
```

- Hátul tesztelő ciklus, végrehajtja a ciklusmagot, majd a megadott kifejezést kiértékelve dönt a folytatásról.
- Amíg a megadott kifejezés hamis, addig újra és újra végrehajtja a ciklusmagban megadott utasításokat.
- A kifejezés megadása a while ciklusnál leírtakkal megegyező módon történhet.

A true és false parancsok:

- Az igaz és hamis érték reprezentálására a shell scriptekben a true és false parancsok szolgálnak.
- Felhasználhatók elágazások és ciklusok logikai kifejezést tartalmazó részében.
- Példa:

```
while true; do echo „igaz”; done
until false; do echo „hamis”; done
if true; then echo „igaz”; done
```

A shift utasítás:

- Egy számot vár paraméterként (opcionális), ezt nevezzük N-nek.
- A pozicionális paramétereket „eltolja” N-nel.
- A pozicionális paraméterekre shift után ${N+1}-től $#-ig helyett $1-${N+1} módon hivatkozhatunk.
- Példa:

```
while [ $# -gt 0 ]; do
     echo „$1 $#”
     shift
done
```

A break és continue vezérlésátadó utasítások:

- A break utasítás megszakítja a ciklusmag végrehajtását, majd kilép a ciklusból.
- A continue utasítás megszakítja a ciklusmag végrehajtását, majd folytatja a következő iterációval.
- Példa break-re (az első páros számig írjuk ki a paraméterek értékét):

```
for i in $@; do
     if [ $(($i % 2)) -eq 0 ]; then break; fi
     echo $i
done
```

- Példa continue-ra (csak a páros számokat írjuk ki):

```
for i in $@; do
     if [ $(($i % 2)) -ne 0 ]; then continue; fi
     echo $i
done
```

Feladat 1: az eddig tanultakat felhasználva írjunk scriptet, amely

- paraméterként vár egy számot (default 100),
- generál egy random számot 1 és a megadott szám között (`$RANDOM`),
- beolvas egy számot, majd megmondja, hogy a beolvasott szám kisebb, nagyobb, vagy egyenlő a generált számmal,
- ha egyenlő, a program leáll,
- ha nem egyenlő, folytatódik a végrehajtás a 3. ponttól.

Feladat 2: az eddig tanultakat felhasználva írjunk scriptet, amely

- paraméterként egy fájl nevét várja,
- ellenőrzi, hogy a megadott fájl létezik-e, szöveges fájl-e, olvasható-e,
- ha valamelyik feltétel nem teljesül, a program leáll,
- egyébként kiírja, hogy a fájl egyes sorai hány karakterből állnak.

Feladat 3: az eddig tanultakat felhasználva írjunk scriptet, amely

- paraméterként vár egy számot (default 1000),
- kiírja az összes tökéletes számot 1 és a megadott szám között.