Programozás alapjai 1. (BMEVIEEA100)

Gyakorlat anyaga az 5. oktatási héten (3-4. gyakorlat)

A múlt heti anyag elején látható ütemezés szerint ezen a héten a pénteki gyakorlaton a múlt heti téma menjen (operátorok), a hétfői és szerdai csoportokban pedig az állapotgépekről és többdimenziós tömbökről legyen szó!

A 6. héten (jövő hét) kerül sor az első kis ZH-ra, ami teszt, erről később.

Pénteki csoportok

A múlt heti gyakorlat anyaga az irányadó, aki még nem nyomtatta ki, annak érdemes a http://www.eet.bme.hu/~pohl/cjegyzet.html oldalról letöltött változatot kinyomtatni, mert Izsó Tamásnak és Szandi Lajosnak köszönhetően fény derült egy hibára a „Logikai operátor mint kiértékelési pont” résznél, a bekezdés eleje így módosult:

Az első esetben az 'a++' értéke 'a' eredeti értéke, a 'c=a' részben viszont már fordítótól függ, hogy 'a' eredeti, vagy megnövelt értéke kerül c-be, és így az összegbe, csak az a biztos, hogy a ';' után 'a' már inkrementált.

Az óra első része ezúttal is a házi feladat ellenőrzése. Egyszerű megírandó programok voltak. Ezek közül válasszunk néhányat, és írassuk fel a megoldást a hallgatókkal! A múlt héten kértem, hogy a péntekiek adják fel előre az **„Azonosítók, operátorok”** részt a tesztek közül. Talán van olyan hallgató, aki foglalkozott vele, de a többség valószínűleg nem. A **fejezet példái, valamint a „Bevezetés” fejezet új példái jelen dokumentum végén találhatók. Ossza ki mindenki ezeket, és lehetőleg beszéljétek is végig a megoldásokat.** Az órai anyagot a tesztfeladatok kapcsán is el lehet mondani, de ennél talán jobb megoldás, ha még a teszt megbeszélése előtt gyorsan összefoglaljuk az operátorokat, hogy a füzetükben áttekinthetően legyen meg ez a rész.

Házi feladat: a Bevezetés és az Azonosítók, operátorok tesztkérdések ismétlése mellett az Adatbevitel billentyűzetről, kiírás képernyőre című fejezet tesztkérdései, valamint a Szente-Varga féle példatár 4. fejezetéből (Bitműveletek, bináris számok) a múlt heti anyaghoz mellékelt része. A Ciklusok fejezetet most nem kell feladni, lesz így is dolguk bőven.

Hétfői és szerdai csoportok

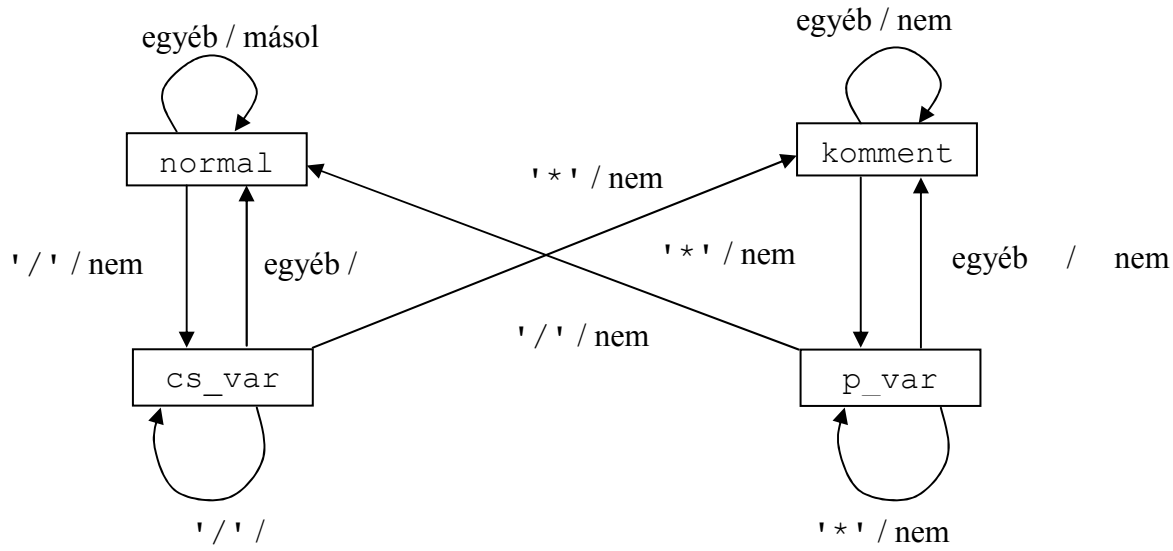
Az óra első felében ellenőrizzük a házi feladatot! A múlt héten kiosztott Szente-Varga Domonkos féle példák meg kell, hogy legyenek nekik, viszont a **tesztgyűjteményben szereplő Azonosítók, operátorok fejezet** még nem volt teljes, ezért nyomtatva még nem kapták meg a kérdéseket. Jelen dokumentum végén találjátok nyomtatható formában ezt a fejezetet, valamint a **Bevezetés fejezet azon példáit, melyek újak**. Ezeket a tesztsorokat mindenképp beszéljük át, hiszen ez a ZH anyaga!

Állapotgép

Konkrétan állapotgépekkel csak a gyakorlatokon találkoznak a hallgatók, előadáson csak említésre kerülnek az enum kapcsán, laboron sem kerül terítékre, ezért helyezzünk nagy súlyt arra, hogy megértsék!

Feladat: a szabványos bemeneten érkező szöveget úgy továbbítsuk a szabványos kimenetre, hogy eltávolítjuk belőle a /**/ stílusú megjegyzéseket.

A program állapotgépes modelljének állapotgráfja:




```
/* C forrasprogram komment mentesitese:
   A szabvanyos bemenetrol EOF-ig erkezo szoveget szuri.
   A szurt allomanyt a szabvanyos kimenetre irja.

   A programot allapotgepes modellel valositottuk meg.
*/

#include <stdio.h>
typedef enum {normal,komment,cs_var,p_var} allapotok; // allapotgepes modellhez

void main(void){
        int ch;
        allapotok allapot=normal;

        while((ch=getchar())!=EOF){
                switch (allapot){
                        case normal:    if(ch!='/')putchar(ch);else allapot=cs_var;break;
                        case cs_var:    if(ch=='*')allapot=komment;

        else{putchar('/');if(ch!='/'){putchar(ch);allapot=normal;}}
                                        break;
                        case komment:   if(ch=='*')allapot=p_var;break;
                        case p_var:     if(ch=='/')allapot=normal;else if(ch!='*')allapot=komment;
                                        break;
                }
        }
}
```

Többdimenziós tömbök

Néhány játék, ahol kétdimenziós tömbben tárolhatjuk a játék tábláját:

- Tetrisz
- Sakk
- Amőba
- Torpedó
- Labirintus => PacMan

Hogyan kezdjük el a játék írását? Határozzuk meg, hogy mekkora tábla kell, pl. a torpedó játékban legyen egy x irányban 20, y irányban 15 egységből álló mező.

