2. fejezet

Feladatsorok

2.1. 1994. május 16., nappali tagozat

1. Készítsen **eljárást**, amely a standard bemenetről beolvas egy legfeljebb 10×10-es méretű, egész elemekből álló négyzetes mátrixot, és eldönti, hogy az háromszög mátrix-e! Az eljárásnak van egy input paramétere, amely a mátrix aktuális méretét adja meg, és egy logikai típusú output paramétere, amely akkor igaz, ha a mátrix háromszögmátrix.

2. Adva van egy `szoveg.txt` nevű szöveges állomány, amely soraiban magyar szavak állnak egymástól egy szóközzel elválasztva. Az utolsó szó után nem áll szóköz, hanem közvetlenül a sorvége jel következik. Írjon **programot**, amely végigolvassa az állományt, és

   (a) megszámolja az állományban lévő szavakat;

   (b) létrehoz egy másik szöveges állományt, amelybe a **magas**, **mély**, **vegyes** szavak valamelyikét helyezi el annak megfelelően, hogy az adott szó milyen hangrendű. Az új állományban annyi sor és minden sorban annyi szó legyen, mit az eredetiben volt.

3. Adva van egy egészeket tartalmazó típusos állomány. A mérete akkora, hogy nem fér be a tárba. Rendezze az állományt a saját helyén. Segédállományt nem használhat, segédváltozókat igen.

4. Írjon egy olyan logikai **függvényt**, amely programozási nyelvek (Pascal, C, Ada, LISP, COBOL, ALGOL, . . .) egy halmazát definiálja, majd eldönti, hogy a paramétereként megadott programozási nyelv benne van-e a halmazban. Használjon felsorolásos típust!

5. Adott egy állomány, amelyben az egyes rekordok különböző típusú adatokat tartalmaznak. A rekordok szerkezete a következő:

   1. bájt: a tartalmazott adatok típusa
      - 0: integer
      - 1: real
      - 2: boolean
      - 3: char

   2. bájt: az adott típusú adatok száma (max. 255),
   3. és további bájtok: az adatok.

   Válogassuk le az állományból a karaktereket tartalmazó rekordokat és hozzunk létre belőlük egy szöveges állományt `szo.txt` néven. Ha már létezik ilyen nevű állomány, akkor bővítsük azt.

2.2. 1995. május 15., nappali tagozat

1. Tekintsük az alábbi láncolt listát, amely `node` típusú elemekből áll:

```
type
  node_m = ^node;
  node   = record
             value: real;
             next:  node_m;
           end;
```

Írjon olyan Turbo Pascal függvényt, amely két ilyen listán kompozíciós szorzást végez, azaz az eredménye egy olyan lista, amelyben az első elem a két lista első elemének szorzata, a második elem a két második elem szorzata stb. Ha az egyik lista rövidebb, mint a másik, akkor feltételezzük, hogy a hátralévő elemei nullák. A függvény egy mutatót adjon vissza a szorzatlista első elemére (fej). A függvény két paramétere mutató az első, illetve a második lista első elemére (fej):

```
function listmult( list1, list2: node_m ): node_m;
```

2. Írjon olyan hash függvényt, amely egy paraméterként átvett sztringre előállít egy számot (címet) 1 és N között. A függvény vegye figyelembe a sztringben előforduló konkrét karaktereket és a sztring hosszát. A függvény első paramétere a sztring, a második az N szám legyen, a visszaadott érték pedig a cím:

```
function hash( key: string; N: longint ): longint;
```

3. Tekintsünk egy keresőfát az alábbi csomópont-szerkezettel:

```
type
  node_m = ^node;
  node   = record
             value: integer;
             left:  node_m;
             right: node_m;
           end;
```

A `left` a bal oldali, a `right` a jobb oldali részfára mutat.

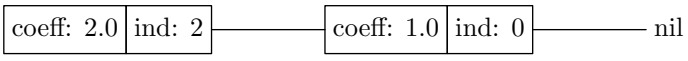
A bal oldali részfában lévő elemek kisebbek a gyökérben lévő elemnél (`value`), a jobb oldali részfában lévő elemek nagyobbak. Írjon olyan függvényt, amely meghatározza, hogy a paraméterként kapott egész hányadik szinten van a fában. A gyökér szintje legyen 0. A visszaadott érték a szintszám. Ha a keresett érték nem fordul elő a fában, a visszaadott érték legyen -1.

```
function depth( x: integer ): integer;
```

4. Egy valós polinomot a következőképpen ábrázolunk láncolt listában: a lista elemei a következő típusúak:

```
type
  element_m = ^element;
  element   = record
                coeff: real;
                ind:   integer;
                next:  element_m;
              end;
```

Az `ind` a kitevőt, a `coeff` az adott taghoz tartozó együtthatót jelenti. Például a $2x^2+1$ polinomot a következőképpen ábrázoljuk:



Írjon olyan függvényt, amelyik a polinom helyettesítési értékét kiszámolja! A függvény első paramétere mutató az első elemre (fej), a második pedig pedig az a hely, ahol a helyettesítési értéket venni kell.

```
function value( head: element_m; x: real ): real;
```

5. Készítsen egy függvényt, amely kiértékel egy kifejezésfát! A kifejezésfa levélelemei tartalmazzák a numerikus konstansokat, míg a további csomópontok a műveleti jeleket (négy alapművelet: `'+'`, `'-'`, `'*'`, `'/'`). A függvény paraméterként egy mutatót kap a fa gyökérelemére, függvényértékként pedig a kifejezés értékét szolgáltatja. Feltesszük, hogy a megadott kifejezés helyes.

```
type
  pnode = ^node;
  node  = record
            left:  pnode;
            right: pnode;
            case boolean of
              true:  ( value:  real; );
              false: ( opcode: char; );
          end;

function eval( root: pnode ): real;
```

6. Készítsen egy függvényt, amely eldönti egy négyzetes mátrixról, hogy háromszögmátrix-e, illetve hogy diagonális mátrix-e! A függvény paraméterként egy mutatót és egy méretet kap. A méret paraméter a mátrix sorainak (és oszlopainak) a számát tartalmazza, a mutató pedig egy egydimenziós egész elemű tömbre mutat. A mátrix elemei ezen tömbben helyezkednek el sorfolytonosan.

```
const
  maxmatrix = 100;
type
  mtipus  = ( also_haromszog, felso_haromszog, diagonalis, egyeb );
  pmatrix = ^array[ 1..maxmatrix*maxmatrix ] of integer;

function matrix_teszt( matrix: pmatrix; meret: integer ): mtipus;
```

7. Készítsen egy függvényt, amely egy egész számot átalakít egy adott számrendszerbeli sztringgé! A függvény első paramétere az átalakítandó szám, második paramétere a számrendszer alapja, a visszatérési érték pedig az eredmény sztring. A számrendszer alapszáma a $2, \ldots, 35$ intervallumba esik.

```
function konvertal( mit: longint; alap: integer ): string;
```

2.3. Időpontja ismeretlen, nappali tagozat

1. Tekintsük az alábbi szerkezettel adott rendezett bináris fát!

```
type
  str       = string[ 20 ];
  node_m    = ^node;
  node      = record
                job_nev: str;
                memory:  longint;
                next:    node_m;
              end;
  joblist_m = ^joblist;
  joblist   = record
                job_nev: str;
                next: joblist_m;
              end;
```

A fa a `job_nev` mező alapján van rendezve növekvő sorrendben.

Írjon olyan Turbo Pascal függvényt, amely egy láncolt listában növekvő sorrendben visszaadja azokat a `job_nev` értékeket, amelyekre a `memory` értéke egy adott `x` értéknél nagyobb vagy egyenlő. A függvény első paramétere a fa gyökere, a második paramétere pedig az `x`. A visszaadott lista szerkezete: `joblist`.

```
function usage( root: node_m; x: longint): joblist_m;
```

2. Írjon olyan eljárást, amely az alábbi szerkezetű listával megvalósított veremben csak a benne előforduló legnagyobb számot hagyja meg. A veremben különböző számok vannak.

```
type
  stack_m = ^stack;
  stack   = record
              x:    longint;
              next: stack_m;
            end;
```

Az eljárás paramétere: mutató a verem tetejére.

```
procedure max( top: stack_m );
```

3. Mutasson a **ptr** nevű mutató egy 52 elemű `integer` típusú elemekből álló memóriaterületre. Tekintsük az alábbi szerkezetű láncolt listát!

```
type
  lapt   = 1..52;
  node_m = ^node;
  node   = record
             mit  : lapt;
             mivel: lapt;
             next : node_m;
           end;
```

E lista elemei egy cserét határoznak meg a fenti (`ptr`) tömbben. A `mit` indexű elemet fel kell cserélni a `mivel` indexűvel. Írjon egy olyan eljárást, amely az összes, a listában felsorolt cserét elvégzi a megadott tömbben. Az eljárás első paramétere mutató a tömb (`ptr`) első elemére, a második paraméter pedig mutató a lista első elemére.

```
procedure csere( ptr: ^integer; list: node_m );
```

4. Adva van egy szövegfájlban egy valósakból álló zárt intervallumsorozat. Minden intervallum új sorban van, a két végpont pedig vesszővel van elválasztva. Például:

```
12.45, 12.55
1.0, 2.5
```

Írjon olyan Turbo Pascal eljárást, amely a képernyőre kiírja azt a legbővebb intervallumot, amely egy adott pontot tartalmaz, de nincs a felsorolt intervallumokkal egyetlen közös pontja sem. Az eljárás első paramétere a szövegfájl neve, a második paramétere pedig a kérdéses pont.

```
procedure maxi( fnev: string; point: real );
```

5. Készítsen egy olyan függvényt, amely egy tárban ábrázolt, `integer` típusú elemekből álló mátrixról eldönti, hogy szimmetrikus-e! Ha szimmetrikus, a visszaadott érték `true`, ellenkező esetben `false`.

A mátrix egy összefüggő memóriaterületen van sorfolytonosan tárolva. A függvény paramétere a mátrix első sorában lévő első elemnek a címe.

```
function szimmetrikus( mx: ^integer ): boolean;
```

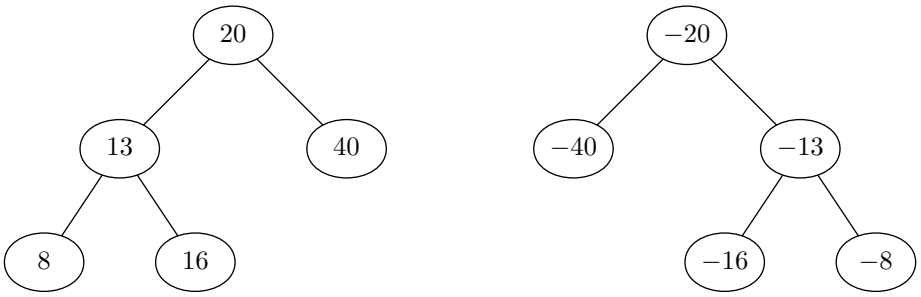
6. Adva van a memóriában egy bináris keresési fa az alábbi csomópont-szerkezettel:

```
type
  key_t  = integer;
  node_m = ^node;
  node   = record
             key:  key_t;
             next: node_m;
           end;
```

Írjon olyan Turbo Pascal eljárást, amely minden csomópont bal és jobb oldali gyermekét felcseréli és ugyanakkor a `key` mezőben található egészeket kicseréli a -1-szeresükre (`-key`). (Az eredmény szintén egy bináris fa.) Az eljárás paramétere a fa gyökere.

```
procedure swaptree( root: node_m );
```

Példa:



2.4. Időpontja ismeretlen, nappali tagozat

1. Írjon az alábbi prototípussal megegyező függvényt:

   ```
   char *concat( int db, ... );
   ```

   A függvény működése: az első változóban megadott darabszámú `char *` típusként átvett változók által meghatározott karakterláncokat fűzze össze, és az új karakterlánc kezdőcímét adja vissza.

2. Írjon az alábbi prototípussal megegyező függvényt:

   ```
   int include( FILE *inp, FILE *out );
   ```

   A függvény működése: a preprocesszor hasonló funkcióját látja el. Az első paraméterben megadott állományból a `#include <állománynév>` alakú sorokat helyettesíti a megadott állomány tartalmával. Az új állomány is tartalmazhat hivatkozásokat. Az így kifejtett szöveg a második paraméterben megadott állományba kerüljön. A függvény visszatérési értéke 0, ha sikeres volt a végrehajtás; 1, ha megnyitási hiba történt; 2, ha újból meg akarunk nyitni egy már használatban lévő állományt (végtelen ciklus).

