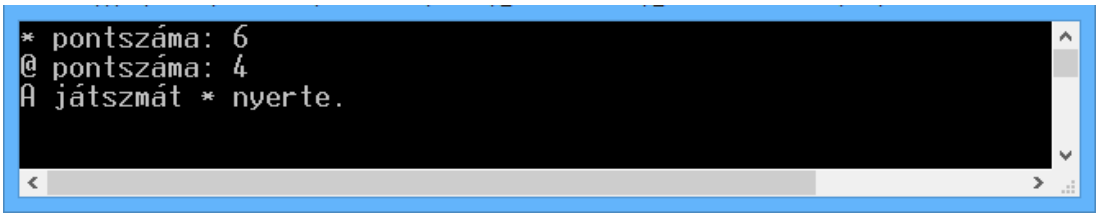
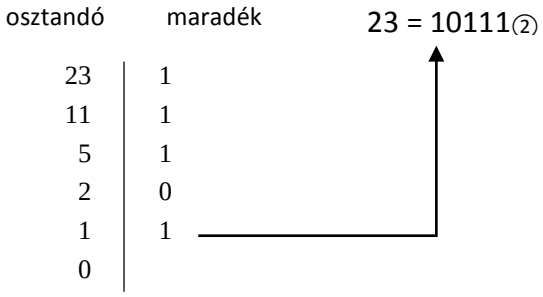
While ciklus, do-while ciklus

1. Egy változó értékét növeljük véletlenszerűen 1-gyel, 2-vel vagy 3-mal, amíg a változó értéke kisebb, mint 50. Minden lépésben írjuk ki a változó értékét, a kezdőértéke 0 legyen.
   1.1. Az előbbi példát oldjuk meg úgy, hogy a változó értékének növekedését egy * karakter mozgása mutassa a képernyőn.
   1.2. Mozgassunk a * karakter alatt 2 sorral egy @ karaktert is hasonló módon, és a végén írjuk ki, hogy ki „győzött”, azaz melyik érte el hamarabb az 50-et!
   1.3. Csináljunk az előbbi versenyből 10 futamot úgy, hogy minden futam után várjon a program az Enter leütésére, és számoljuk, hogy hányszor győz a * és hányszor a @.
   1.4. A végén írjuk ki, hogy a * vagy a @ nyerte a 10 futamos játszmát!
   1.5. Fejlesztési lehetőség: A játszma elején fogadhasson a felhasználó a *-ra vagy a @-ra, és a játszma végén azt is írja ki a program, hogy megnyerte, vagy elvesztette a fogadást!



2. **Eldöntés, keresés, kiválasztás**
   Egy internetes játékoldal 100-as ranglistáján a felhasználók életkorát vizsgáljuk. Készíts programot, amely a korok nevű tömböt 8…80 közötti véletlen-számokkal tölti fel, a számokat kiírja a képernyőre sorfolytonosan, szóközzel elválasztva, majd a képernyőre írja a válaszokat a következő kérdésekre?
   2.1. Van-e köztük 10 éves? Ha van, hányadik a ranglistán?
   2.2. Van-e köztük kiskorú? (14 év alatti)
   2.3. Van köztük fiatalkorú? (14. évét már betöltötte, de a 18.-at még nem)
   2.4. Mindenki elmúlt 12 éves?
   2.5. Mennyi a nagykorúak átlagéletkora?
3. A felhasználótól beolvasott számot a program váltsa át kettes számrendszerbe az alábbi ismert algoritmus szerint!



4. **Adatbevitel ellenőrzéssel.** Az alábbi feladatokban különböző számokat vagy szöveget kérünk be a felhasználótól. A program minden esetben ellenőrizze, hogy a megadott szám/szöveg megfelel-e a feltételeknek, és ha nem, kérje újra mindaddig, amíg helyes számot/szöveget ad meg a felhasználó.

   Minta:

```
Adj meg egy pozitív számot!
-1

A -1 nem pozitív szám.
Adj meg egy pozitív számot!
0

A 0 nem pozitív szám.
Adj meg egy pozitív számot!
3

Köszönöm. A program kilép.
```

   4.1. Kérjünk be egy pozitív számot!

   4.2. Kérjünk be egy 10 és 50 közé eső számot!

   4.3. Kérjünk be egy háromjegyű pozitív számot!

   4.4. Kérjünk be egy páros számot!

   4.5. Kérjünk be egy 3-mal és 5-tel is osztható számot!

   4.6. Kérjünk be egy 5-tel vagy 7-tel osztható számot!

   4.7. Kérjünk be egy „b” betűvel kezdődő szót!

   4.8. Kérjünk be egy „ny” betűvel kezdődő szót!

   4.9. Kérjünk be egy „ász”-ra végződő szót!

5. Készítsünk játékprogramot, amely gondol egy számot 1 és 50 között. A felhasználó addig találgathat, amíg nem találja el a keresett számot. A számítógép minden rossz tipp után írja ki, hogy a gondolt szám nagyobb vagy kisebb.

6. Készítsünk a mobiltelefonok PIN kódjának bekérését szimuláló programot! A felhasználó kapjon 3 próbálkozási lehetőséget, ha ez alatt nem találja el, jelenjen meg a „Hibás PIN kód, csak a 112 hívható!” üzenet!