Mit tárolunk? Jelezze 0, ha az adott mezőn víz van, 1-5, ha hajó, a szám jelzi, hogy mekkora méretű, ugyancsak egész számok jelzik, ha a másik játékos kilőtt egy torpedót, és az adott mezőn vizet/hajót talált. Tehát egész számokat tárolunk, azaz egész számokból álló tömbünk lesz.

```
int tenger[15][20], i;
for(i=0;i<15;i++)for(j=0;j<20;j++)tenger[i][j]=0; // kezdetben mindenhová tengert teszünk
... // beállítjuk, hogy hol vannak hajók
for(i=0;i<15;i++){// kiírjuk a tengert táblázatos formában
    for(j=0;j<20;j++)printf("%2d ",tenger[i][j]);
    printf("\n");
}
```

Szorgalmi házi feladat: akinek kedve van, írja meg teljesen a torpedó programot! Tegyen be hajókat, akár fix helyre, aztán egy ciklusban kérjen koordinátákat, és reagáljon megfelelően.

Példa négydimenziós (+1) tömbre: könyvtár:

```
const int X=100,Y=30,polc=5,hely=25;
char konyvtar[X][Y][polc][hely][100];
printf("Adja meg a bal szelso sor 10. oszlopában, a 2. polc 22. helyen"
       "levo konyv cimet!");
scanf("%s",konyvtar[0][9][1][21]);
```

Házi feladat

A következő oldal utáni három oldalon található a tesztgyűjtemény mai órára elkészítendő része, azaz a *Bevezetés* új feladatai, valamint az *Azonosítók, operátorok* fejezet, továbbá két új fejezet. A második tesztet a 9. héten írják, tehát két héttel az első teszt után, annak anyaga három kérdéscsoport, melyek közül az első kettő van itt rajta. Ebből egyet mindenki adjon fel, ha kímélni szeretnétek a hallgatókat, akkor az *Adatbevitel…* kérdéscsoportot eltehetitek egy héttel későbbre. A péntekiek mindenképp tegyenek így.

A negyedik lapon szereplő feladatokat a hétfő-szerdaiak adják föl, a péntekieknek ez a jövő hétre csúszik.

A feladatokat nem kötelező fénymásolva kiosztani, de ha nem okoz problémát, ajánlott. Aki nem tudja megoldani, az kérheti a hallgatóktól, hogy nyomtassák ki a tesztfeladatokat maguknak és úgy járjanak órára, a gyakorlópéldákat pedig elég a sorszámukkal megadni, ha felírjátok a példatár linkjét (http://www.eet.bme.hu/~szvdom/peldatar/), akkor a hallhatók meg fogják találni a feladatot.

ZH

A következő oldalon látható a zh mintája (aki akar, ilyet is oszthat). A sikeres megoldásokért + pont jár, a sikertelenekért -. Ponthatárok a lap alján. Az éles zh-nál az oktatók dolga lesz, hogy minimum 2 csoportnyi feladatot készítsen a teszthez mellékelt programmal. A zh első 6 feladata feleletválasztós, az utolsó kettő megoldandó, ez utóbbiak kisebb súllyal esnek a latba.

Ha olyan feladatot találtok, amelyik szerintetek rossz, azt jelezzétek, és még a zh előtt törlésre kerül!

Minta ZH (használható hozzá kétoldalas C összefoglaló és kézzel írott puska).

Név:		FV sikeres:		× 3 pont=	
Neptun kód:		FV sikertelen:		×–3 pont=	
Hallgató aláírása:		MF sikeres:		× 2 pont=	
		MF sikertelen:		×–2 pont=	
		FV=FeleletVálasztós, MF=Megoldandó Feladat		Össz:	

---

1. Mit mondhatunk a következő kódrészletről?

```
  int a,b; a=b=2; b=a+++++b;
```

[ ] A kifejezés egyértelmű: b = a++ + ++b, így a értéke 3 b értéke 5 lesz.
[ ] A mohó előfeldolgozó a következőképpen értelmezi: b = a++ ++ +b,
ami szintaktikai hiba, mert a ++ operátor csak balértékre alkalmazható.

---

2. Mi a következő kifejezés értéke? `!(1&&1||1&&0)`

[ ] Igaz
[ ] Hibás kifejezés
[ ] Hamis

---

3. Mi a kifejezés értéke az adott definíció mellett?
Definíció: `int a=3; int c='z';`
Kifejezés: `a||(c=getchar())`
[ ] Ez egy logikai kifejezés, csak 0 vagy 1 lehet, ráadásul logikai VAGY,
melyben az első elem rögtön logikai IGAZ, tehát ki sem értékelődik
a getchar, azaz a kifejezés értéke 1 lesz.
[ ] Nem montható meg, mert nem definiáltuk, mi van az inputon.
[ ] Szintkaktikai hiba, mert int változóba olvasunk a getchar függvénnyel.

---

4. Mennyi az `i+++i++` kifejezés értéke?
[ ] 2*i.
[ ] 2*i+1.
[ ] A fordítótól függ, mert a mellékhatások bekövetkezési ideje nem rögzített.

---

5. Melyik állítás igaz?
[ ] Minden szabványos C programban kell legyen egy és csak egy main függvény.
[ ] A main függvénnyel tisztelgünk az első C fordítót készítő előtt aki Maine
államban született.
[ ] Ha a szabványos C programban nincs main függvény, akkor a program végre-
hajtása az első függvény első utasításával kezdődik.