3. Írjon olyan függvényt, amely egy paraméterben megkapott fájlról eldönti, hogy 8 bites-e, azaz a bájtjainak legmagasabb helyiértékű bitje magas-e vagy nem. (Ha egy bájt 8 bites, akkor az egész fájl 8 bites.) Ha a fájl 8 bites, akkor 1-et, különben 0-t ad vissza. Prototípus:

   ```
   int eight( FILE *fp );
   ```

4. Készítsen függvényt, amely beolvas egy X11 bitmap fájlt a memóriába. A bitmap számára a függvény foglaljon helyet. A függvény prototípusa:

   ```
   void *readxbm( FILE * );
   ```

   Az X11 bitmap formátum: a fájl tartalmaz két `#define` sort, amelyek a bitmap szélességét és magasságát adják meg. Ezt követi egy C szintaktikájú kezdőértékkel ellátott tömb deklaráció. A bitmap minden pontot egy biten ábrázol, soronként bájthatárra igazítva. Elnevezések:

   ```
   #define filename_width  100
   #define filename_height 150
   static char filename_bits[] = { 0x12, 0x34, ... };
   ```

5. Készítsen függvényt, amely veremkalkulátort valósít meg! A függvény bemenő paramétere az input állomány, visszatérési értéke egy valós szám, a műveletek eredménye. Amennyiben a fájlból számot olvas a függvény, az bekerül egy verembe, amennyiben műveleti jelet, úgy azt végre kell hajtani a verem tetején található két értékkel, és az eredmény visszakerül a verembe. A függvény visszatérési értéke az a szám, amely az input fájl végének elérésekor a verem tetején található. Minimálisan a négy alapműveletet kell implementálni, s tekintettel kell lenni a negatív számokra.

```
float scale( FILE * );
```

6. Készítsen függvényt, amely paraméterben megkapott bitképbe téglalapot rajzol. A képben egy képpont 1 bites, 0 vagy 1 értékű lehet. A téglalap rajzolása során a téglalap oldalaihoz tartozó pontokat 1 értékűre kell állítani. A függvény első paramétere a kép címe a memóriában, a második a kép mérete képpontokban x irányban, a harmadik az y irányú méret, a negyedik és ötödik a bal felső sarok, a hatodik és hetedik a téglalap jobb alsó sarkának koordinátái.

```
Rec( char *kep, int xw, int yw, int x1, int x2, int y1, int y2 );
```

2.5. 1996. május 18., nappali tagozat

1. Egy **bináris fa tükrözésének** nevezzük a következőt: a fa gyökerétől indulva, szintenként megcseréljük minden elem bal és jobb oldali részfáját.

   Írjon egy **logikai függvényt**, amely paraméterként megkap egy **egészeket** tartalmazó bináris fa gyökerét címző **mutatót**, **tükrözi** a fát, és **eldönti**, hogy a tükrözés után kapott fa **megegyezik-e** az induló fával!

2. Egy szöveges állományban a sorokban **postfix** alakú kifejezések vannak adva. A kifejezésekben a $+$, $-$, $*$, / műveleti jelek és előjel nélküli **egész** számok mint operandusok szerepelnek. A műveleti jeleket és az operandusokat egy-egy szóköz választja el egymástól.

   Írjon egy **eljárást**, amely paraméterként megkapja az **állomány nevét**, és **kiértékeli** a kifejezéseket, ezek értékét pedig elhelyezi egy **típusos** állományban!

3. Adva van egy **típusos** állomány a következő **rekordszerkezettel**:

kulcs	–	egész
név	–	szöveges
lakcím	–	szöveges

   Írjon egy **eljárást**, amely bemenő paraméterként megkapja az **állomány nevét**! Az állomány rekordjaiból hozzon létre egy **keresőfát**! Kimenő paraméter legyen a **fa gyökerét címző mutató**!

4. Adva van egy **szöveges állomány**, amely **maximum 10** hosszúságú, az **angol** ábécé betűiből álló, egymástól **1 szóközzel** elválasztott szavakat tartalmaz.

   Írjon **programot**, amely a szavakat egy **egyirányban láncolt rendezett listában** helyezi el! Írassa ki **képernyőre** azt a **szót** és az **előfordulás számát**, amely a leggyakrabban fordul elő az állományban (vigyázat, több ilyen is lehet)!

5. Írjon egy **eljárást**, amely bemenő paraméterként kap egy **valós** elemekből felépített **bináris fa** gyökerét címző **mutatót**! A bináris fához készítse el a **hierarchikus listánál** megismert **zárójeles reprezentációt**, és helyezze el azt egy **szöveges** állományban!

6. Írjon egy **eljárást**, amely egy **valósakat** tartalmazó **tetszőleges kétdimenziós** tömb sorait úgy rendezi át, hogy azok a sorokban elhelyezett elemek **átlaga** szerint **növekvően** helyezkedjenek el!

7. Írjon egy **hash függvényt**, amelynek bemenő paraméterei egy **sztring** és egy **egész** szám. A függvény határozzon meg **egy számot** 1 és az egész szám között, a meghatározásnál vegye figyelembe a sztring **hosszát** és a sztringben előforduló **konkrét karaktereket**!

2.6. 1996. június 13., nappali tagozat

1. Egy **bináris fa tükrözésének** nevezzük a következőt: a fa gyökerétől indulva, szintenként megcseréljük minden elem bal és jobb oldali részfáját.

   Írjon egy **logikai függvényt**, amely paraméterként megkap egy **egészeket** tartalmazó bináris fa gyökerét címző **mutatót**, **tükrözi** a fát, és **eldönti**, hogy a tükrözés után kapott fa **megegyezik-e** az induló fával!

2. Egy szöveges állományban a sorokban **postfix** alakú kifejezések vannak adva. A kifejezésekben a $+$, $-$, $*$, / műveleti jelek és előjel nélküli **egész** számok mint operandusok szerepelnek. A műveleti jeleket és az operandusokat egy-egy szóköz választja el egymástól.

   Írjon egy **eljárást**, amely paraméterként megkapja az **állomány nevét**, és minden kifejezéshez **felépíti** a megfelelő (szétszórt módon ábrázolt) bináris fát, továbbá a fák gyökereinek címét **elhelyezi** egy **egyirányban láncolt listában**. Az eljárás kimenő paramétere a **lista első elemét címző mutató** legyen.

3. Adva van egy **típusos** állomány a következő **rekordszerkezettel**:

kulcs	–	egész
név	–	szöveges
lakcím	–	szöveges

   Írjon egy **eljárást**, amely bemenő paraméterként megkapja az **állomány nevét**! Az állomány rekordjaiból hozzon létre egy **rendezett táblázatot**! Kimenő paraméter legyen a táblázat elemeinek **száma** és (reprezentációtól függően) a **táblázat** vagy a **táblázat első elemét címző mutató**!

4. Adva van egy **szöveges állomány**, amely **maximum 10** hosszúságú, az **angol** ábécé betűiből álló, egymástól **1 szóközzel** elválasztott szavakat tartalmaz.

   Írjon **programot**, amely meghatározza, hogy az állományban **hány különböző szó** van, és ezekhez megállapítja az előfordulásuk **gyakoriságát** is! A különböző szavakat a gyakoriságokkal együtt egy **típusos állományban** kell elhelyezni!

5. Írjon egy **függvényt**, amely bemenő paraméterként kap egy **valós** elemekből felépített **keresőfa gyökerét címző mutatót** és egy **valós** értéket! A függvény határozza meg, hogy az adott érték a fa **hányadik szintjén** fordul elő! A gyökér szint a 0. szint. Ha az érték nincs a fában, a visszaadott érték: -1.

6. Írjon egy **eljárást**, amely egy **valósakat** tartalmazó **kvadratikus mátrix** főátló alatti, főátló feletti és főátlóban álló elemeinek az **összegét** határozza meg!

7. Adva van egy **bináris fa folytonos reprezentációja** három egydimenziós tömbben (`adat`, `bal`, `jobb`). Az első elem a gyökérelem.

   Írjon egy **eljárást**, amely bemenő paraméterként megkapja a **fa elemeinek számát** és a **három tömböt**. Az eljárás **állítsa elő** a fa **szétszórt reprezentációját**, és kimenő paraméterként adja meg a **gyökér címét**!

2.7. 1996. december 14., nappali tagozat

1. Írjon olyan programot, amely két állomány tartalmát hasonlítja össze, és az első különbség pozícióját a standard hibakimenetre írja! Amennyiben a két állomány tartalma azonos, a program nem ír ki semmit. A program egy vagy két állománynevet kaphat futásidejű paraméterként (argumentumként). Ha egy paramétert kapott, akkor a standard bemenetet kell tekinteni a másik hasonlítandó állományként.

2. Írjon olyan programot, amely egy állománynevet kap futásidejű paraméterként (argumentumként). Ezen állomány tartalmát olvassa be egy kétirányban láncolt listába. Ezután jelenítse meg a listába felolvasott állomány 24 sorát a képernyőn úgy, hogy a 'B' billentyű lenyomására egy sorral visszafelé, az 'N' hatására egy sorral előrefelé lehessen „lapozni". Kilépés a 'Q' billentyűvel.

   Megjegyzés: az állomány sorai maximum 80 karakter hosszúak.

3. Írjon olyan függvényt, amely két állománynevet kap paraméterként. Az egyik állomány (alapállomány) alkalmazottak adatait tárolja rendezetlenül. A másik állomány (indexállomány) egy-egy rekordja az alapállomány rekordjainak elsődleges azonosítóját és az adott alaprekord fizikai pozícióját tartalmazza. A függvény feladata – az indexállomány felhasználásával – az alapállomány rekordjainak rendezett kiíratása a képernyőre.

   Az alapállomány rekordszerkezete:

```
struct employee
{
  char name[ 50 ];
  char pid[ 12 ];   /* személyi azonosító, elsődleges kulcs */
  long salary;
};
```

   Az indexállomány rekordszerkezete:

```
struct employee
{
  char pid[ 12 ];
  long pos;
};
```

4. Írjon olyan függvényt, amelynek prototípusa a következő:

```
char *char_dup( char *str, char c );
```

   A függvény feladata a kapott karakterláncban a második paraméterként megadott karakter összes előfordulásának megduplázása, s az eredmény elhelyezése egy megfelelő méretű, a függvény által allokált karakterláncban. A paraméterként kapott karakterlánc tartalma maradjon változatlan.

   A függvény visszatérési értéke a létrehozott karakterlánc címe legyen, illetve `NULL`, ha nem sikerült a memóriafoglalás!

5. Írjon olyan függvényt, amely egy bináris fában megadott kifejezést értékel ki! A kifejezésekben csak a négy alapműveletre ($+$, $-$, $*$, $/$) kell számítani. A kifejezésekben szereplő numerikus értékek előjeles egész számok. A kifejezés megadására szolgáló fa egy csomópontjának típusa:

```
struct node
{
  char operator;
  int value;
  struct node *right;
  struct node *left;
};
```

   A struktúra `operator` nevű mezőjében vagy a műveleti jel (`'+'`, `'-'`, `'*'`, `'/'`), vagy egy szóköz szerepel, ez utóbbi esetben a struktúra `value` mezőjének tartalma egy egész szám (ekkor a csomópont levélelem). A függvény visszatérési értéke a fában megadott kifejezés értéke legyen!

   A függvény prototípusa:

```
float evaluate( struct node *expression );
```

2.8. 1997. május 26., nappali tagozat

1. Adva van egy `feladat1.txt` nevű **szöveges** állomány, amelynek soraiban **magyar** szavak helyezkednek el úgy, hogy minden szó után áll egy **szóköz**. Az állomány sorainak száma tetszőleges, egy sorban akárhány szó lehet, a szavak maximális hossza **15**.

   Írjon **programot**, amely meghatározza az állományban előforduló **különböző** szavak **előfordulási gyakoriságát**! Feltesszük, hogy **legfeljebb 500** különböző szó lehet. Az eredményt `szo`, `gyak` felépítésű rekordokban helyezze el a `feladat2.dat` nevű **típusos** állományban!

2. Az előző feladat eredmény állományát felhasználva írjon olyan **függvényt**, amely megadja a **leggyakoribb** szavakat!

3. Az első feladat eredmény állományát felhasználva írjon olyan **függvényt**, amely megadja az ábécé sorrendben **legnagyobb** szót!

4. Adva van a `feladat3.txt` szöveges állomány, amely egy **Pascal programot** tartalmaz. Írjon **eljárást**, amely a program szövegéből eltávolítja a **megjegyzéseket**, és az eredményt helyezze a `feladat4.txt` szöveges állományba!

5. Billentyűzetről megadunk egy **teljesen zárójelezett kifejezést**, amely csak egyetlen betűből álló azonosítójú változókat és a $+$, $-$, $*$, / kétoperandusú operátorokat tartalmazhatja a zárójeleken kívül. Írjon **programot**, amely beolvassa a kifejezést, és ha az szintaktikusan helyes, akkor felépíti a hozzátartozó **bináris fát**, ha pedig hibás, akkor képernyőre írja azt a részkifejezést, amely még hibátlan volt, és ezután megjelöli a hiba okát! Feltesszük, hogy redundáns zárójelek nincsenek.

6. Írjon egy **eljárást**, amely az előző feladat szintaktikusan helyes kifejezését képernyőre írja és megjelöli azt az operátort, amely **utoljára** lesz végrehajtva a kifejezés kiértékelésénél!

2.9. 1997. június 10., nappali tagozat

1. Írjon egy **eljárást**, amely bemenő paraméterként megkap egy **létező szöveges állomány** nevet és egy **sztringet**. Az állomány sorainak száma tetszőleges, egy sor **maximálisan 999** karakterből állhat. Az eljárás határozza meg a **legrövidebb** és **leghosszabb** sorokat, és ezeket írja egy szövegállományba, továbbá döntse el, hogy **van-e** az állományban olyan sor, amely **megegyezik** a megadott sztringgel!

2. Adva van egy egészeket tartalmazó **egyirányban láncolt lista**. Írjon **eljárást**, amely megkapja bemenő **paraméterként** a **lista fejét** és egy **verembe** helyezi (a listabeli előfordulás sorrendjében) azokat at elemeket, melyeknek a listaelemek átlagától való eltérése kisebb mint 2!

3. Adva van egy **Pascal azonosítókat** tartalmazó szöveges állomány. Az azonosítókat **vessző** választja el. Írjon **programot**, amely az azonosítókból létrehoz egy olyan **keresőfát**, ahol a csomópontok szerkezete: **azonosító**, **gyakoriság**. Adja meg a **leggyakrabban** előforduló azonosítót!