7. Kérjünk be két egész számot billentyűzetről a 10...90 intervallumból. Amennyiben a beírt számok ezen kívül eső egész számok lennének, úgy addig ismételjük a bekéréseket, amíg megfelelő értékeket nem kapunk. A két számot fogjuk fel mint életkorok, egy apa és a fia életkorait. Adjuk meg, hány éves volt az apa, amikor a fia megszületett.(nem tudni melyik életkort adják meg előbb, a fiút vagy az apáét). Amennyiben az apa fiatalabb volt ekkor mint 18, vagy idősebb mint 50, akkor a program írja ki, hogy ’bár ez nehezen hihető’.

8. Kérjünk be egy egész számot (’a’) billentyűzetről. Ezen értéknek csakis pozitív számot fogadhatunk el (ha nem – ismételjük meg a bekérést, amíg megfelelő értéket nem kapunk). Majd kérjünk be egy ’b’ számot is billentyűzetről. Ezen ’b’ érték legalább 10-el nagyobb kell legyen mint az ’a’ értéke (ha nem – ismétlés amíg megfelelő nem lesz). Írassuk ki az [a..b] intervallumbeli páros számokat a képernyőre.

9. Legyen X=1 induló értékünk. Egy ciklusban kérjünk be egy számot (A), valamint egy műveleti jelet. Végezzük el X és A között a műveleti jelben megadott műveletet (+,-,*,/,négyzetre emelés), az eredményt tegyük vissza az X-be. Kérjünk újabb számot és műveleti jelet, és folytassuk ugyanígy. Akkor fejezhetjük be, ha a művelet elvégzése után az eredmény nagyobb lesz mint 10000, vagy műveleti jelnek az egyenlőségjelet adtuk meg. Ekkor írjuk ki a végeredményt.

10. Írj programot, amely véletlen számokat generál 1 és 50 között és a képernyőre írja a számokat. A program addig generáljon számokat, amíg nem generálja a 25-öt.

    10.1. A program írja ki a számok összegét,

    10.2. a páratlan számok darabszámát,

    10.3. valamint a legnagyobb és a legkisebb számot!

11. Egy 12 fős csoport dolgozat jegyeit szeretnénk rögzíteni oly módon, hogy ha 0-t adunk meg, akkor fejeződjön be az adatbevitel, valamint, ha nem érvényes osztályzatot adunk meg (nem 1…5 közötti számot), akkor kérje újra az adatot a program, akár többször is!

    11.1. Az adatbekérés befejeztével írjuk ki, hogy hány jegyet vittünk be!

    11.2. Mennyi az osztályzatok átlaga?

    11.3. Mi a leggyengébb osztályzat és legjobb osztályzat?

    11.4. Határozzuk meg az osztályzatok gyakoriságát!

File-kezelés

12. Készítsünk programot, amely beolvas egy szöveges állományt, majd egy másik állományba kiírja a beolvasott szöveget csupa nagybetűkkel.
13. Írj programot, amely beolvassa a `nevsor.txt` fájl tartalmát egy `nevsor` nevű tömbbe, a „K” betűvel kezdődő neveket kiírja a képernyőre a sorszámukkal együtt (hányadik név a névsorban), az „M” betűvel kezdődő neveket pedig a egy m.txt nevű fájlba írja ki. A `nevsor.txt` fájl minden sora egyetlen nevet tartalmaz.
14. Egy szövegfájlban (`szavak.txt`) soronként egy szót találunk. Írjuk át a szavakat egy másik szövegfájlba (`szavak2.txt`) úgy, hogy megfordítjuk a szavakban a karakterek sorrendjét!
15. Egy szöveges fájlban (`szavak.txt`) soronként egy szót találunk. Írjuk át az adatokat egy másik fájlba (`szavak2.txt`) úgy, hogy soronként három szó szerepeljen szóközzel elválasztva!
16. Egy szövegfájlban (`verseny.txt`) egy verseny eredményeit találjuk. A fájl szerkezete a következő. Az első sorban találunk egy nevet, majd a következő öt sorban egy-egy 0 és 20 közötti számot, ami az egyes versenyszámokra kapott pontszámot jelenti. A továbbiakban az előzőek ismétlődnek.
    16.1. A `továbbjutók.txt` nevű szövegfájlba írjuk ki a továbbjutó versenyzők nevét! Továbbjut az a versenyző, akinek az összesített pontszáma eléri a maximális pontszám 40%-át.
    16.2. Az `összesítés.txt` fájlban jelenjen meg egymás mellett szóközökkel tagolva a név és az összpontszám! A két fájlban a nevek sorrendje azonos legyen!
    16.3. Az `elsők.txt` nevű fájlba írjuk ki az első három legjobb összesített eredményt elérő tanuló adatait csökkenően rendezve! Egy sorban szóközökkel tagoltan jelenjen meg a tanuló neve, egyes feladatokra kapott pontszáma és az összpontszám!
17. A `fordulo1.txt` és a `fordulo2.txt` fájlokban egy verseny 1. és 2. fordulójában elért pontszámokat találjuk (0 – 50 közötti értékek). Mindkét fájlban legfeljebb 100 résztvevő adata található. A résztvevők sorszámmal voltak azonosítva, és a fájlban a sorszámok szerint rendezetten találhatók a versenyzők eredményei. Olvassuk be mindkét fájl tartalmát, számítsuk ki az 1. és 2. fordulóban elért összpontszámokat, majd írassuk ki csökkenő sorrendben az összpontszámokat az `eredmeny.txt` fájlba úgy, hogy a tanulók sorszáma is szerepeljen az összpontszámok után.