---

6. Mit ír ki az alábbi kódrészlet?

```
int a;
printf("%d", a);
```

[ ] 1
[ ] 0
[ ] szemetet

---

7. Mit lesz a maxplus10 változó értéke?

```
int x = 4, y = 29;
int maxplus10 = 10 + (x>y)?x:y;
```

---

8. Mennyi az értéke a következő kifejezésnek:

```
7 & 7 && 20
```

Pontozás: -2: 1, 3-8: 2, 9-14: 3, 15-18: 4, 19-22: 5

```
X ********************************************************************************
C Bevezetés - új kérdések
X ********************************************************************************
--------------------------------------------------------------------------------
20. Melyik állítás igaz az alábbiak közül?
[ ] Egy függvény deklarációjának és definíciójának mindig ugyanabban a file-ban
kell szerepelnie.
[ ] Egy függvény deklarációjának mindig meg kell előznie a függvény meghívását
egy file-on belül.
[ ] Egy függvény definíciójának mindig meg kell előznie a függvény meghívását
egy file-on belül.
--------------------------------------------------------------------------------
21. Mennyi lesz a MAX makró értéke?
#define MAX 4
#define MAX 5
[ ] 5, mert a legutolsó define az érvényes
[ ] Fordítási idejű hibát kapunk, mert először az #undef-et kell használni, ha
egy makró értékét megváltoztatjuk
[ ] 4
--------------------------------------------------------------------------------
22. Mit ír ki az alábbi kódrészlet?
int a;
printf("%d", a);
[ ] szemetet
[ ] 0
[ ] 1
--------------------------------------------------------------------------------
23. Mit ír ki az alábbi programrészlet?
printf("%d", 0247);

--------------------------------------------------------------------------------
24. Mit mondhatunk a következő kódrészletről?
 int p[2]={2,3}; int j=6;
 j=j/*p; /* j-t elosztjuk a p első elemével */
[ ] Nem használhatjuk a * operátort egy tömb neve előtt
[ ] A j változó nem változik, mert az első /*-gal kezdődik a megjegyzés
a mohó előfeldolgozó miatt
[ ] A megjegyzésnek megfelelően a j változó értéke 3 lesz (6/2)
--------------------------------------------------------------------------------
25. Vizsgálhatjuk-e a túlcsordulást a következő kódrészlettel?
 int i,j; ...
 if ((i+j)>INT_MAX)) ...
[ ] Nem, mert semmilyen egész művelet ideiglenesen sem lesz nagyobb, mint az
ábrázolható legnagyobb szám, ezért ez a feltétel akkor sem teljesül a
programban, ha matematikailag az összeg nagyobb, mint INT_MAX
[ ] Igen, ez így jó
[ ] Nem, mert nem INT_MAX a legnagyobb ábrázolható egész, hanem MAX_INT
--------------------------------------------------------------------------------
26. Ha egy double visszatérési típusú, egyetlen double paramétert váró függvényt
meg akarunk hívni a main függvényből, akkor a forrásállományban a main
függvény definíciója előtt:
[ ] a meghívandó függvénynek legalább a prototípus deklarációjának szerepelnie
kell (ill. egy header állományt kell #include-olni, amiben szerepel).
[ ] a meghívandó függvény deklarációjának és definíciójának is szerepelnie
kell (ill. egy header állományt kell #include-olni, amiben szerepel).
[ ] nem szükséges semmi, amennyiben a fordító valahol máshol megtalálja
a meghívandó függvényt.
--------------------------------------------------------------------------------
27. Mit ír ki a program
#define ot 1+4
#define tiz 1+9
printf("%d", ot*tiz );
[ ] 14
[ ] 38
[ ] 50
--------------------------------------------------------------------------------
28. Csak a megjegyzések használatából tudunk ideiglenesen eltávolítani
kódrészeket?
[ ] Nem lehet ideiglenesen eltávolítani kódrészeket.
[ ] Igen.
[ ] Nem, például a #if 0 ... #endif preprocesszor utasításokkal is lehet.

********************************************************************************
C Azonosítók, operátorok
********************************************************************************


--------------------------------------------------------------------------------
2. Az alábbi azonosítók különbözőnek számítanak-e?
krumpli, kRuMpLi
[ ] Az adott operációs rendszertől függ.
[ ] Igen.
[ ] Nem.
--------------------------------------------------------------------------------
3. Mely azonosítók hibásak az alábbiak közül?
A: Ag97KSrzd_sfa
B: Pici%Maci
C: Almas Pite
D: switch
E: ThisIsAVariable
[ ] A, B és E
[ ] A, B és D
[ ] B, C és D
--------------------------------------------------------------------------------
4. Mi a 24|29 kifejezés értéke?
--------------------------------------------------------------------------------
5. Mi a 20&13 kifejezés értéke?

--------------------------------------------------------------------------------
6. Mi a 26^30 kifejezés értéke?

--------------------------------------------------------------------------------
7. Mi a 26^30^26 kifejezés értéke?

--------------------------------------------------------------------------------
8. Mi a(z) !25 || 28 kifejezés kiértékelésének eredménye?

--------------------------------------------------------------------------------
9. Mennyi lesz 'b' változó értéke az utasítás lefutása után, ha 'a' változó
jelenlegi értéke 53, 'b' változó jelenlegi értéke pedig 21?
(a >= b || b++ < 5)

--------------------------------------------------------------------------------
10. Mi a(z) 7 ^ ~7 kifejezés kiértékelésének eredménye?

--------------------------------------------------------------------------------
11. Mennyi 1<<3+0?

--------------------------------------------------------------------------------
12. Mennyi sizeof(char) értéke?
[ ] 8, hiszen egy karakter egy bájtot foglal, ami nyolc bit, a sizeof pedig
definíció szerint a zárójelei közé tett típus vagy változó bitben kifejezett
méretét adja.
[ ] 1, hiszen a sizeof definíció szerint a zárójelei közé tett típus vagy
változó karakterméretben kifejezett méretét adja, egy karakter mérete pedig
egy karaktermeretnyi.
[ ] 8 vagy 16, hiszen a karakter bitben kifejezett mérete nincs definiálva a C
nyelvben, UNICODE karakterek esetén pedig 16 bites a karakter. A sizeof
definíció szerint a zárójelei közé tett típus vagy változó bitben kifejezett
méretét adja.
--------------------------------------------------------------------------------
13. Melyik a nagyobb, sizeof(unsigned) vagy sizeof(int)
[ ] sizeof(unsigned) nagyobb.
[ ] Egyforma méretűek.
[ ] Nem lehet megmondani, mert a C nyelv nem definiálja a típusok konkrét
méretét.
--------------------------------------------------------------------------------
14. Melyik a nagyobb, sizeof(int) vagy sizeof(float)
[ ] sizeof(float) nagyobb.
[ ] Egyforma méretűek.
[ ] Nem lehet megmondani, mert a C nyelv nem definiálja a típusok konkrét
méretét.
--------------------------------------------------------------------------------
15. Mit ír ki az alábbi függvény, ha szamol("3M5ewGwD12") módon hívjuk meg?
void szamol(const char * s){
    unsigned i,sum;
    for(i=sum=0;s[i];i++)sum+=(s[i]>='0'&&s[i]<='9')?s[i]-'0':0;
    printf("%u\n",sum);
}

--------------------------------------------------------------------------------
16. Helyes-e az alábbi változódefiníció?

int Harom_perecnel_kevesebb_a_2;
[ ] Nem.
[ ] Igen.
--------------------------------------------------------------------------------
17. Helyes-e az alábbi változódefiníció?

int 3_perecnel_kevesebb_a_ketto;
[ ] Nem.
[ ] Igen.
--------------------------------------------------------------------------------
18. Milyen értéket ad vissza az alábbi függvény, ha hívásakor a = 14, b = -63?
int f(int a, int b) {return a - b ? (a > b ? 1 : -1) : 0; }
[ ] 0, mert a - b nem egyenlő nullával.
[ ] 1, mert a > b.
[ ] 77, mert ennyi a - b;
--------------------------------------------------------------------------------
19. Melyik igaz? Az (a == 0 || b % a ) kifejezés
[ ] logikai igaz értéket ad, ha a nem osztója b-nek.
[ ] hibás, mert a == 0 esetén a VAGY kapcsolat második tagjában 0-val való osztás
történik.
--------------------------------------------------------------------------------
20. Melyik kódolási technika ajánlott az alábbiak közül?
A: int i=5; if (i!=0)printf("i értéke nem 0!\n");
B: int i=5; if (i)   printf("i értéke nem 0!\n");
[ ] Mindkettő.
[ ] Csak a B.
[ ] Csak az A.
--------------------------------------------------------------------------------
21. Mennyi lesz x értéke?
#define MAX(A,B) ((A)>(B)?(A):(B))
int a=5, b=7, x;
x = MAX(++a, ++b);
[ ] 9.
[ ] 8.
[ ] 7.
--------------------------------------------------------------------------------
22. Mit ír ki a következő kód: printf("%d", 2<3);
16? TRUE? 1?

--------------------------------------------------------------------------------
23. Melyik kifelyezés helyes?
a) (x++)+3
b) (3++)+x
c) x++++
[ ] a
[ ] b
[ ] c
--------------------------------------------------------------------------------
24. Mit ír ki az alábbi program?
void main(){ int a = 2, b =3, x; x = --a*(a+b); printf("x erteke: %d",x); }
[ ] semmit, mivel hibás a kód
[ ] x erteke: 4
[ ] x erteke: 9
--------------------------------------------------------------------------------
25. Mi az x értéke ha y = -2?
x = ((y + 2) < 0 ? (-(y + 3)) : (y + 2));

--------------------------------------------------------------------------------
26. Az int i = 3, j = 1, k = 2; kezdeti értékeket mi lesz az alábbi kifejezés
értéke?
k = -i+++j;
[ ] i = 4, j = 1, k = -2
[ ] i = -4, j = 1, k = -2
[ ] i = -3, j = 2, k = -1
--------------------------------------------------------------------------------
27. Mennyi 19 % 3 értéke?

--------------------------------------------------------------------------------
28. Mennyi lesz a=4,b=4,a+++b; kifejezés érétke? (a és b int típusú)

--------------------------------------------------------------------------------
29. Mennyi lesz az alábbi sor lefutása után k értéke?
int i = 17, j = 5, k = 0;
k += ++i - j++

--------------------------------------------------------------------------------
30. Mi a(z) (17*2)>>1 kifejezés kiértékelésének eredménye?

--------------------------------------------------------------------------------
31. Mi lesz x értéke az x = (19+1, (8+1, 25+1)) kifejezés kiértékelése után?

--------------------------------------------------------------------------------
32. Mit ír ki az alábbi programrész?
   int a = 0;
   if (a = 0) printf("0"); else printf("nem 0");
[ ] 0
[ ] nem 0
--------------------------------------------------------------------------------
33. Mit ír ki az alábbi programrészlet?
   int i = 2;
   i == 1 || i++;
   printf("%d", i);

--------------------------------------------------------------------------------
34. Helyes-e az alábbi változódefiníció?
int double=1;
[ ] Nem.
[ ] Igen.
--------------------------------------------------------------------------------
35. Mit jelent a következő sor:
int size, len(char []);
[ ] Hibás, mert a len változó típusa char[], a size pedig int
[ ] size változó definíciója és a len függvény prototípusa
[ ] Hibás, mert függvény prototípust nem lehet változó definícióval együtt
használni
--------------------------------------------------------------------------------
36. Mennyi az értéke a következő kifejezésnek:
26 & 26 && 14

--------------------------------------------------------------------------------
37. Mit ír ki a következő program:
int m=13, n=12;
if (n++ < (m>n ? --m: ++m)) printf("%d, %d\n", n, m);
else printf("%d", n);

--------------------------------------------------------------------------------
38. Mennyi az a=b kifejezés értéke?
[ ] Az a változó új értéke.
[ ] Az a változó régi értéke.
[ ] 1, mert az értékadás sikere logikai igaznak felel meg.
--------------------------------------------------------------------------------
39. Mennyi az i+++i++ kifejezés értéke?
[ ] 2*i.
[ ] 2*i+1.
[ ] A fordítótól függ, mert a mellékhatások bekövetkezési ideje nem rögzített.
--------------------------------------------------------------------------------
40. Mit mondhatunk a következő kódrészletről?
int i=2000;
printf("%d %d\n",!!++i,i);
[ ] Csak a következő kiírás a helyes: 1 2001, mert i-t előbb inkrementáljuk
[ ] A paraméterként szereplő kifejezések kiértékelési sorrendje nem definiált,
így nem tudhatjuk milyen értékkel hívódik meg a printf (1,2000 vagy 1,2001)
```