4. Írjon egy **logikai függvényt**, amely egy billentyűzetről megadott karaktersorozatról eldönti, hogy az **szabályos Pascal valós konstans-e**!

5. Egy **szöveges állományban** az egyes sorokban **szabályos prefix** alakú kifejezések vannak, amelyekben a $+$, $-$, $*$, / operátorok és egész számok mint operandusok szerepelnek. Az operátorokat és operandusokat egymástól egy **szóköz** választja el. Írjon **programot**, amely kiértékeli a kifejezéseket, és ezek eredményét elhelyezi egy **típusos** állományban!

2.10. 1997. június 16., nappali tagozat

1. Írjon egy **sztring** típusú **függvényt**, amely bemenő paraméterként megkapja egy kifejezés fájának gyökerét címző mutatót, és előállítja a kifejezés **teljesen zárójelezett** alakját!

2. Írjon egy **eljárást**, amely bemenő paraméterként megkap egy **létező szöveges állomány** nevet és egy **sztringet**! Az állomány sorainak száma tetszőleges, egy sor **maximálisan** 512 karakterből állhat. Az eljárás hozzon létre egy egyirányban láncolt listát azon sorokból, amelyeknek a **vége megegyezik** a megadott sztringgel! A további sorokat írja egy másik szöveges állományba!

3. Adva van egy **típusos állomány a következő** rekordszerkezettel:

SOR	–	byte
OSZLOP	–	byte
ERTEK	–	real

Az állomány egy ritka mátrix **háromsoros oszlopfolytonos** reprezentációját tartalmazza. Az **első rekord** SOR és OSZLOP értéke a mátrix **méretét** adja meg. Írjon **eljárást**, amely bemenő paraméterként megkapja az állomány nevét, és képernyőre írja a mátrix azon **sorát**, amely a legtöbb nem-nulla elemet tartalmazza (feltesszük, hogy csak egy ilyen van)!

4. Írjon **függvényt**, amely egy `string[12]` típusú elemeket tartalmazó **veremből** eltávolítja azokat az elemeket, amelyek **nem Pascal azonosítók**!

5. Egy **nem bináris** fa elemeit reprezentáljuk egy **sztring** típusú adatelemmel és egy **ötelemű** mutatóvektorral. Írjon **mutató** típusú **függvényt**, amely paraméterként megkapja a fa gyökérmutatóját, és előállítja annak **bináris** reprezentációját! A visszaadott érték a bináris fa gyökerét címezi.

2.11. 1998. május 18., nappali tagozat

1. Adva van egy szövegállomány, amely egy szintaktikusan helyes Pascal programot tartalmaz. A programban alprogramok nincsenek, és egyetlen `var` utasítás szerepel benne. Írjon **programot**, amely megadja, hogy a deklarált **változókra** mely **sorokban** történik hivatkozás a szövegben. Az output a **képernyőn** keletkezzen! A sorokat 1-től kezdődően sorszámozza!

2. Írjon egy **függvényt**, amely paraméterként egy **nemüres bináris fa** gyökérmutatóját kapja meg, és visszaadja, hogy melyik a **legalacsonyabb szintszámú** olyan elem, amelynek nincs baloldali rákövetkezője! A gyökér szintszáma 0.

3. Írjon **programot**, amely **billentyűzetről** megadott **kifejezésekről** eldönti, hogy **szabályos Pascal** kifejezések-e! Minden kifejezést egy **Enter** zár le, benne szóköz nem lehet. A kifejezés sorainak végét egy **szóköz Enter** sor begépelése jelzi. A kifejezés operandusai **egybetűs azonosítók** és előjel nélküli **egész számok** lehetnek. Operátorok: $+$, $-$, $*$, $/$. **Zárójelek** tetszőlegesen alkalmazhatók. A program írja a képernyőre a kifejezést, ha az szabályos, akkor adja ezt az üzenetet, egyébként jelölje meg azt az elemet, ami a hibát okozza és írjon ki hibaüzenetet!

4. Adva van egy szövegállomány, benne **egy szóközzel elválasztott** (nem feltétlenül különböző) **szavak**. Két szó „hangzási távolságán” értjük a következőt: a szavakat az elejükön kezdve összehasonlítjuk karakterenként. Eltérő karakterek esetén a résztávolság értéke 1, ha azonos pozíción azonos mássalhangzó van, akkor 0.5, azonos magánhangzó esetén 0.1. A távolságot a résztávolságok összege adja.

   Írjon **eljárást**, amely paraméterként megkapja az állomány **nevét**, és egy **szót**, majd egy output állományban elhelyezi azt a **tíz** szót, amely **legközelebb** van a megadott szóhoz! A szavak mellett szerepeljen a **távolság** is.

5. Adva van egy szöveges állomány, amely egymástól **egy szóközzel** elválasztott **különböző szavakat** tartalmaz. Írjon **programot**, amely feldolgozza az állományt úgy, hogy a szavakból egy **hash függvény** segítségével **1000 elemű táblázatot** állít elő. A hash függvényt tessék úgy megválasztani, hogy az vegye figyelembe a szavak hosszát is! A szinonimákat láncolni kell.

6. Egy rendezett nembináris fát, amelyben minden elemnek legfeljebb K rákövetkezője lehet, reprezentáljunk úgy, hogy minden elem mellé felveszünk egy K elemű mutatóvektort. Írjon **eljárást**, amely úgy járja be a fát, hogy először feldolgozza a **gyökérelemet**, majd **jobbról balra** a rákövetkezőit!

7. Egy operációs rendszerben a memória szabad helyeit **egyirányban láncolt listával** kezeljük, **méret**, **következő szabad hely címe** formában, a **cím** szerint **növekvően**. Készítsen egy olyan **eljárást**, amely paraméterként megkapja a lista **fejét** és egy **tartomány címét és méretét**, és ezt a tartományt felfűzi a listába. Ha a tartomány összeér egy másik tartománnyal (folytonos tárterület!), akkor azokat össze kell olvasztani és a listában csak egyszer kell szerepeltetni őket.

2.12. 1998. május 29., nappali tagozat

1. Adva van egy szövegállomány, amely egy szintaktikusan helyes Pascal programot tartalmaz. A főprogramban egyetlen `const` utasítás van, amelyben **skalár** nevesített konstansokat deklaráltunk.

Írjunk **programot**, amely a program szövegében a nevesített konstansok **neveit kicseréli az értékükre**!

2. Adva van egy szövegállomány, amely **magyar szavakat** tartalmaz. Minden szó **után egy szóköz** áll. A sorokban maximum 30 szó lehet. Egy szó maximum 10 karakterből áll. Írjon **programot**, amely egy másik szövegállományba másolja azokat a **sorokat**, amelyekben **nem** fordul elő a **leggyakoribb** szó (több ilyen is lehet)!

3. Adva van egy szabályos Pascal kifejezés **fája**, az elemek **sztringek**. Írjon egy olyan **függvényt**, amely megkapja a fa **gyökérmutatóját**, és **sztringként** visszaadja a kifejezés **teljesen zárójelezett** alakját!

4. Egy operációs rendszerben a memória szabad helyeit **egyirányban láncolt listával** kezeljük, **méret**, **következő szabad hely címe** formájában, a **cím** szerint **növekvően**. Készítsen egy olyan **eljárást**, amely megkapja a lista **fejét** és egy **méretet**, és megadja annak a szabad helynek a **címét**, amelyik a méretet felülről leginkább megközelíti! Ha több ilyen van, akkor válasszuk a **legnagyobb** címűt! Ha a megtalált szabad hely mérete 2K-nál kevesebbel tér el a megadott mérettől, akkor **töröljük** a listából az őt reprezentáló elemet, egyébként maradjon a listában egy **megfelelően csökkentett** méretű elem!

5. Adva van egy tipizált állomány, amely **kvadratikus ritka mátrixok négysoros reprezentációját** tartalmazza (a mátrix elemei **valósak**, a standard elem a **nulla**). Írjon olyan **logikai függvényt**, amely tetszőleges ilyen állomány esetén eldönti, hogy a mátrix **főátlója alatti** elemek között van-e **nem nulla** elem!

2.13. 1998. június 2., nappali tagozat

1. Adva van egy szövegállomány, amely egy szintaktikusan helyes Pascal programot tartalmaz. Írjon **programot**, amely meghatározza, hogy az **alapszavak** milyen **gyakorisággal** fordulnak elő a programban! Az eredményt egy **szövegállományban** helyezze el!

2. Adva van egy szövegállomány, amely **magyar szavakat** tartalmaz. Minden szó **után egy szóköz** áll. A sorokban maximum 30 szó lehet. Egy szó maximum 10 karakterből áll. Írjon **programot**, amely egy másik **szövegállományba** másolja át azokat a **szavakat** (utánuk egy szóközzel), amelyekben **több** a magánhangzó, mint a mássalhangzó! Az output állományban ne legyenek sorok!

3. Adva van egy szabályos Pascal kifejezés **fája**. Írjon egy olyan **függvényt**, amely megkapja a fa **gyökérmutatóját** és meghatározza, hogy **hány egyoperandusú operátor** van benne!

4. Egy operációs rendszerben a memória szabad helyeit **egyirányban láncolt listával** kezeljük, **méret**, **cím** formában, a **cím** szerint **növekvően**. Készítsen egy olyan **eljárást**, amely paraméterként megkapja a lista **fejét** és egy **méretet**, és meghatározza azon **két** szabad hely **címét**, amelyek mérete **együttesen** a megadott méretet felülről a legjobban közelíti (a megadott méret nagyobb bármely listabeli elem méreténél).

5. Adva vannak **tipizált** állományok, amelyek mindegyike egy-egy **kvadratikus ritka mátrix négysoros reprezentációját** tartalmazza (a mátrix elemei **egészek**, a standard elem a **nulla**), Az állomány első eleme a mátrix méretét tartalmazza **0, 0, méret, 0** formában. Írjon olyan **logikai függvényt**, amely tetszőleges ilyen állomány esetén eldönti, hogy a mátrix **szimmetrikus-e**!

2.14. 1998. december 12., nappali tagozat

1. Adva van egy szövegállomány, amely egy szintaktikusan helyes Pascal programot tartalmaz, amelyben vannak **függvénydeklarációk**. Írjon **eljárást**, amely paraméterként megkapja az **állomány nevét**, és egy **másik** szövegállományba átírja a főprogram **első függvényének** a szövegét!

2. Adva van egy **egészeket** tartalmazó **keresőfa**, és tudjuk, hogy az elemek között van olyan, amelynek értéke megegyezik az elemek **átlagával**. Írjon **függvényt**, amely paraméterként megkapja a fa gyökerét címző mutatót, és meghatározza, hogy a rendezett elemsorozatban **hányadik** a fenti elem!

3. Írjon **programot**, amely egy **billentyűzetről** megadott karaktersorozatról eldönti, hogy az szabályos Pascal kifejezés-e! A kifejezés operandusai **kétkarakteres** azonosítók, illetve háromjegyű előjel nélküli **egész számok** lehetnek. Operátorok: $+$, $-$, $*$, $/$. Zárójelek tetszőlegesen szerepelhetnek. A program írja **képernyőre** a beolvasott karaktersorozatot. Adja a `szabályos kifejezés` üzenetet, ha szabályos, és adjon értelmes hibaüzenetet, ha nem szabályos!

4. Adva van egy **sztringeket** tartalmazó **multilista** két mutatóval. Az egyik lánc a **bekerülés sorrendjében** fűzi fel az elemeket, tehát mindig a végén bővítünk. A másik lánc a sztringeket **hosszuk** szerint **csökkenő** sorrendben tartalmazza. Azonos hosszak esetén szintén a bekerülési sorrendet kell tartani. Írjon **eljárást**, amely **paraméterként** megkapja a két láncfejet és egy sztringet, és a sztringet **beilleszti** a multilistába!

5. Adva van egy szövegállomány, amely soronként egymástól vesszővel elválasztott **egész** értékpárokat tartalmaz. Tekintsük ezeket intervallumoknak. Maximum ezer sor van. Írjon **programot**, amely **képernyőre** írja azt az intervallumot (több ilyen is lehet), amely a **legtöbb** másik intervallumot tartalmazza!

2.15. 1999. május 17., nappali tagozat

1. Írjon egy olyan **függvényt**, amely egy **nemüres**, **nembináris** fa **bináris** reprezentációját állítja elő! A nembináris fában minden elemnek legfeljebb **5** rákövetkezője lehet.

2. Adva van egy C forrásállomány, a forrásszöveg elején paraméter nélküli **define** makrókkal. Írjon **programot**, amely elvégzi a makróhelyettesítést!

3. Írjon **függvényt**, amely egy **tetszőleges bináris fában** megadja azon elemek **darabszámát**, amelyeknek nincs **jobb oldali** rákövetkezőjük!

4. Írja meg azt a **blokkot**, amely kiértékel egy billentyűzetről megadott **prefix** kifejezést! A kifejezésben csak a kétoperandusú $+$, $-$, $*$, $/$ operátorok és előjel nélküli egész számok mint operandusok szerepelnek. Az operátorokat és operandusokat egy-egy **szóköz** választja el egymástól. Ha a kifejezés **szabályos**, akkor a **képernyőn** jelenjen meg az értéke! Ha **nem szabályos**, akkor írja képernyőre a **kifejezést**, és jelölje meg a hiba **helyét** és **okát**!

5. Írjon **programot**, amely egy szöveges állományból leválogatja és képernyőre írja azokat a szavakat, amelyek egy billentyűzetről megadott szó anagrammái (pontosan ugyanazokat a betűket tartalmazzák)! Az állományban a szavakat egy szóköz választja el. A szavak az angol ábécé betűit tartalmazzák. A kis- és nagybetűket nem különböztetjük meg.