[ ] Az eredmény egyértelmű, mert ugyan nem tudható, hogy 2000 vagy 2001-nek
kell-e a logikai negáltját venni, de mindegy, hiszen az mindkettőre 0
és annak a negáltja biztos 1, tehát ami kiíródik: 1 2000
---
41. Mit mondhatunk a következő kódrészletről?
  int a,b; a=b=2; b=a---b;
[ ] A mohó előfeldolgozó a következőképpen értelmezi: b = a-- - b ;
ami teljesen meghatározott, a értéke 1, b értéke pedig 0 lesz.
[ ] Szintaktikai hiba.
[ ] Nem lehet eldönteni, hogy a-- -b vagy a- --b, tehát nem definiált.
---
42. Mit mondhatunk a következő kódrészletről?
  int a,b; a=b=2; b=a+++++b;
[ ] A kifejezés egyértelmű: b = a++ + ++b, így a értéke 3 b értéke 5 lesz.
[ ] A mohó előfeldolgozó a következőképpen értelmezi: b = a++ ++ +b,
ami szintaktikai hiba, mert a ++ operátor csak balértékre alkalmazható.
---
43. Mi a kifejezés értéke az adott definíció mellett?
Definíció: int a=3; int c='z';
Kifejezés: a||(c=getchar())
[ ] Nem montható meg, mert nem definiáltuk, mi van az inputon.
[ ] Ez egy logikai kifejezés, csak 0 vagy 1 lehet, ráadásul logikai VAGY,
melyben az első elem rögtön logikai IGAZ, tehát ki sem értékelődik
a getchar, azaz a kifejezés értéke 1 lesz.
[ ] Szintkaktikai hiba, mert int változóba olvasunk a getchar függvénnyel.
---
44. Mi a kifejezés értéke az adott definíció mellett?
Definíció: double d=39.9;
Kifejezés: ((int)d-3)/10
[ ] 3.69
[ ] 3
[ ] 3.9
---
45. Mit lesz a maxplus10 változó értéke?
int x = 7, y = 17;
int maxplus10 = 10 + (x>y)?x:y;