2.16. 1999. május 21., esti tagozat

1. Írjon egy **függvényt**, amely **tetszőleges bináris fában** megadja azon elemek **darabszámát**, amelyeknek **két** rákövetkezőjük van!

2. Írja meg azt a **blokkot**, amely kiértékel egy **billentyűzetről** megadott **postfix** kifejezést! A kifejezésben csak a kétoperandusú $+$, $-$, $*$, $/$ operátorok és előjel nélküli egész számok mint operandusok szerepelnek. Az operátorokat és operandusokat egy-egy szóköz választja el egymástól. Ha a kifejezés **szabályos**, akkor a **képernyőn** jelenjen meg az értéke. Ha nem szabályos, akkor írja képernyőre a **kifejezést** és a `Hiba!` üzenetet!

3. Adva van egy **szavakat** tartalmazó szöveges állomány. A szavakban csak az angol ábécé betűi szerepelnek. A szavakat egy-egy szóköz választja el egymástól. Írjon **programot**, amely egy **szövegállományba** írja azokat a szavakat, amelyek **többször** is előfordulnak az állományban!

4. Írjon **függvényt**, amely egy **valósakat** tartalmazó kétdimenziós tömb azon sorainak **indexét** adja meg, amelyekben az elemek **összege** a **legkisebb**!

5. Írjon **programot**, amely egy szöveges állomány azon sorait, amelyek megegyeznek a fordítottjukkal, egy másik szöveges állományba másolja át!

2.17. 1999. december 10., levelező tagozat

1. Adva van egy `szavak.txt` nevű szöveges állomány, amelyben különböző **angol** szavak vannak. Minden szó után **egy szóköz** áll.

   Két szó „távolságán" értsük a következőt: a szavakat karakterről karakterre összehasonlítjuk. Eltérő karakter esetén a résztávolság 1, azonos mássalhangzó esetén 0.5, azonos magánhangzó esetén 0.1. A távolságot a résztávolságok összege adja. Ebből le kell vonni még annyiszor 0.1-et, ahány karakterrel különbözik a két szó hossza.

   Írjon **programot**, amely a **billentyűzetről** beolvas egy **szót**, majd az állomány szavai közül a képernyőre írja azt az **öt** szót, amely a legközelebb van a beolvasott szóhoz! A szavak mellett szerepeljen a távolság is!

2. Írjon egy olyan **függvényt**, amely egy **paraméterként** megadott, **valósakat** tartalmazó kétdimenziós tömb azon **sorának indexét** adja meg (egy ilyen van), amelyben az elemek **összege a legkisebb**!

3. Írjon egy olyan **eljárást**, amely egy **paraméterként** megadott, **egészeket** tartalmazó vektor elemeit úgy rendezi át, hogy elöl álljanak a negatívak, aztán a pozitívak, aztán a nulla értékűek! Az elemek az egymáshoz viszonyított relatív sorrendjüket tartsák meg!

4. Írjon egy **programot**, amely billentyűzetről beolvas egy **egészeket** tartalmazó vektort, majd eldönti, hogy van-e olyan eleme, amelynek értéke **megegyezik** az elemek **átlagával**! Az ilyen elem(ek) indexét írassa ki a képernyőre, vagy ha nincs ilyen elem, akkor egy ilyen értelmű szöveg jelenjen meg!

5. Írjon **programot**, amely egy szöveges állomány azon sorait, amelyek megegyeznek a fordítottjukkal, egy másik szöveges állományba másolja át!

2.18. 2000. január 12., levelező tagozat

1. Írjon egy olyan **függvényt**, amely egy **paraméterként** megadott, **egészeket** tartalmazó kétdimenziós tömb azon **oszlopának** az **indexét** adja meg (csak egyetlen ilyen van), amelyben a **legtöbb pozitív** elem szerepel!

2. Írjon egy olyan **programot**, amely a billentyűzetről **magyar szavakat** olvas be a +++ végjelig, majd egy szövegállományba kiírja a szavakat ábécé sorrendbe rendezve!

3. Írjon **eljárást**, amely egy **paraméterként** megadott szöveges állomány **legrövidebb** sorait (több ilyen is lehet) a képernyőre írja!

4. Írjon egy **eljárást**, amely **paraméterként** egy olyan egydimenziós tömböt kap, amelynek elemei **angol** szavakat tartalmaznak! Az eljárás írja egy **globális** (már létező) szövegállomány **végére** azokat a szavakat, amelyekben a magánhangzók száma megegyezik a mássalhangzók számával!

5. Írjon egy olyan **függvényt**, amely egy **paraméterként** megadott, **különböző egészeket** tartalmazó egydimenziós tömb **második** legkisebb elemét határozza meg!

2.19. 2000. május 10., levelező tagozat

1. Írjon egy olyan **függvényt**, amely egy **paraméterként** megkapott, **egészeket** tartalmazó **kétdimenziós** tömb azon **sorának indexét** adja vissza, amely sorösszeg **nulla** (pontosan egy ilyen sor van)!

2. Írjon **eljárást**, amely egy **paraméterként** megkapott, **karaktereket** tartalmazó **egydimenziós** tömbben meghatározza azon **rész kezdő-** és **végindexét**, amelyben **azonos** karakterek vannak! Több ilyen rész esetén válassza ki azt, amelyben a karakterek száma **maximális**!

3. Írjon **programot**, amely a **billentyűzetről angol** szavakat olvas be a $*$ végjelig, majd **képernyőre** írja azokat a szavakat, amelyekben a **magánhangzók** száma **több**, mint a **mássalhangzók** száma!

4. Adva van egy **szöveges állomány**. Írjon **programot**, amely végigolvassa az állományt, és átmásolja egy **másik** szöveges állományba azokat a sorokat, amelyekben valahol **két szóköz** áll egymás mellett!

5. Írjon egy **függvényt**, amely egy **paraméterként** megkapott, **egészeket** tartalmazó **négyzetes mátrix** esetén a következő értékkel tér vissza:

   1, ha a mátrix szimmetrikus,
   2, ha a mátrix alsó háromszög mátrix,
   3, ha a mátrix felső háromszög mátrix.

2.20. 2000. május 20., nappali tagozat

1. Írjon egy **függvényt**, amely paraméterként megkapott **magyar** szóról eldönti, hogy az **milyen hangrendű**! A visszatérési érték:

   1, ha magas,
   0, ha mély,
   −1, ha vegyes.

2. Adva van egy olyan **szöveges állomány**, amelyben a sorok hossza maximum 80 karakter. Írjon egy olyan **programot**, amely minden sort szóközök segítségével 80 hosszúságúra egészít ki úgy, hogy a szóközöket a szövegelemek között **egyenletesen** osztja szét (sorkizárás)! Az új sorokat helyezze el egy másik szöveges állományban!

3. Adott egy csak **bináris** operátorokat tartalmazó **szabályos** kifejezés fájának **postorder** bejárásával kapott sorozat. Az operátorokat és az operandusokat **egy szóköz** választja el. A sorozatot billentyűzetről kapjuk. Írjon **programot**, amely képernyőre írja az adott kifejezés **preorder** alakját!

4. Írjon egy olyan **eljárást**, amely egy paraméterként megkapott maximum négyjegyű pozitív egész szám értékét **betűvel** a képernyőre írja! (Például **615** esetén **hatszáztizenöt**öt.)

5. Adva van egy teljesen zárójelezett szabályos kifejezés. Írjon egy olyan **függvényt**, amely meghatározza a kifejezés fájának magasságát!

6. Írjon egy olyan **függvényt**, amely egy sztringről eldönti, hogy az egy szabályos C-beli felsorolásos típusú definíciót tartalmaz-e!

2.21. 2000. augusztus 21., esti tagozat

1. Írjon **programot**, amely egy C forrásprogram első utasításaként megadott **nevesített konstans** definíciót figyelembe véve a forrásszövegben a nevet mindenütt helyettesíti a konstanssal!

2. Írjon egy **logikai függvényt**, amely egy paraméterként megkapott **magyar** szóról eldönti, hogy magánhangzóra vagy mássalhangzóra végződik-e!

3. Írjon egy **programot**, amely egy szöveges állomány soraiból eltávolítja a fölösleges (egymás mellett álló) szóközöket!

4. Adott egy csak **bináris** operátorokat tartalmazó **szabályos** kifejezés fájának **postorder** bejárásával kapott sorozat. Az operátorokat és az operandusokat **egy szóköz** választja el. A sorozatot billentyűzetről kapjuk. Írjon **programot**, amely képernyőre írja az adott kifejezés **preorder** alakját!

5. A felvételi után adott a felvételizők eredményeit tartalmazó **egyirányban láncolt lista**, amely elemei a nevet, értesítési címet és az elért pontszámot tartalmazzák. Az adatelemek rendezettek **pontszám** szerint **csökkenőleg**. Adott a biztos felvétel ponthatára és a fellebbezési lehetőség alsó ponthatára. Írjon **eljárást**, amely megadja a fellebbezésre jogosultak adatait, és azt névsor szerint rendezve elhelyezi egy másik egyirányban láncolt listában!

6. Írjon egy **függvényt**, amely **egészeket** tartalmazó **kétdimenziós** tömb azon **oszlopainak** (több ilyen is lehet) az **indexét** adja meg, amelyekben a pozitív elemek száma nagyobb a negatívakénál!

2.22. 2000. december 1., nappali tagozat

1. Adva van egy olyan **szöveges állomány**, amelynek sorai **magyar szavakat** tartalmaznak (legalább egyet és legfeljebb nyolcat), minden szó után **egyetlen szóköz** áll. Írjon **programot**, amely képernyőre írja azokat a **sorokat** (több ilyen is lehet), amelyekben a **legkevesebb** szó van!

2. Írjon egy olyan **logikai függvényt**, amely **igaz** értéket ad, ha a paraméterként megadott, **valós** értékeket tartalmazó egydimenziós tömb **utolsó** elemének értéke megegyezik az elemek átlagának értékével, egyébként pedig **hamis** értékkel tér vissza.

3. Írjon **programot**, amely egy **szöveges állomány** azon sorait, amelyekben a legtöbb **azonos** karakter van, egy másik szöveges állományba írja át!

4. Írjon egy olyan **függvényt**, amely egy paraméterként megkapott, **egészeket** tartalmazó kétdimenziós tömb azon elemeinek **darabszámát** adja meg, ahol az elemek értéke megegyezik az adott elem indexeinek szorzatával (az indexek típusa egész)!

5. Írjon **eljárást**, amely egy első paraméterként megkapott, **sztringeket** tartalmazó egydimenziós tömb minden elemében a második paraméterként megkapott **karakterrel** megegyező karaktereket átírja a harmadik paraméterként megkapott **karakterre**!

2.23. 2001. január 3., levelező tagozat

1. Adva van egy szöveges állomány, amelynek minden sora **vesszővel elválasztott angol szavakat** tartalmaz. A sorokban mindig van legalább **két** szó. Írjon **programot**, amely egy másik szöveges állományba másolja át azokat a sorokat, amelyeknek **utolsó** szava **magánhangzóval** kezdődik!

2. Írjon egy olyan **függvényt**, amely a paraméterként megkapott, **egészeket** tartalmazó **kétdimenziós** tömb azon **oszlopának** az **indexét** adja meg, amely oszlopban a legtöbb **0** elem van! Feltételezzük, hogy csak **egyetlen** ilyen oszlop létezik.

3. Írjon olyan **eljárást**, amely egy paraméterként megkapott **sztringet** megfordít!

4. Adva van egy olyan szöveges állomány, amely egy olyan Pascal (C) programot tartalmaz, amelyik csak **főprogramból** áll. Írjon **függvényt**, amely megadja, hogy a programban hány **változót** deklaráltak!

5. Írjon egy olyan **programot**, amely billentyűzetről **sztringeket** olvas mindaddig, amíg egy üres sztringet nem adunk meg, majd ezekből a sztringekből **hármasával** összeállít egy-egy **sort**, és elhelyezi azt egy **új** szöveges állományban!

2.24. 2001. április 28., nappali tagozat

1. Írjon **programot**, amely egy **létező szöveges állományból**, amelynek soraiban **egyetlen szóközzel elválasztott angol** szavak vannak, egy másik, **még nem létező szöveges állományba** írja át a **magánhangzóra végződő** szavakat! **A sorok száma ne változzon!**

2. Írjon egy **függvényt**, amely egy paraméterként megkapott, **egészeket** tartalmazó **egydimenziós tömbben** megszámolja azon elemek számát, amelyek egy szintén **paraméterként megadott** értéknél **nagyobbak**!

3. Írjon egy **eljárást**, amely a **billentyűzetről beolvas** egy **magyar mondatot**, és a benne szereplő **szavak kezdőbetűjét nagybetűsre** írja át, majd a képernyőre írja a mondatot!

4. Írjon egy **eljárást**, amely egy **paraméterként megkapott sztringben** megkeresi az összes **sorszámot** (pl. **1.**, **2.**, **5.**), és azokat lecseréli a **betűzött** alakjukra (**első**, **második**, **ötödik**)!

5. Írjon egy **eljárást**, amely egy **paraméterként megkapott**, **angol** szavakat tartalmazó egydimenziós tömb elemeit rendezi nagyság szerint **csökkenő** sorrendbe a **végük** szerint!