---
46. Mi lesz az average változó értéke?
int sum = x + (x + 1);
float average = sum / 2 * 1.0;
[ ] x
[ ] x + 1
[ ] x + 0.5
---
47. Az alábbiak közül melyik sorban szerepelnek az operátorok szigorúan csökkenő
precedencia-sorrendben?
[ ] ! - < ||
[ ] * ++ & >
[ ] + > && !
---
48. Mennyi 28 ^ 255 ?

---
49. Mit eredményez az alábbi programrészlet?

 int a=b=17;
 while(1=2){ a/=23; if(b=a) printf("X"); else break; }
[ ] Egy X kerül a képernyőre.
[ ] Nem fordul le.
[ ] Végtelen ciklusba kerül a program.
---
50. Ha egy változó értékét folyamatosan növeljük, akkor az
egy idő után negatív lesz.
[ ] Igaz
[ ] A változó típusától függ
[ ] Hamis
---
51. Mi a következő kifejezés értéke?   !(1&&1||1&&0)
[ ] Hibás kifejezés
[ ] Hamis
[ ] Igaz
---
52. Ha az x változó kezdeti értéke 29, mi a teljes x+=31,12+21 kifejezés
kiértékelésének eredménye?

---
53. Mely operátorok esetében van sequence point a kifejezés kiértékelése során?
[ ] Csak 3 esetben: a logikai ÉS és VAGY, valamint a vessző operátornál: && || ,
[ ] Kifejezések belsejében egyáltalán nincs, csak utasítások között.
[ ] Minden összetett kifejezésben az asszociativitásnak megfelelően.
---
54. Az alábbi változódefiníciók közül melyik hibás?
[ ] int nicelx;
[ ] int nicelx=0xface;
[ ] int 0xface,nicelx;
---
55. a=24; b=24 esetén mi lesz az (a++<++b || a++>2) kifejezés kiértékelése
után 'a' új értéke?

---
56. a=8; b=8 esetén mi lesz az (a++<b++ && 2 * a++ > --b + 28) kifejezés
kiértékelése után 'a' új értéke?

---
57. A ciklus végrehajtása után mennyi lesz az i változó értéke?
  int a[7] = {1,2,3,4,5,0,0};
  int i;
  for(i=0; a[i] & a[i+1]; i++) {}
[ ] A ciklus addig fut, amíg az aktuális és a rákövetkező tömbelem értéke 0,
tehát az i értéke 5 lesz.
[ ] Az i értéke 0 marad.
[ ] A ciklus addig fut, amíg az aktuális vagy a rákövetkező tömbelem értéke 0,
tehát az i értéke 4 lesz.
---
58. A flag egész típusú változó minden bitje valamilyen tulajdonság igaz vagy
hamis értékét jelent. Mit ír ki a program?
  #define FLAG 2
  int flags = 6;
  if(flags & FLAG != 0) printf("IGAZ\n"); else printf("HAMIS\n");
[ ] Az if utasítás feltétel része nem jó, mert az & operátor helyett && kell írni.
[ ] A precedencia szabály miatt a program a HAMIS szót írja ki.
[ ] Mivel 6 = 2 + 4 ezért alulról a második bit 1, tehát a program az IGAZ szót
írja ki.
---
59. Mennyi lesz az r változó értéke? int r, a=1, b=2; r = a << 2 + b;
[ ] 6
[ ] 8
[ ] 16
---
60. Mit ír ki a következő kódrészlet?
  int a=2, b=6, c=2;
  if(a < b < c) printf("novekvo"); else printf("csokkeno");
[ ] novekvo
[ ] az a<b<c kifejezés szintaktikailag hibás, ezért a program nem fordul le.
[ ] csokkeno
---
61. Mit ír ki a következő kódrészlet?
      #define ZZ(i) ((i)<2?(i):(i)+1)
      int j = 0;
      printf("%d", ZZ(j++)); printf("%d", ZZ(j++)); printf("%d\n", ZZ(j++));
[ ] 146
[ ] 123
[ ] 246
---
62. Mit ír ki a következő kódrészlet?
  int a[4]={1,0,3,5}; printf("%d", !!a[0]+!!a[1]+!!a[2]+!!a[3]);
[ ] A kódrészlet a nem nulla elemek számát írja.
[ ] A kétszer egymás után írt logikai nem műveletet el lehet hagyni,
ezért a korszerű fordítók azt kioptimalizálják, és az eredmény a
számok összege, azaz 9 lesz.
[ ] A program szintaktikailag hibás, mert !! operátor nem létezik.