2.25. 2001. május 14., nappali tagozat

1. Az Elektronikus Tanulmányi Osztály két szöveges állományt használ a nyilvántartásban. Az egyik állomány **soronként** tartalmazza a **tantárgykódot 5** karakteren és a **tantárgy megnevezését maximum 35** karakteren. Az állomány a tantárgykódokat **növekvő** sorrendben tartalmazza. A másik állomány **soronként** tartalmazza a **hallgató nevét pontosan 25** karakteren, a **szakot** és **évfolyamot 4** karakteren és a hallgató által az adott félévben felvett **maximum 7** tantárgy **kódját** és **jegyét**. Írjon **programot**, amely egy szöveges állomány **soraiba** írja azon tantárgyak **nevét**, amelyeket az adott félévben senki sem vett fel! Tudjuk, hogy **77** tantárgy van.

2. Írjon egy **eljárást**, amely egy paraméterként megkapott **sztringben** megkeresi az összes **sorszámot** (pl. **351.**), és azokat lecseréli a betűzött alakjukra (pl. **háromszázötvenegyedik**)! A sorszámok **350** és **399** közé esnek.

3. Írjon egy **logikai függvényt**, amely egy paraméterként megadott, **egészeket** tartalmazó **kétdimenziós** tömb esetén **igaz** értéket ad, ha van a tömbben legalább két olyan **oszlop**, amelynek **átlaga** megegyezik!

4. Adva van egy szöveges állomány, amely egy olyan szabályos **C függvényt** tartalmaz, amelynek **3** formális paramétere van. Írjon **programot**, amely **képernyőre** írja, hogy az egyes formális paraméterekre a függvény mely **soraiban** történik hivatkozás! Feltehetjük, hogy minden sorban egy utasítás áll.

5. Írjon egy **függvényt**, amely paraméterként egy **kifejezés fáját** kapja meg, és visszaadja azt a **sztringet**, amely a kifejezés **teljesen zárójelezett infix** alakját tartalmazza!

2.26. 2001. augusztus 21., esti tagozat

1. Írjon egy **függvényt**, amely egy paraméterként megkapott, **egészeket** tartalmazó **kétdimenziós** tömb esetén megadja azon **oszlopok** számát, amelyekben egy szintén paraméterként megadott értéknél csak **nagyobb** értékek szerepelnek! (1.36.)

2. Írjon egy **programot**, amely egy szöveges állomány sorait úgy írja át egy másik szöveges állományba, hogy megkeresi a **leghosszabb** sort, és az összes többi sort az **elején** kiegészíti annyi **szóközzel**, hogy minden sor hossza azonos legyen!

3. Adva van egy szöveges állomány, amely egy olyan szabályos **C függvényt** tartalmaz, amelynek **egy (1)** formális paramétere van. Írjon egy **logikai függvényt**, amely akkor ad igaz értéket, ha a függvény **rekurzív**! (1.69.)

4. Írjon egy **eljárást**, amely egy paraméterként megkapott, **karaktereket** tartalmazó **bináris fában** a kisbetűket nagybetűre cseréli! (1.48.)

5. Adva van két szöveges állomány. Az egyikben a **tantárgykódok** (**5** karakteren) és a tantárgyak **nevei** (maximum **35** karakteren) vannak. A másikban a hallgatók **neve** (pontosan **25** karakteren) és az általa felvett maximum **12** tantárgy **kódja** szerepel. Írjon **programot**, amely egy szöveges állományba írja azon **tantárgy(ak) nevét**, amelyet a **legtöbben** vettek fel!

2.27. 2001. október 26., levelező tagozat

1. Írjon **programot**, amely egy **magyar** szavakat tartalmazó **szöveges állomány** szavait ábécé sorrendben írja át egy **másik** állományba! (1.66.)

2. Írjon **függvényt**, amely egy paraméterként kapott, **egészeket** tartalmazó **kétdimenziós** tömb azon oszlopának **indexét** adja vissza, amelyben a **legkevesebb pozitív elem** van! (1.35.)

3. Adott egy **szöveges** állomány, amelyben **magyar** szavak vannak, minden szó után egy **szóköz** áll. Írjon **eljárást**, amely képernyőre írja azon sorokat (több ilyen is lehet), amelyekben a **legkevesebb szó** van! (1.60.)

4. Írjon **eljárást**, amely egy paraméterként megkapott, **karaktereket** tartalmazó **egydimenziós** tömbben meghatározza azon rész **kezdő-** és **végindexét**, amelyben **azonos karakterek** vannak! Több ilyen esetén válassza ki azt, amelyben a karakterek száma **maximális**! (1.11.)

5. Írjon **programot**, amely **angol** szavakat kér be **billentyűzetről** *** végjelig, és kiírja egy **szöveges** állományba közülük azokat, amelyek tartalmazzák a `b`, `c`, `x`, `y` karaktereket! (1.57.)

2.28. 2001. november 30., levelező tagozat

1. Írjon egy **függvényt**, amely egy paraméterként megkapott **sztringben szóközzel** felülírja a **nem betű** karaktereket és visszaadja az új sztringet! (1.24.)

2. Írjon **eljárást**, amely paraméterként megkap **két szöveges állomány nevet** és **egy sztringet**, majd az első állomány azon sorait, melyeknek a **vége** azonos a sztringgel, átírja a másik állományba! Az első állomány létezik, a másodikat most kell létrehozni. (1.63.)

3. Írjon egy **programot**, amely a **billentyűzetről** beolvas egy pozitív egész számot (értéke maximum **110** lehet), majd a billentyűzetről beolvas ennyi darab **valós** számot és közülük **képernyőre** írja azokat, amelyek értékének a beolvasott számok **átlagától** való eltérése az átlag felénél nagyobb! (1.9.)

4. Írjon **eljárást**, amely paraméterként megkapott, **tetszőleges méretű**, egészeket tartalmazó kvadratikus mátrixot **tükröz** a **mellékátlójára**! (1.32.)

5. Írjon **eljárást**, amely egy paraméterként kapott **tetszőleges méretű**, egészeket tartalmazó egydimenziós tömbben meghatározza a legnagyobb összegű résztömb **kezdő-** és **végindexét** két output paraméterében! (1.13.)

2.29. 2001. december 17., nappali tagozat

1. Írjon **eljárást**, amely egy paraméterként kapott, **sztringeket** tartalmazó **egydimenziós** tömb minden elemét az **elején** és a **végén egyenletesen elosztott** szóközökkel kiegészíti olyan **hosszúságúra**, mint a **leghosszabb** elem hossza! Az eredeti tömb nem módosulhat! (1.21.)

2. Írjon **függvényt**, amely egy **szintaktikusan helyes C függvényt** tartalmazó **szöveges** állományt dolgoz fel úgy, hogy megszámolja a benne használt **különböző tömböket**!

3. Írjon **programot**, amely **billentyűzetről** beolvas egy **szabályos, teljesen zárójelezett C kifejezést**, amely operandusként csak **konstansokat** és **változókat** tartalmaz, és a **képernyőre** írja azt a **részkifejezést**, amelyet **először** kell kiértékelni! (1.52.)

4. Írjon **eljárást**, amely paraméterként megkap egy **bitmátrixot** és egy **bitvektort**, majd a bitmátrix minden oszlopa és a bitvektor között **kizáró vagy** műveletet végez! Az eredeti bitmátrixra a továbbiakban nincs szükség. (1.33.)

5. Írjon **logikai függvényt**, amely egy paraméterként megkapott, **egészeket** tartalmazó, **szigorú értelemben vett bináris fáról** eldönti, hogy **tökéletesen kiegyensúlyozott-e**! (1.47.)

2.30. 2001. december 18., levelező tagozat

1. Írjon egy **eljárást**, amely egy paraméterként megkapott, **tetszőleges méretű**, egészeket tartalmazó kvadratikus mátrix **főátlójában** elhelyezi **soronként** a főátló fölötti elemek **összegét**! (1.45.)

2. Írjon egy **függvényt**, amelynek első paramétere egy **sztring**, második paramétere egy **pozitív egész szám**, és a függvény adja vissza azt a sztringet, amely az eredetiből úgy keletkezik, hogy azt az elején és a végén kiegészítjük **szóközökkel** (egyenletesen elosztva azokat) úgy, hogy az új sztring hossza a második paraméter értéke legyen! (1.29.)

3. Írjon **programot**, amely egy szöveges állományban elhelyezett, **szintaktikailag helyes** Pascal (C) forrásprogram szövegét úgy másolja át egy másik szöveges állományba, hogy közben kihagyja belőle a **megjegyzéseket**! (1.67.)

4. Írjon egy **eljárást**, amely a paraméterként megkapott, **tetszőleges méretű**, **sztringeket** tartalmazó, egydimenziós tömböt a sztringek **hosszának csökkenő** sorrendjébe teszi! Azonos hosszak esetén a sztringek sorrendje az eredeti sorrend legyen! (1.15.)

5. Írjon **programot**, amely a billentyűzetről pozitív egész számokat olvas mindaddig, míg 0-t nem adunk. A program válassza ki a számok közül a **legkisebbeket** és a **legnagyobbakat** (lehetnek azonos értékűek is, de legfeljebb 3-3), írja azok értékét a **képernyőre**, és adja meg, hogy **hányadikként** olvastuk be őket! (1.4.)

2.31. 2002. január 2., levelező tagozat

1. Írjon **logikai függvényt**, amely egy paraméterként megadott, sztringeket tartalmazó négyzetes mátrixról eldönti, hogy **szimmetrikus-e**! (1.31.)

2. Írjon **eljárást**, amely egy paraméterként megkapott **bitmátrix** minden sora és egy szintén paraméterként kapott **bitvektor** között elvégzi a **vagy** műveletet!

3. Írjon **eljárást**, amely egy paraméterként megkapott, angol szavakat tartalmazó egydimenziós tömbben meghatározza és **képernyőre** írja a szavak **gyakoriságát**! (1.17.)

4. Írjon **programot**, amely billentyűzetről egyenként, elsőre nem meghatározható darabszámú **pozitív egész** értéket olvas be mindaddig, amíg **0** értéket nem kap! A program minden érték beolvasása után határozza meg az **addig beolvasott** értékek **átlagát**! Ha az új érték az eddigi átlag **kétszeresénél** nagyobb, akkor írja képernyőre az eddig beolvasott értékek **darabszámát** és az **új** értéket, majd a hátralévő értékeket másolja át egy **szöveges állományba**, különben viszont a beolvasás végén írja képernyőre a **darabszámot** és az **átlagot**!

5. Írjon **eljárást**, amely egy paraméterként megadott, **sztringeket** tartalmazó egydimenziós tömböt **elemeinek hossza** szerint **csökkenő** sorrendbe rendez! (1.14.)

2.32. 2002. január 2., nappali tagozat

1. Adva van egy olyan **szöveges állomány**, amely sorai **egyetlen szóközzel** elválasztott angol szavakat tartalmaznak. Írjon **programot**, amely meghatározza és **képernyőre** írja a szövegben előforduló szavak **gyakoriságát**! (1.62.)

2. Írjon **programot**, amely **billentyűzetről** beolvas egy olyan **teljesen zárójelezett kifejezést**, mely csak a $+$, $-$, $*$, / operátorokat és olyan **változó** operandusokat tartalmaz, amelyek nevében csak **betű** szerepel! Tudjuk, hogy a kifejezésben **zárójelhiba** van. A program írja képernyőre a kifejezést, és jelölje meg a hiba helyét! (1.55.)

3. Írjon **eljárást**, amely egy paraméterként megkapott, **egészeket** tartalmazó **keresőfából** kitöröl egy szintén paraméterként megadott **értéket**! Az eljárás írjon megfelelő hibaüzenetet a képernyőre, ha a törlés valamilyen okból nem hajtható végre! (1.50.)

4. Írjon **logikai függvényt**, amely egy paraméterként megkapott, **sztringeket** tartalmazó négyzetes mátrixról eldönti, hogy **szimmetrikus-e**! (1.31.)

5. Írjon **függvényt**, amely egy paraméterként megkapott, **szintaktikusan helyes C függvényt** tartalmazó **szöveges** állományt dolgoz fel úgy, hogy meghatározza a függvény által tartalmazott **blokkok** legnagyobb skatulyázási mélységét!

2.33. 2002. április 20., levelező tagozat

1. Írjon **programot**, amely **billentyűzetről** pozitív valós számokat olvas be mindaddig, amíg **nullát** nem adunk (tudjuk, hogy maximum **100** szám lehet). A program írja egy **most létrehozott** szöveges állományba azokat a beolvasott számokat, amelyeknek a **számok átlagától való eltérése** nagyobb, mint az **átlag fele**! (1.10.)

2. Írjon **logikai függvényt**, amely egy paraméterként megkapott **sztringről** eldönti, hogy az **palindróma-e**! (1.22.)

3. Írjon **függvényt**, amely egy paraméterként megkapott **sztringben** az egymás mellett álló **szóközök** közül csak egyet hagy meg, és visszatér az **új** sztringgel! (1.28.)

4. Írjon **eljárást**, amely egy paraméterként megkapott, **egészeket** tartalmazó **kétdimenziós tömb** oszlopait úgy rendezi át, hogy az első sor elemei nagyság szerint **csökkenő sorrendben** legyenek! Feltehetjük, hogy az első sor elemei különbözőek. (1.38.)

5. Írjon **eljárást**, amely egy paraméterként megkapott, **sztringeket** tartalmazó **egydimenziós tömb** minden elemét **azonos hosszúságúra** (a maximális hosszúságú elem hosszúságára) hozza úgy, hogy a sztringek **elejére** megfelelő számú **szóközt** szúr be! (1.19.)

2.34. 2002. május 13., nappali tagozat

1. Írjon **függvényt**, amely egy egydimenziós, sztringeket tartalmazó tömböt kap **paraméterként**, és annak minden elemét **csonkítja** a legrövidebb elem hosszára, és visszaadja az **új** tömböt! (1.20.)

2. Adva van egy **táblázat**, amelynek kulcsa egész, érték része sztring típusú és maximum 500 elem fér el benne. Írjon **függvényt**, amely **paraméterként** megkapja a táblázatot, az aktuális elemszámot és egy egészet. A függvény **bináris kereséssel** keresse meg az adott egész által meghatározott kulcsú elemet és adja vissza annak **érték részét**! Ha nincs ilyen elem, a függvény az **üres sztringgel** térjen vissza! (1.8.)

3. Írjon **eljárást**, amely egy **paraméterként** megadott kétdimenziós, egészeket tartalmazó tömb azon **oszlopát** határozza meg, amelyben benne van az egész tömb **legnagyobb** eleme (csak egy ilyen van)! (1.44.)

4. Adva van egy **szövegállomány**, amely egy olyan **C főprogramot** tartalmaz, amelyben van **felsorolásos típus** deklarálva (több is lehet). Írjon **programot**, amely a főprogram végrehajtható utasításaiban minden felsorolásos típusú literált **felülír** annak értékével, és az új főprogramot elhelyezi egy másik **szöveges** állományban!

5. Írjon **programot**, amely billentyűzetről megkap egy szabályos **prefix** alakú kifejezést. A program írja **képernyőre** az **elsőnek** kiértékelentő részkifejezést **infix** alakban! A kifejezés csak a $+$, $-$, $*$, / kétoperandusú operátorokat és operandusként olyan változókat tartalmaz, amelyek neve egyetlen karakterből áll. (1.53.)

2.35. 2002. május 23., nappali tagozat

1. Adva van egy **szöveges állomány**, amely egymástól egy szóközzel elválasztott **különböző angol** szavakat tartalmaz. Írjon **programot**, amely képernyőre írja azokat a szavakat (ezekből akármennyi lehet), amelyekben a **legtöbb magánhangzó** van. (1.59.)

2. Írjon logikai **függvényt**, amely egy paraméterként kapott **sztringről** eldönti, hogy van-e benne olyan **részsztring**, amely **pontosan 4 azonos** karakterből áll! (1.27.)

3. Írjon **eljárást**, amely egy paraméterként megkapott, **egészeket** tartalmazó, **tetszőleges méretű kétdimenziós** tömb minden olyan elemének a **0** értéket adja, amelynek a **tömb elemeinek átlagától való eltérése az átlag felénél nagyobb**! Az eredeti tömböt változatlanul kell hagyni. (1.46.)

4. Adva van egy **tetszőleges egészeket** tartalmazó **bináris** fa. Írjon **függvényt**, amely paraméterként megkapja a bináris fa **gyökérmutatóját**, a fában elhelyezett értékekből felépít egy **keresőfát**, és visszaadja annak **gyökérmutatóját**! Az új fa elemeinek szerkezete: **kulcs** (a különböző egész értékek), **gyakoriság** (az eredeti fában az adott kulcs hányszor fordult elő). (1.49.)

5. Adva van egy **szöveges** állomány, amely egy **C főprogramot** tartalmaz. Írjon **programot**, amely az összes **azonosítót nagybetűsre** írja át, és az új főprogramot elhelyezi egy **másik szöveges** állományba! (1.68.)

2.36. 2002. augusztus 21., esti tagozat

1. Írjon **logikai függvényt**, amely egy paraméterként megkapott **angol** szó esetén igazzal tér vissza, ha a szóban **nincs egymás mellett két mássalhangzó**! (1.25.)

2. Írjon **programot**, amely egy **létező szöveges állomány** minden sorát **80** karakter hosszúságúra **egészíti ki szóközökkel**, ha rövidebbek a sorok 80 karakternél, és **csonkítja** a végén a sorokat, ha azok hosszabbak 80 karakternél! Az új sorokat egy **új** szöveges állományba kell írni. (1.64.)

3. Írjon **eljárást**, amely egy paraméterként megkapott, **egészeket** tartalmazó **kétdimenziós** tömbben meghatározza azon oszlopok **indexét** (akárhány ilyen lehet), amelyekben a negatív elemek száma **legalább kétszerese** a nulla értékű elemek számának! A tömb mérete **tetszőleges**.(1.43.)

4. Írjon **függvényt**, amely paraméterként egy olyan sztringet kap, amely egy szabályos, **teljesen zárójelezett infix** kifejezést tartalmaz, és meghatározza a kifejezés fájának **magasságát**! A kifejezésben csak a $+$, $-$, $*$, / bináris operátorok és maximum 3 jegyű egész szám operandusok fordulnak elő. (1.51.)

5. Adva van egy **szöveges állomány**, amely soraiban egymástól **egyetlen szóközzel** elválasztott **magyar** szavak állnak. Írjon **eljárást**, amely meghatározza az állományban előforduló szavak **gyakoriságát**! Feltételezhetjük, hogy maximum 200 különböző szó fordul elő. (1.61.)

2.37. 2002. november 29., levelező tagozat

1. Írjon egy **logikai függvényt**, amely egy paraméterként megkapott **sztring** esetén **igaz** értékkel tér vissza, ha a sztringben a betűk (angol ábécé!) száma **nagyobb**, mint a nem-betű karakterek száma, és **hamissal** tér vissza egyébként! (1.23.)

2. Írjon **eljárást**, amely paraméterként megkap egy **tetszőleges** méretű, **egészeket** tartalmazó **egydimenziós** tömböt. Az eljárás határozza meg a tömbben lévő **pozitív**, **negatív** és **nulla** elemek darabszámát! **Az eljárás nem írhat a képernyőre!** (1.6.)

3. Írjon **programot**, amely billentyűzetről egyenként beolvas egész értékeket addig, amíg a -1 értéket nem kapja. A program írja **képernyőre** a beolvasott számok azon szekvenciáját, amely a **leghosszabb szigorúan monoton csökkenő** sorozatot alkotja! (1.12.)

4. Adva van egy **szöveges** állomány, amely egy szintaktikailag helyes, **valós** visszatérési értékkel rendelkező C (Pascal) **függvényt** tartalmaz. Írjon **programot**, amely eldönti, hogy a függvény **rekurzív-e**! Az eredmény jelenjen meg a **képernyőn**! (1.69.)

5. Írjon **függvényt**, amely **tetszőleges** méretű, **valós** értékeket tartalmazó **kétdimenziós** tömböt kap paraméterként. A függvény határozza meg azon **oszlop indexét**, amelyben van olyan elem, amelynek az értéke megegyezik az oszlop elemeinek **átlagával**! Ha több ilyen oszlop is van, akkor a **legnagyobb** indexértéket adja vissza! (1.34.)

2.38. 2003. január 6., esti tagozat

1. Írjon **eljárást**, amely egy paraméterként megkapott, tetszőleges méretű, valósakat tartalmazó kétdimenziós tömb **sorait** úgy rendezi át, hogy az **utolsó oszlop** értékei **csökkenő** sorrendben legyenek! (1.39.)

2. Írjon **logikai függvényt**, amely egy paraméterként megkapott sztringnél igaz értéket ad vissza, ha a sztringben van olyan **4 elemű** részsztring, amely **legalább háromszor** ismétlődik. (1.26.)

3. Adott egy csak a `+`, `-`, `/`, `*` **bináris** operátorokat tartalmazó **szabályos** kifejezés fájának **post-order** bejárásával kapott sorozat. Az operátorokat és az operandusokat **egy szóköz** választja el egymástól. A sorozatot billentyűzetről kapjuk. Írjon **programot**, amely képernyőre írja a kifejezés **prefix** alakját! (1.54.)

4. Írjon **eljárást**, amely egy paraméterként megkapott, tetszőleges méretű, egészeket tartalmazó kétdimenziós tömb **oszlopátlagai** közül meghatározza a **legnagyobbat** (több ilyen is lehet)! **Az eljárás nem írhat képernyőre és állományba!** (1.40.)

5. Adva van egy szöveges állomány, amely egy szintaktikailag helyes, **egész** visszatérési értékkel rendelkező C **függvényt** tartalmaz. Írjon **programot**, amely meghatározza, hogy a függvény **rekurzív-e**! Az eredmény a **képernyőn** jelenjen meg! (1.69.)

2.39. 2003. január 6., levelező tagozat

1. Írjon **eljárást**, amely egy paraméterként megkapott, tetszőleges méretű, valósakat tartalmazó kétdimenziós tömb **sorait** úgy rendezi át, hogy az **utolsó oszlop** értékei **csökkenő** sorrendben legyenek! (1.39.)

2. Írjon **logikai függvényt**, amely egy paraméterként megkapott sztringnél igaz értéket ad vissza, ha a sztring **tükörszimmetrikus** (pl. `görög`, `kosarasok`)! (1.22.)

3. Írjon **programot**, amely billentyűzetről **látható karaktereket** olvas mindaddig, amíg a `@` karaktert meg nem kapja. A program határozza meg és írja **képernyőre** a beolvasott **különböző** karaktereket és azok **gyakoriságát**! (1.1.)

4. Írjon **eljárást**, amely egy paraméterként megkapott, tetszőleges méretű, egészeket tartalmazó kétdimenziós tömb **oszlopátlagai** közül meghatározza a **legnagyobbat** (több ilyen is lehet)! **Az eljárás nem írhat képernyőre és állományba!** (1.40.)

5. Adva van egy szöveges állomány, amely egy szintaktikailag helyes, **egész** visszatérési értékkel rendelkező C (Pascal) **függvényt** tartalmaz. Írjon **programot**, amely meghatározza, hogy a függvény **rekurzív-e**! Az eredmény a **képernyőn** jelenjen meg! (1.69.)

2.40. 2003. május 17., levelező tagozat

1. Adva van egy **szöveges** állomány, benne egy szintaktikailag szabályos C (Pascal) **függvény**. Írjon **függvényt**, amely megadja a függvény **végrehajtható utasításainak** darabszámát.

2. Írjon **eljárást**, amely paraméterként megkap egy **karaktereket** tartalmazó, **tetszőleges méretű egydimenziós** tömböt, és a tömb **nem betű** karaktereit kicseréli **szóközre**. (1.5.)

3. Írjon **függvényt**, amely paraméterként megkap egy **tetszőleges méretű**, **egészeket** tartalmazó **kvadratikus mátrixot**, és visszaadja a **főátló** maximális és minimális elemét. **A képernyőre nem írunk!** (1.41.)

4. Írjon **eljárást**, amely paraméterként megkap egy **azonos hosszúságú sztringeket** tartalmazó, **tetszőleges méretű egydimenziós** tömböt, továbbá egy **karaktert** és egy **pozitív egész** számot, és a **képernyőre írja** azokat a tömbelemeket, amelyekben a karakter pontosan az adott számszor fordul elő. **A szélsőséges eseteket vizsgálni kell**! (1.16.)

5. Írjon **programot**, amely a **billentyűzetről angol szavakat** olvas mindaddig, amíg **üres sztringet** nem kap. A program írja egy **szöveges állományba** azokat a szavakat, amelyekben egymás mellett van **legalább három mássalhangzó**. (1.58.)

2.41. 2003. május 23., nappali tagozat

1. Adva van egy **szöveges állomány** és benne egy **szintaktikailag szabályos C függvény**. Írjon **eljárást**, amely egy **új** szöveges állományba átmásolja a **formázott** forrásszöveget. A formázás: minden utasítás külön sorban legyen; a blokk kezdő és záró kapcsos zárójele külön sorban legyen; a beágyazott programegységek a tartalmazó programegység utasításaitól 3 karakternyi hellyel beljebb kezdődjenek; a szöveg balra zárt legyen.

2. Írjon **programot**, amely **billentyűzetről** megkap egy olyan **teljesen zárójelezett** kifejezést, amely csak a $-$ és a $+$ **egy-** és **kétoperandusú** operátorokat, operandusként pedig olyan **C-beli változókat** tartalmaz, melyek neve **maximum két** karakterből áll. Ellenőrizze le, hogy a kifejezés **szabályos-e**. A képernyőre írjon értelemszerű hibaüzeneteket. (1.56.)

3. Írjon **függvényt**, amely paraméterként megkap egy olyan **tetszőleges** méretű **egydimenziós** tömböt, amely **angol** szavakat tartalmaz. A függvény **visszatérési értéke** adja meg a szavak **gyakoriságát**. (1.18.)

4. Írjon **eljárást**, amely paraméterként megkap egy **tetszőleges** méretű, **valósakat** tartalmazó **kétdimenziós** tömböt, és előállít egy olyan **egydimenziós** tömböt, amely a **sorok átlagát** tartalmazza. **Az eljárás a képernyőre nem írhat!** (1.37.)

5. Írjon **programot**, amely nullától különböző **egész** értékeket olvas be a **billentyűzetről** a **0** végjelig. A program határozza meg és írja **képernyőre** azt a **három** értéket, amelynek **átlaga maximális**. (1.3.)

2.42. 2003. május 28., nappali tagozat

1. Adva van egy **szöveges állomány** és benne egy **szintaktikailag szabályos C függvény**. Írjon **eljárást**, amely egy **új** szöveges állományba átmásolja a **formázott** forrásszöveget. A formázás: minden utasítás külön sorban legyen; a blokk kezdő és záró kapcsos zárójele külön sorban legyen; a beágyazott programegységek a tartalmazó programegység utasításaitól 3 karakternyi hellyel beljebb kezdődjenek; a szöveg balra zárt legyen.