X **
C Adatbevitel billentyűzetről, kiírás képernyőre
X **

---
2. Ha scanf függvénnyel olvasunk be értéket egy változóba, melyik állítás igaz?
[ ] A char s[100]; scanf("%s",&s[0]); beolvasás hibás, a helyes a
scanf("%s",s); lett volna.
[ ] Minden esetben a változó memóriacímét kell megadni a formátumstringet követő
felsorolásban.
[ ] A string esetét kivéve mindig a változó memóriacímét kell megadni a formá-
tumstringet követő @ felsorolásban.
---
3. Mi a getchar() függvény visszatérési típusa és miért?
[ ] A típus int, mert a modern rendszerekben UNICODE kódolásúak a karakterek,
ami 2 bájton ábrázol egy betűt.
[ ] A típus int, mert a speciális karaktereket olyan kóddal jelzi, ami nincs
benne a karakterek értékkészletében (EOF: -1).
[ ] A típus char, hiszen a getchar() karakteres adatok beolvasására való, ennek
típusa pedig a C-ben a char.
---
4. Helyesen működik-e az alábbi kódrészlet?
printf("%6d %3d%%", -874, 79);
[ ] Nem, mert a harmadik és a negyedik százalékjel után nem szerepel az
adatformátumot meghatározó karakter.
[ ] Nem, mert a 6d és a 3d helytelen adatformátum-sorozatok.
[ ] Igen.
---
5. Helyesen működik-e az alábbi kódrészlet?
printf("%6d %3d%%", -874);
[ ] Nem, mert a formátumfüzér alapján a függvény két bemeneti argumentumot várna.
[ ] Igen.
[ ] Nem, mert hibás a formátumot megadó első paraméter.
---
6. Helyes-e az alábbi program?
char *sztring;
scanf("%s", sztring);
[ ] Nem; a beolvasott szövegnek helyet kell foglalni.
[ ] Igen; tetszőleges hosszúságú szót olvas be a billentyűzetről.
---
7. Mit ír ki az alábbi kód, ha beírjuk, hogy "Hello, vilag!"?
char sztring[10]; scanf("%s", sztring);
[ ] "Hello, vi", mert a sztringünkbe csak 9 karakter fér, plusz egy lezáró nulla.
[ ] "Hello,", mert csak a szóköz karakterig olvas.
[ ] "Hello, vilag!", mert az enter leütéséig olvas.
---
8. Mennyi strlen("Hello, vilag!\n")?
[ ] 15.
[ ] 14.
[ ] 16.
---
9. Mit ír ki az alábbi program?
main(){ char c = 'X'; printf(c); }
[ ] Az X karakter ASCII kódját
[ ] Egy X karaktert.
[ ] A program futása memóriaelérési hibához vezet.
---
10. Mit jelent a printf("%d", variable); kifejezésben a %d?
[ ] Azt jelenti, hogy a változó értékét decimális egész formában kell kiírni.
[ ] Azt jelenti, hogy csak 1 változó értékét kell kiírni.
[ ] Azt jelenti, hogy a változó értékét dupla pontosságú lebegőpontos formában
kell kiírni
---
11. Melyik header fájlban található a scanf és a printf prototípusa?
[ ] stdio.h
[ ] stdlib.h
---
12. Hány karakterre fogja kiírni a következo kódrészlet a számot?
double d = 1827.732;
printf("%10.2f", d);

---
13. Mi az értéke az a, b, c-nek ha a felhasználó a következő szöveget adja meg?
Hello 14 736.55 uncle sam
int b; char a[30], c[30];
scanf("%s %d %*f %s", a, &b, c);
[ ] Szintaktikai hibás a kódrészlet.
[ ] Hello 14 736.55
[ ] Hello 14 uncle
---
14. Mennyi a kiírt átlag?
printf("Átlag: %d\n", (5+30+9)/3);

---
15. Mit ír ki a következő program?
#include <stdio.h>
void main() {   int a=234; printf(ä%06d\nö,a);} /* main() */
[ ] 000234
[ ] <SPACE><SPACE><SPACE>234
[ ] 234<SPACE><SPACE><SPACE>
---
16. Mi a visszatérési értéke a scanf("%lf", &d)-nek
[ ] Nincs is visszatérési értéke.
[ ] Ahány változó értéket kapott.
[ ] Ha sikeres a beolvasás, akkor 0 egyébkent -1.
---
17. Mit ír ki a következő kódrészlet?
printf("\"alma\"");

---
18. Melyik változat helyes, ha a standard bemenetrol érkezo karaktereket végig
akarjuk olvasni? (A void process(char c) egy saját, definiált függvény.)
[ ] while (c = getchar() != EOF)
  process(c);
[ ] while ((c = getchar()) != EOF)
  process(c);
---
19. Melyik kifejezés helyes?
(A) printf("%s%s", "%s");
(B) printf("%s", "%s%s");
[ ] Mindkettő
[ ] Csak (B)
[ ] Csak (A)
---
20. Mit ír ki a következő kódrészlet?
int i = 0;printf("%d", (i++)+(++i));
[ ] 2
[ ] 3
[ ] Nem lehet tudni
---
21. Meddig olvassa be a bemenetről a karaktereket a scanf függvény?
[ ] Amíg Enter-t nem ütünk.
[ ] Amíg Space-t nem olvas.
[ ] Ameddig illeszteni tudja a bemenetet a megadott formátum string-re.
---
22. Mit ír ki az alábbi program?
main(){ char c = 'X'; printf("%d", c); }
[ ] A program futása memóriaelérési hibához vezet.
[ ] Az X karakter ASCII kódját
[ ] Egy X karaktert.
---
23. Mit ír ki az alábbi kód?(ha az integer mérete 4 bájt)
unsigned int u= UINT_MAX;
printf("u= %d",u);
[ ] u= 4294967295 (=UINT_MAX)
[ ] u= -1
---
24. Mit ír ki az alábbi kód?(ha az integer mérete 4 bájt)
unsigned int u= UINT_MAX;
int z=-1;
printf("%d",u==z);

[ ] Szintaktikailag hibás a kód
[ ] 0
[ ] 1

---

25. Meddig fut a következő ciklus:
```
int i;
while( scanf( "%d", &i ) == 1 );
```
[ ] Csak 1 db számot olvas be
[ ] Végtelen ciklusban olvassa be a számokat, amíg az 1-es számot nem írjuk be
[ ] Csak addig olvas, amíg számokat írunk be

---

26. Mit ír ki a képernyőre az alábbi függvényhívás:
```
printf("%0*d", 5, 6);
```

---

27. A printf függvénnyel nem lehet pointert kiíratni.
[ ] Igaz.
[ ] Nem igaz és lehet értelme is.
[ ] Nem igaz, de nincs értelme, mert visszaolvasva már nem lehetünk benne biztosak, hogy érvényes a kiírt cím.

---

28. Mit ír ki az alábbi program?
```
printf("%.f", 12.25);
```

---

29. Igaz-e hogy két egész számot így célszerű beolvasni: scanf("%d%d",&n,&m)?
[ ] Igaz
[ ] Nem, mert a két szám között kell legyen whitespace karakter, ezért ezt csak így lehet: scanf("%d %d",&n,&m)
[ ] Felesleges az & operátor használata

---

30. Melyik állítás igaz? (csak egy jó)
[ ] A scanf függvény visszatérési értéke EOF ha nem sikerült beolvasni az adott formátummegadással egy adatot sem
[ ] Ha a scanf függvény formátumvezérlőjében egy fix szöveg van akkor addig olvasunk, ameddig meg nem kapjuk ezt a szöveget az inputon
[ ] A scanf fügvény terminálódik, amikor az inputon olyan adat érkezik, ami nem feleltethető meg a formátumvezérlő stringnek

---

31. Melyik a helyes módja a beírt számok megszámolásának?
[ ] `int i,n=0; while (!feof(stdin)) { n++; scanf("%d", &i); }`
[ ] `int i,n=0; while (scanf("%d", &i)==1) n++;`
[ ] `int i,n=0; while (scanf("%d", &i)==1 && i!=EOF) n++;`