2. Írjon **függvényt**, amely **paraméterként** megkap egy olyan **teljesen zárójelezett** kifejezést, amely csak a $-$ és a $+$ **egy-** és **kétoperandusú** operátorokat, operandusként pedig **C-beli literálokat** tartalmaz. A kifejezés szintaktikailag **helyes**. A függvény építse fel a kifejezés **fáját**.

3. Írjon **függvényt**, amely paraméterként megkap egy olyan **tetszőleges** méretű **egydimenziós** tömböt, amely **angol** szavakat tartalmaz. A függvény **visszatérési értéke** adja meg azon szavakat, melyekben **legalább 3 mássalhangzó** szerepel egymás mellett.

4. Írjon **eljárást**, amely paraméterként megkap egy **tetszőleges** méretű, **valósakat** tartalmazó **kétdimenziós** tömböt és egy **pozitív valós** értéket, majd meghatározza azon **sorok indexét** (ezekből akármennyi lehet), amelyekben a főátlóbeli elemek sorátlagtól való **eltérése** a második paraméter értékétől **kisebb**. **Az eljárás a képernyőre nem írhat!**

5. Írjon **programot**, amely nullától különböző **egész** értékeket olvas be a **billentyűzetről** a **0** végjelig. A program határozza meg és írja **képernyőre** azt a **leghosszabb** számsorozatot (amennyiben több ilyen is van, akkor mindet), amelyben a számok **előjele azonos**.

2.43. 2003. augusztus 25., esti tagozat

1. Adva van egy szöveges állomány, amely egy **szintaktikailag helyes**, **karakteres** visszatérési értékkel rendelkező **C függvényt** és egy **szintaktikailag helyes főprogramot** tartalmaz. Írjon **programot**, amely megállapítja, hogy a főprogram **szabályosan** hívta-e meg a függvényt!

2. Írjon **eljárást**, amely paraméterként megkap egy **sztringeket** tartalmazó, **tetszőleges** méretű **egydimenziós** tömböt, és ebből a **hívás helyén felhasználható** új egydimenziós tömböt állít elő, amely azokat a **sztringeket** tartalmazza, amelyekben a **betűk** (angol ábécét tételezve fel) száma megegyezik a **nem betű karakterek** darabszámával.

3. Írjon **függvényt**, amely egy paraméterként megkapott, **tetszőleges méretű**, **valósakat** tartalmazó **kétdimenziós** tömb **minimális** és **maximális** átlagú **oszlopainak** (ezekből csak egy-egy lehet) **indexét** határozza meg visszatérési értékként.

4. Írjon **programot**, amely **nullától** különböző **egész** értékeket olvas be a billentyűzetről a **0** végjelig. A program határozza meg és írja **képernyőre** a beolvasott különböző egész **értékeket** és azok **gyakoriságát**.

5. Írjon **függvényt**, amely meghatározza a paramétereként megkapott, **tetszőleges méretű** (de legalább négyelemű), **egészeket** tartalmazó **egydimenziós** tömb **harmadik legkisebb** elemének az **indexét** (amennyiben több ilyen elem is van, akkor a legnagyobb indexet kell visszaadni).

2.44. 2003. december 5., levelező tagozat

1. Írjon **programot**, amely egy **létező** szöveges állomány maximum 100 karakrterből álló sorainak mindegyikét **100 hosszúságúra** egészíti ki, a sorok elején és végén egyenletesen elosztott módon **szóközöket** szúrva be („**középre igazítás**"). Az új sorokat egy **most létrehozott** szöveges állományba kell elhelyezni.

2. Írjon **logikai függvényt**, amely akkor tér vissza igaz értékkel, ha a paraméterként kapott **sztring szabályos C azonosító**.

3. Írjon **eljárást**, amely paraméterként megkap egy **tetszőleges méretű**, **valósakat** tartalmazó **kétdimenziós** tömböt, és meghatározza azt a **vektort**, amely az **oszlopok átlagát** tartalmazza. **Az eljárásban nem lehet semmilyen I/O művelet!**

4. Írjon **függvényt**, amely paraméterként megkap egy **tetszőleges** méretű **négyzetes bitmátrixot** és egy megfelelő méretű **bitvektort**, és visszatér azzal a **bitvektorral**, amely a mátrix főátlója és a vektor elemei között képzett **kizáró vagy** eredményeként keletkezik.

5. Írjon **programot**, amely **billentyűzetről** egész értékeket olvas be a **0** végjelig. A program írja **képernyőre** azokat az értékeket, amelyek **előjele** eltér a **megelőző** érték előjelétől. Például 1 2 3 4 0 esetén nincs ilyen érték, 1 −1 2 −2 −5 0 esetén kiírandó −1 2 −2.

2.45. 2004. január 9., levelező tagozat

1. Írjon **programot**, amely egy **létező** szöveges állomány sorainak mindegyikét **100 hosszúságúra** egészíti ki a sorok végén **szóközöket** szúrva be, ha rövidebb, illetve **elhagyva** a fölösleges karaktereket, ha hosszabb. Az új sorokat egy **most létrehozott** szöveges állományba kell elhelyezni.

2. Írjon **logikai függvényt**, amely akkor tér vissza igaz értékkel, ha a paraméterként kapott **sztring szabványos C egész literál**.

3. Írjon **függvényt**, amely paraméterként megkap egy **tetszőleges méretű**, **valósakat** tartalmazó **kétdimenziós** tömböt, és visszatérési értékként meghatározza a **sorok átlagának minimumát** és az **oszlopok átlagának maximumát**.

4. Írjon **eljárást**, amely paraméterként megkap egy **tetszőleges** méretű, **sztringeket** tartalmazó **vektort**, és előállít egy olyan **vektort**, amelymek elemei rendre a paraméterként kapott vektor elemeinek annyiadik **karakterét** tartalmazzák, amennyi az adott elem **indexe**, illetve a `@` karaktert, ha nem létezik ilyen elem. Egy sztring karaktereit 0-tól sorszámozzuk. **Az eljárásban nem lehet semmilyen I/O művelet!**

5. Írjon **programot**, amely **billentyűzetről** egész értékeket olvas be a **0** végjelig. A program írja **képernyőre** azokat az értékeket, amelyek **megegyeznek** az **előző két** érték **összegével**.

2.46. 2004. május 15., levelező tagozat

1. Írjon **programot**, amely **billentyűzetről nempozitív** valós számokat olvas be mindaddig, amíg **1.1**-et nem adunk. A program írja egy most létrehozott **szöveges** állományba a beolvasott számok közül a **10** legnagyobbat, vagy ha tíznél kevesebb számot olvastunk be, akkor mindet!

2. Írjon **logikai függvényt**, amely egy paraméterként megkapott **sztringről** eldönti, hogy abban egy **másik paraméterként** megkapott **karakter legalább háromszor** előfordul-e!

3. Írjon **függvényt**, amely paraméterként megkapott **sztringben**, amely **szóközökkel** elválasztott tetszőleges karaktersorozatokat tartalmaz, a szóközök és szóközcsoportok helyett **vesszőt** helyez el, és **visszatér az új sztringgel**! Például `"b n mm. 76("` esetén az eredmény `"b,n,mm.,76(".`

4. Írjon **eljárást**, amely egy **paraméterként** megkapott, **egészeket** tartalmazó, tetszőleges méretű, **kétdimenziós** tömb **oszlopai** közül meghatározza azt az oszlopot (mint **egydimenziós** tömböt), amelyben a legtöbb **páros szám** van!

5. Írjon **eljárást**, amely egy **paraméterként** megkapott, **sztringeket** tartalmazó, tetszőleges méretű **egydimenziós** tömb minden elemét egy **másik paraméter** által megadott hosszúságúra **csonkítja**, ha az adott elem **hosszabb**, illetve az **elején szóközökkel** kiegészítve adott hosszúságúra hozza, ha az adott elem **rövidebb**! **Az eredeti tömb nem változhat meg!**

2.47. 2004. június 2., levelező tagozat

1. Írjon **programot**, amely **billentyűzetről egész** számokat olvas be **egyenként** mindaddig, amíg **0**-t nem adunk. A program írja egy most létrehozott **szöveges** állományba **soronként** a beolvasott számok közül azokat az **egymás melletti számhármasokat**, amelyek lehetnek egy **háromszög** oldalhosszai!

2. Írjon **logikai függvényt**, amely egy paraméterként megkapott **sztringről** eldönti, hogy abban **két másik**, szintén **paraméterként** megkapott **karakter** előfordul-e úgy, hogy közöttük pontosan egy szóköz van!

3. Írjon **függvényt**, amely egy **paraméterként** megkapott **tetszőleges sztringben** megszámolja, hogy az **utolsó** karakter hányszor fordul elő!

4. Írjon **eljárást**, amely egy **paraméterként** megkapott, **egészeket** tartalmazó, tetszőleges méretű, **kétdimenziós** tömb oszlopaiból kiválogatja a **legkisebb abszolút értékű** elemeket (oszloponként csak egy-egy ilyen van) és azokat elhelyezi egy **egydimenziós** tömbben!

5. Írjon **függvényt**, amely egy **paraméterként** megkapott, **valósakat** tartalmazó, tetszőleges méretű, **kétdimenziós** tömb **legnagyobb** elemének az **indexeivel** tér vissza!

2.48. 2004. december 3., levelező tagozat

1. Írjon **programot**, amely **billentyűzetről egész** számokat olvas be **egyenként** mindaddig, amíg **0**-t nem adunk. A program írja egy most létrehozott **szöveges** állományba **soronként** a beolvasott számok közül azokat az **egymás melletti legalább 3 elemből álló számsorozatokat**, amelyek számtani sorozatot alkotnak!

2. Írjon **logikai függvényt**, amely egy paraméterként megkapott **sztringről** eldönti, hogy abban **két másik**, szintén **paraméterként** megkapott **karakter** a **megadás sorrendjében** előfordul-e úgy, hogy közöttük **legfeljebb** két másik karakter van!

3. Írjon **eljárást**, amely egy paraméterként megkapott sztringet alakít át. A sztringben olyan mondatok vannak, amelyek végén **.**, **?** vagy **!** áll. A mondatok első karakterét alakítsa nagybetűssé, a többit kisbetűssé!

4. Írjon **eljárást**, amely egy **paraméterként** megkapott, **valósakat** tartalmazó, tetszőleges méretű, **kétdimenziós** tömb minden elemét egészre kerekíti!

5. Írjon **függvényt**, amely egy **paraméterként** megkapott, **egészeket** tartalmazó, tetszőleges méretű **kétdimenziós** tömb **legnagyobb abszolút értékű** elemének (egy ilyen van) az **indexeivel** tér vissza!

2.49. 2004. december 20., levelező tagozat

1. Írjon **programot**, amely **billentyűzetről egész** számokat olvas be **egyenként** mindaddig, amíg **0**-t nem adunk. A program írja egy most létrehozott **szöveges** állományba **soronként** a beolvasott számok közül azokat az **egymás melletti legalább 3 elemből álló** számsorozatokat, amelyek **szigorúan monoton** sorozatot alkotnak!

2. Írjon **logikai függvényt**, amely egy paraméterként megkapott **sztringről** eldönti, hogy abban **két másik**, szintén **paraméterként** megkapott **karakter** előfordul-e egymás mellett úgy, hogy az elsőből pontosan kettő, a másodikból pedig egy szerepel!

3. Írjon **eljárást**, amely egy paraméterként megkapott sztringet úgy alakít át, hogy a benne lévő, **vesszővel** elválasztott szavakat **megfordítja**!

4. Írjon **eljárást**, amely egy **paraméterként** megkapott, **valósakat** tartalmazó, tetszőleges méretű, **kétdimenziós** tömb negatív elemeiből egy vektort készít!

5. Írjon **függvényt**, amely egy **paraméterként** megkapott, **egészeket** tartalmazó, tetszőleges méretű **kétdimenziós** tömb **legnagyobb abszolút értékű** elemének (egy ilyen van) az **indexeivel** tér vissza!

2.50. 2005. május 14., levelező tagozat

1. Írjon **programot**, amely **billentyűzetről egész** számokat olvas be **egyenként** mindaddig, amíg **0**-t nem adunk. A program írja egy most létrehozott **szöveges** állományba **soronként** a beolvasott számok közül azokat az **egymás melletti számhármasokat**, amelyek közül a középső a másik kettő szorzata!

2. Írjon **logikai függvényt**, amely egy paraméterként megkapott **sztringről** eldönti, hogy abban **két másik**, szintén **paraméterként** megkapott **karakter együttesen** egy negyedik paraméter által meghatározott értéknél **többször** fordul elő!

3. Írjon **eljárást**, amely egy paraméterként megkapott sztringet alakít át. A sztringben olyan mondatok vannak, amelyek végén **.**, **?** vagy **!** áll, bennük minden szót **valahány szóköz** választ el. A szavak **első** és **utolsó** karakterét alakítsa **nagybetűssé**, a többit **kisbetűssé**! Az eredeti sztring nem változhat meg!

4. Írjon **eljárást**, amely egy **paraméterként** megkapott, **valósakat** tartalmazó, **tetszőleges** méretű, **kétdimenziós** tömb **oszlopainak átlagát** határozza meg!

5. Írjon **függvényt**, amely egy **paraméterként** megkapott, **sztringeket** tartalmazó, **tetszőleges** méretű **egydimenziós** tömb **leghosszabb** elemeinek (akárhány ilyen lehet) az **indexeivel** tér vissza!

2.51. 2005. június 14., levelező tagozat

1. Írjon **programot**, amely **billentyűzetről negatív egész** számokat olvas be **egyenként** mindaddig, amíg **1**-et nem adunk. A program írja egy most létrehozott **szöveges** állományba **soronként** a beolvasott számok közül azokat a **részsorozatokat**, amelyek **legalább kételeműek** és **szigorúan monoton növekvőek**!

2. Írjon **logikai függvényt**, amely **igennel** tér vissza, ha a paramétereként megkapott **angol szóban** minden betű különbözik!

3. Írjon **függvényt**, amely egy **paraméterként** megkapott **tetszőleges sztringben** megszámolja a karakterek gyakoriságát!

4. Írjon **eljárást**, amely egy **paraméterként** megkapott, valósakat tartalmazó, tetszőleges méretű, **kétdimenziós** tömbből kiválogatja azokat az elemeket, amelyek egy második paraméterként megadott értéknél kevésbé térnek el a tömb összes elemének átlagától, és azokat elhelyezi egy **egydimenziós** tömbben!

5. Írjon **függvényt**, amely egy **paraméterként** megkapott, **egészeket** tartalmazó, tetszőleges méretű **kétdimenziós** tömbben meghatározza, hogy melyik **sorban** (több ilyen is lehet, egy biztos van) fordul elő egy második paraméterként megadott elem, és visszaadja a sorok **indexeit**!

2.52. 2005. november 25., levelező tagozat

1. Írjon **programot**, amely **billentyűzetről karaktereket** olvas be mindaddig, amíg **szóközt** nem adunk. A program írja egy most létrehozott **szöveges** állományba a beolvasott karakterek közül azokat, amelyek csak egyszer fordultak elő!

2. Írjon **logikai függvényt**, amely egy paraméterként megkapott **sztringről** eldönti, hogy abban a magánhangzók és mássalhangzók száma **azonos-e**!

3. Írjon **függvényt**, amely egy **paraméterként** megkapott **sztringben**, amely **szóközökkel** elválasztott tetszőleges karaktersorozatokat tartalmaz, az esetleges szóközcsoportok helyett egyetlen **szóközt** helyez el, és **visszatér az új sztringgel**!

4. Írjon **eljárást**, amely egy **paraméterként** megkapott, **egészeket** tartalmazó, **tetszőleges** méretű, **kétdimenziós** tömbben meghatározza azon oszlop(ok) indexeit, amely(ek)ben a legtöbb **5-re végződő** szám van!

5. Írjon **eljárást**, amely egy **paraméterként** megkapott, **sztringeket** tartalmazó, **tetszőleges** méretű, **egydimenziós** tömb minden eleméből **eltávolítja** azon karakter minden előfordulását, amely karaktert egy **másik paraméter** ad meg! **Az eredeti tömb nem változhat meg!**

2.53. 2005. december 22., levelező tagozat

1. Írjon **programot**, amely **billentyűzetről egész** számokat olvas be **egyenként** mindaddig, amíg **0**-t nem adunk. A program írja egy most létrehozott **szöveges** állományba **soronként** a beolvasott számok közül azokat az **egymás melletti legalább 3 elemből álló** számsorozatokat, amelyek **szigorúan monoton csökkenő** sorozatot alkotnak!

2. Írjon **logikai függvényt**, amely egy paraméterként megkapott **egész számról** eldönti, hogy annak **két másik**, szintén **paraméterként** megkapott egész szám **osztója-e**!

3. Írjon **eljárást**, amely egy paraméterként megkapott sztringet **megfordít**!

4. Írjon **eljárást**, amely egy **paraméterként** megkapott, egészeket tartalmazó, tetszőleges méretű, **kétdimenziós** tömb **nem nulla** elemeiből egy **vektort** készít!

5. Írjon **függvényt**, amely egy **paraméterként** megkapott, **egészeket** tartalmazó, tetszőleges méretű, **kétdimenziós** tömb **legnagyobb abszolút értékű** elemének (egy ilyen van) az **indexeivel** tér vissza!

2.54. 2006. november 24., levelező tagozat

1. Írjon **eljárást**, amely paraméterként megkap egy **egészeket** tartalmazó **egydimenziós tömböt**, amely egy **nemcsökkenő** sorozatot tartalmaz, és előállít egy **új tömböt**, amely a megkapott tömb elemeit tartalmazza, de minden számból **csak egy** előfordulást! **A hívónak az eredeti és az új tömböt is látnia kell!**

2. Írjon **logikai függvényt**, amely igaz értékkel tér vissza, ha a paraméterként megkapott két **pozitív egész szám** tízes számrendszerben felírva **nem tartalmaz azonos számjegyeket**, különben pedig hamissal! Pl.: 194 és 57622 esetén igaz, 194 és 57621 esetén hamis a függvény visszatérési értéke.

3. Írjon **függvényt**, amely egy paraméterként megkapott **valósakat** tartalmazó **tetszőleges méretű négyzetes mátrix** mellékátlójában lévő elemekből képzett **új vektorral** tér vissza!

4. Írjon **eljárást**, amely paraméterként két **állománynevet** kap! Az első állományban ábécérendben **soronként** egy-egy szó található, a másodikat most kell létrehozni. Írja át az első állományból a második állományba a szavakat **hosszuk szerint növekvő sorrendben**, szintén soronként egyet-egyet! Az állományban legfeljebb 1000 szó található, és minden szó legfeljebb 15 karakterből áll.

5. Írjon **programot**, amely a **billentyűzetről újsorokkal** elválasztott **sztringeket** olvas, amíg a "vége" szót nem kapja! A program írja a **képernyőre** a beolvasott sztringek közül azokat, amelyeknek az **első és utolsó karaktere megegyezik**! A sztringek legfeljebb 100 karakter hosszúak.

2.55. 2006. december 17., levelező tagozat

1. Írjon **programot**, amely **képernyőre** írja a `bemenet.txt` nevű **szöveges állomány** sorait, **soronként megfordítva**!

2. Írjon **függvényt**, amely paraméterként megkap egy **sztringet**, amely egy kizárólag **decimális számjegyekből** álló pozitív **számot** tartalmaz! A függvény adja vissza **egész számként** a legkisebb olyan **számrendszernek** az alapszámát, amelyben a megkapott szám értelmezhető!

3. Írjon **függvényt**, amely paraméterként megkap egy legalább két karakter hosszúságú **sztringet**, és visszaad egy **új sztringet**, amely az eredeti sztringből úgy áll elő, hogy **elhagyja** annak **első** és **utolsó karakterét**! Az eredeti sztring nem változhat!

4. Írjon **logikai függvényt**, amely paraméterként megkap egy tetszőleges méretű, **egészeket** tartalmazó egydimenziós **tömböt**, és igazzal tér vissza, ha a tömb **minden páros indexű eleme páros**, különben pedig hamissal!

5. Írjon **eljárást**, amely a **billentyűzetről egész számokat** olvas, amíg a paraméterként megkapott számot nem kapja, majd **képernyőre** írja a beolvasott **pozitív** számok **átlagát** és **darabszámát**! Figyeljen arra, hogy egész számok átlaga valós!

2.56. 2007. május 19., levelező tagozat

1. Írjon **logikai függvényt**, amely paraméterként megkap egy **pozitív egész számot**, és igazzal tér vissza, ha a szám **tökéletes**, különben pedig hamissal! Egy szám tökéletes, ha a nála kisebb osztóinak az összege maga a szám (például $6 = 1 + 2 + 3$).

2. Írjon **eljárást**, amely paraméterként megkap egy tetszőleges méretű, **valósakat** tartalmazó **mátrixot**, és előállít egy **vektort**, amely az egyes **oszlopok legnagyobb elemeit** tartalmazza! **A hívónak látnia kell az új vektort!**

3. Írjon **függvényt**, amely paraméterként megkap egy **pozitív egész számot**, és visszaadja, hogy (tízes számrendszerben felírva) **hány különböző számjegyet** tartalmaz!

4. Írjon **függvényt**, amely paraméterként megkap egy **állománynevet**, és az állomány azon **sorainak számával** tér vissza, amelyek **hossza maximális**!

5. Írjon **programot**, amely a billentyűzetről egész számokat olvas, amíg 0-t nem kap, majd képernyőre írja a beolvasott **páratlan számok összegét**!

2.57. 2007. június 2., levelező tagozat

1. Írjon **logikai függvényt**, amely paraméterként megkap két **pozitív egész számot**, és igazzal tér vissza, ha azok **barátságos** számpárt alkotnak, különben pedig hamissal! Két szám barátságos, ha az egyik szám nála kisebb osztóinak az összege a másik szám, és fordítva. Például a $(220, 284)$ barátságos számpár, mert $284 = 1 + 2 + 4 + 5 + 10 + 11 + 20 + 22 + 44 + 55 + 110$ és $220 = 1 + 2 + 4 + 71 + 142$.

2. Írjon **eljárást**, amely paraméterként megkap egy tetszőleges méretű, **valósakat** tartalmazó **mátrixot**, valamint egy egész számot, amely a mátrix egy érvényes sorindexe, és előállít egy **vektort**, amely a mátrix **adott** indexű **sorában** lévő elemeket tartalmazza! **A hívónak látnia kell az új vektort!**

3. Írjon **logikai függvényt**, amely paraméterként megkap egy **sztringet**, és igazzal tér vissza, ha a sztring egy **érvényes rendszámot** tartalmaz! A rendszám csak akkor számít érvényesnek, ha **pontosan hét karakterből** áll, az első három karaktere **angol nagybetű**, a negyedik karaktere egy **kötőjel**, az utolsó három karaktere pedig decimális **számjegy** karakter.

4. Írjon **függvényt**, amely paraméterként megkap egy **állománynevet**, és visszaadja, hogy az állomány **hány üres sort** tartalmaz!

5. Írjon **programot**, amely a billentyűzetről **egész számokat** olvas, amíg 0-t nem kap, majd képernyőre írja az „OK” sztringet, ha a beolvasott számok sorozatában a **sorozat elején** csak **páratlan** számok, **azt követően** pedig csak **páros** számok szerepelnek!

2.58. 2007. augusztus 26., levelező tagozat

1. Írjon **eljárást**, amely paraméterként megkap egy **sztringeket** tartalmazó **egydimenziós tömböt**, valamint egy **karaktert**, és előállít egy **új tömböt**, amelynek az egyes elemei megadják, hogy a megkapott tömb megfelelő (azonos indexű) elemében **hányszor fordul elő** a megkapott karakter! **A hívónak látnia kell az új tömböt!**

2. Írjon **függvényt**, amely paraméterként megkap egy **állománynevet**, és visszaadja, hogy **hány szó** fordul elő az állomány **legtöbb szót** tartalmazó **sorában**! Szavaknak tekintjük az egy vagy több szóközzel határolt összefüggő karaktersorozatokat. Az állomány sorai legfeljebb 1000 karakter hosszúak.

3. Írjon **logikai függvényt**, amely igaz értékkel tér vissza, ha a paraméterként megkapott **egészeket** tartalmazó **tetszőleges méretű mátrix** csak **különböző elemeket** tartalmaz, különben pedig hamissal!

4. Írjon **logikai függvényt**, amely **igaz értékkel** tér vissza, ha a paraméterként megkapott **sztring** szabályos **magyar rendszám**, azaz **6 karakter** hosszúságú, csak **nagybetűket és számjegyeket** tartalmaz, **nagybetűvel kezdődik, számjeggyel végződik**, és a benne szereplő **nagybetűk megelőzik a számjegyeket**! Minden egyéb esetben a függvény hamis értéket adjon vissza!

5. Írjon **programot**, amely a **billentyűzetről egész számokat** olvas, amíg 0-t nem kap, majd képernyőre írja az „OK” sztringet, ha a beolvasott számok között **ugyanannyi páros** szám van, mint **negatív**!

2.59. 2007. december 1., levelező tagozat

1. Írjon **logikai függvényt**, amely paraméterként megkap egy **karaktereket** tartalmazó 3×3-as **mátrixot**, és igazat ad vissza, ha a mátrix a **tic-tac-toe** játék egy **érvényes állását** tartalmazza, különben pedig hamisat. Az állás csak akkor érvényes, ha az alábbi feltételek mindegyike teljesül:

   - a mátrix minden eleme a szóköz, az ’X’ vagy az ’O’ karakterek egyike;
   - az ’X’-ek száma vagy megegyezik az ’O’-k számával, vagy eggyel nagyobb nála;
   - nincs mindkét játékosnak hármasa.

2. Írjon **függvényt**, amely paraméterként megkap egy **állománynevet**, és visszaadja, hogy **milyen hosszú** az állomány **leghosszabb szava**! Szavaknak tekintjük az egy vagy több fehér karakterrel (szóközzel, tabulátorral vagy újsorral) határolt összefüggő karaktersorozatokat. Feltehetjük, hogy az állomány nem tartalmaz 100 karakternél hosszabb szót.

3. Írjon **logikai függvényt**, amely paraméterként **két sztringet** kap, és igaz értékkel tér vissza, ha az **első sztringben** a **második sztring minden karaktere** előfordul, különben pedig hamissal!

4. Írjon **eljárást**, amely paraméterként megkap egy tetszőleges méretű, **egészeket** tartalmazó egydimenziós **tömböt**, és előállít egy **vektort**, amely a kapott tömb azon elemeit tartalmazza, amelyek legalább az egyik szomszédjukkal **relatív prímpárt** alkotnak!

5. Írjon **programot**, amely a `szamok.txt` nevű **szöveges állományban** tárolt **valós számok** közül az **átlag alattiakat** kiírja a **képernyőre**! Az állomány legfeljebb 10000 számot tartalmaz, soronként egyet-egyet.