---

32. Melyik programrészlet alkalmas leginkább egy könyv címének bekérésére?
[ ] `printf("A könyv címe:"); scanf("%s", nev);`
[ ] `printf("A könyv címe:"); scanf(nev);`
[ ] `printf("A könyv címe:"); gets(nev);`

---

33. Miért más a formátumvezérlő string double típus esetében?
(A printf-nél %f, míg a scanf-nél %lf)
[ ] Valóban más, de fordítva van, mert ilyenre sikerült a szabvány
[ ] Nem feltétlenül más, ugyanazzal is helyes
[ ] A printf esetében nincs értelme megkülönböztetni a float és a double típusokat, mert a float változó double-re konvertálódik a hívás után

---

34. Mi történik a következő kódrészletben, ha a bevitt karakterek száma 20?
```
char s[10]; gets(s);
```
[ ] A 20-ból csak 10 lesz figyelembe véve, mert s mérete ennyi
[ ] A pontos esemény implementációfüggő, de mindenképpen valami hiba lesz, mert a 20 karakterből csak az első 10 kerül az s tömbbe, a többi a memória utána következő részére kerül, amiről általánosan nem mondhatunk semmit
[ ] Semmi különös, az s változóval elérhető lesz mind a 20

X **
C A C nyelv utasításai
X **

---

2. Mit ír ki az alábbi program?
```
int main(){
  char s[] = "Utalom a kaposztat!!"; s[6] = 0; printf (s); return 0;
}
```

---

3. Hányszor fut az alábbi ciklus?
```
while(12)printf("*");
```
[ ] Egyszer sem fut le, mert szintaktikai hibás. A while ciklus zárójelei között egy logikai kifejezésnek kellene állnia, pl. hogy i<12.
[ ] 12-szer fut le, ezt jelzi a megadott szám.
[ ] Végtelen ciklus, addig ismétlődik, amíg le nem lőjük a programot.

---

4. Mi igaz az alábbi ciklusra?
```
for (i = j = 2; i = j; j--) printf("*");
```
[ ] Nem ír ki semmit, mert az inicializáló részben egyenlővé tettük i-t és j-t, ezért rögtön az első vizsgálatnál kilép.
[ ] Két darab csillagot ír ki.
[ ] Hibás, mert a for ciklus középső elemében nem logikai egyenlőség vizsgálat van, @ hanem értékadás.

---

5. Szintaktikailag helyes-e az alábbi kódrészlet?
```
for (; ;) { printf("x"); }
```
[ ] Igen, de a törzs csak egyszer fut le, mert nincs léptetés.
[ ] Igen, egy végtelen ciklust valósít meg.
[ ] Nem, mert hiányzik a ciklusváltozó inicializálása, a feltételrész és a ciklusváltozó léptetése.

---

6. Mit történik a continue utasítás hatására ciklusban?
[ ] A while, for és do-while ciklusok soron következő iterációját indítja el, így az aktuális iteráció hátralévő utasításait átugorja.
[ ] Hatására a program végrehajtása a megadott cimke utáni első utasítással folytatódik.
[ ] Hatására while, for és do-while ciklusok megszakadnak és a vezérlés a ciklus utáni első utasításra adódik át.

---

7. Mit történik a break utasítás hatására ciklusban?
[ ] Hatására while, for és do-while ciklusok megszakadnak és a vezérlés a ciklus utáni első utasításra adódik át.
[ ] Hatására a program végrehajtása a megadott cimke utáni első utasítással folytatódik.
[ ] A while, for és do-while ciklusok soron következő iterációját indítja el, így az aktuális iteráció hátralévő utasításait átugorja.

---

8. Mi a hatása a break-nek az alábbi kódban?
```
for(;a>b;a--){
    switch(x){
        case 1: break;
        case 2: continue;
    }
}
```
[ ] Kilép a switchből, és mivel más utasítás nincs a ciklus végéig, a-- jön.
[ ] A ciklus utáni utasítással folytatja, a-- műveletet már nem hajtja végre.
[ ] A fordító nem tudja eldönteni, hogy a ciklusból vagy a switchből akarunk kilépni, ezért fordítási hibát kapunk.

---

9. Mi a hatása a continue-nak az alábbi kódban?
```
for(;a>b;a--){
    switch(x){
        case 1: break;
        case 2: continue;
    }
}
```
[ ] A switch elejére ugrik, és x-et ismét kiértékelve ugrik a megfelelő case-re.
[ ] A for ciklus a<b kifejezésének kiértékelésével folytatja, a-- nem értékelődik ki ebben ez iterációban.
[ ] A for ciklus a-- kifejezésének kiértékelésével folytatja a ciklus következő iterációját.

---

10. Mi lesz b értéke?
```
int a = 19, b = 13; if(b = a) b = 9;
```

---

11. Mit ír ki a program?
```
main(){
 int a = 2;
 switch(a){
  case 1: printf("A");
  case 2: printf("B");
  default: printf("C");
 }
}
```

---

12. Mennyi lesz a értéke?
```
int a = 30; for(i=0; i<10; i++); a+=( (a%2) ? 5 : (a+4)/3 );
```

---

13. Mi lesz a következo kódrészlet lefutása után az 'a' változó értéke?
```
int i,j;   float a,b;
for (a=i=3.6, j=b=5.5; a<13;  a+=b, j--) do i*=j; while (++i<a);
```

---

14. Hányszor fut az alábbi ciklus?
```
while()printf("hello");
```
[ ] Végtelen ciklus, addig ismétlődik, amíg le nem lőjük a programot.
[ ] Addig fut, amíg meg nem nyomjuk az ENTER billentyűt.
[ ] Egyszer sem fut le, mert szintaktikai hibás. A while ciklus zárójelei között egy logikai kifejezésnek kellene állnia, pl., hogy i<12.

---

15. Hányszor fut az alábbi ciklus?
```
for(i=10; ;i++)
```
[ ] 10-szer fut le, ezt jelzi a megadott szám.
[ ] Végtelen ciklus, addig ismétlődik, amíg le nem lőjük a programot.
[ ] Egyszer sem fut le, mert szintaktikai hibás. A két pontosvessző között egy logikai kifejezésnek kellene állnia.

---

16. Hányszor fut az alábbi ciklus?
```
while(0)printf("hello");
```
[ ] Végtelen ciklus, addig ismétlődik, amíg le nem lőjük a programot.
[ ] Addig fut amíg, meg nem nyomjuk az ENTER billentyűt.
[ ] Egyszer sem fut le.

---

17. Hányszor fut az alábbi ciklus?
```
while(-100)printf("hello");
```
[ ] Egyszer sem fut le, mert szintaktikai hibás.
[ ] Végtelen ciklus, addig ismétlődik, amíg le nem lőjük a programot.
[ ] Addig fut amíg, meg nem nyomjuk az ENTER billentyűt.

---

18. Mi lesz min értéke, az alábbi sor lefutása után?
```
min = min > adat ? min : adat;
```
[ ] Mindig adat
[ ] A min és az adat minimuma.
[ ] A min és az adat maximuma.

---

19. Melyik változat a biztonságos?
```
   int a;
   scanf("%d", &a);
```
a) `if (100/a > 20) do_stg();`
b) `if (100/a > 20 && a != 0) do_stg();`
c) `if (a != 0 && 100/a > 20) do_stg();`
d) `if (a != 0)if(100/a > 20) do_stg();`
[ ] c) és d)
[ ] Mindegyik jó
[ ] b), c) és d)

---

20. Mi az fv(32) értéke?
```
int fv(unsigned x) {
  int c=0;
  for(; x; x>>=1) c+=x&1;
  return c;
}
```

---

21. Mi az fv(25) értéke?
```
int fv(unsigned x) {
  int c;
  for(c=0; x; x>>=1) c++;
  return c;
}
```

---

22. Mit ír ki az alábbi program?
```
int sum(int x, int y) { return x+y; }
int main() {
  int x = 3, y = sum(x++, ++x);
  printf("%d\n", y);
  return 0;
}
```
[ ] 9
[ ] Nem meghatározható, fordítótól függ.
[ ] 8

---

23. Mit ír ki a következő kódrészlet?
```
int j = 5;
for (i = --j; i > 4; i--) { printf("x"); }
```
J Semmit
R xxxx
R x

---

24. Hányszor fut le az alábbi ciklus?
```
int i=2;
while(i--) printf("A");
```
[ ] Egyszer
[ ] Kétszer
[ ] Háromszor

---

25. Hányszor fut le az alábbi ciklus?
```
int i=2;
while(--i) printf("A");
```
[ ] Kétszer
[ ] Egyszer
[ ] Háromszor

---

26. Mit ír ki az alábbi program?
```
int i=3, j=28;
switch (i) {
  default: j=7;
  case 1: j=16;
}
printf("%d\n", j);
```

---

27. Ha egy if utasítás magja üres, akkor akár el is hagyhatjuk a programból.
[ ] Nem igaz, a feltételtől függ.
[ ] Nem igaz, semmiképpen nem szabad elhagyni.
[ ] Igaz.

---

28. Mit csinál a for(;;); utasítás?
[ ] Ez hibás utasítás, a fordító hibát jelez
[ ] Ez egy felesleges utasitás, nem csinál semmit, a fordító kihagyja
[ ] Ez egy végtelen ciklus

---

29. Mi lesz ennek az utasításnak a hatása?
```
a>5;
```
[ ] Semmi; a kifejezés logikai értékét nem használjuk fel semmire.
[ ] 'a' változóba egy 5-nél nagyobb érték kerül.
[ ] Ha ezen a ponton, vagy valamikor később 'a' változó értéke 5, vagy annál kisebb, a program hibával kilép.

7. Állapotgépek

104. Egy repülőgéppel repülünk, és 100 m-ként megmérjük a felszín tengerszint feletti magasságát méterben. Készítsen programot, mely a beérkező adatok (scanf) eltárolása nélkül megállapítja, hogy
a.) jártunk-e a tenger felett?
b.) átrepültünk-e sziget felett? Ha igen, hány sziget felett?
Az első és utolsó mérést szárazföldön végeztük. (Az adatbeolvasás végét a -1 adat bevitele jelezze!)
Nehézség: 2

119. Készíts programot, amely a standard bemenetről érkező karaktersorozatról megmondja, hány ly betű található benne!
Nehézség: 2

120. Készítsen programot, amelyik a standard bemenetről érkező C forráskódú programot a standard kimenetre másolja; közben elhagyva a megjegyzéseket! Figyeljen arra, hogy sztring belsejében, " " karakterek között is lehetnek /* */ részek, amelyek azonban nem számítanak megjegyzésnek.
Nehézség: 4

154. Írjon programot, amelyik HTML formátumú fájlból eltávolítja a szedés jelöléseit, vagyis a <...> szerű karaktersorozatokat, ezzel többé-kevésbé olvashatóvá téve azt normál szövegként. (Például
, <title>).
Nehézség: 2

10. Mutatók, tömbök

103. Egy 195 fős előadóba előadást szerveznek. A székek 13 sorban, összesen 15 oszlopban (téglalap) helyezkednek el. A székek számozása a (színpaddal szembeni) bal alsó sarokból kezdődik, jobbra növekszik, és a legfelső sorban vannak a legnagyobb számú székek.
A jegyirodában egy 195 elemű egész tömbben tárolják a már kiadott foglalásokat. A foglalt helyeket a megfelelő székszámhoz írt 1-es jelöli, 0 a szabad hely. Készítsen programot, mely az újonnan érkező vendégek számára a kívánt számú, egymás melletti ülőhelyet kikalkulálja! Vigyázzon a sorok szélére!
Nehézség: 4

63. Készítsen programot, mely egy valós számokból álló tömb elemeit összegzi!
Nehézség: 1

64. Készítsen programot, amely egy tömb 3 legkisebb elemét határozza meg!
Nehézség: 2

65. Készítsen programot, amely egy v[10] tömb elemeit 3 hellyel ciklikusan balra lépteti! A feladat megoldható segédtömb nélkül is!
Nehézség: 2

66. Készítsen függvényt, mely a paraméterben kapott egész tömbről megvizsgálja, hogy elemeinek értéke
a.) monoton növekvő
b.) szigorú monoton növekvő
c.) szomszédos elemek E sugaron belül helyezkednek el (különbségük nem nagyobb, mint E vagy -E)
A függvény bemenő paramétere a tömbre mutató pointer, a tömb elemeinek száma, valamint a c.) feladatnál E értéke. Visszatérési értéke int, mely igaz esetén 1, egyébként 0 legyen.
Nehézség: 2

3. Ciklusok

123. Írja ki a képernyőre az összes N-nél kisebb négyzetszámot!
Nehézség: 1

127. Készítsen programot, amely a felhasználó által megadott szám prímtényezős felbontását írja ki. Például:

```
20 | 2
10 | 2
 5 | 5
 1 |
```

Nehézség: 2

148. Készítsen programot, amely egy felhasználó által megadott számot kiír az ugyancsak általa megadott számrendszerben. Például a 16 a 3-as számrendszerben: 1*3^2 + 2*3^1 + 1*3^0.
Nehézség: 2

*