Programozás alapjai 1. (BMEVIEEA100)

Gyakorlat anyaga az 7. oktatási héten

Figyelem, ez az anyag a szokottnál előbb érkezett, az október 18-22 (szombat-szerda) időszakhoz tartozik!

A 8. héten írják a második kis ZH-t az „Adatbevitel billentyűzetről, kiírás képernyőre”, „A C nyelv utasításai”, „Függvények használata” című fejezetekből. Ezek mindegyike feladásra került már a hétfői és szerdai csoportokban, a péntekiekben most adjuk föl az utolsót.

Röviden

Hétfői és szerdai csoportok feladata: Stringkezelő függvények írása, elmaradások pótlása, gyakorlás. Házi feladat az „Enum, tömb, struktúra, stringek” című fejezet a tesztek közül, és a példák az anyag végén. A struktúrák előadáson már szerepeltek, így ezek ismerete sem jelenthet problémát.

Pénteki azaz szombati csoportok feladata: A múlt heti anyagban szereplő egyszerű függvények írása, pointerek és dinamikus tömbök. Házi feladat: az anyag végén látható feladatokból azt adjátok, amit jónak láttok. Továbbá a „Függvények használata” című fejezet a tesztkérdések közül. Ezen kívül mindaz, ami ezen a héten a hétfői és szerdai csoportoké, most érik utol őket.

Új anyag

Írjunk néhány stringmanipuláló függvényt, pl.: másolás, hozzáfűzés, helyben megfordítás, összefésülés. Mutassunk példát tömbös és pointeres megoldásokra egyaránt. Hívjuk fel a figyelmet arra, hogy függvényparaméterként a char* és a char [] általában egyenértékű, de a char [] módon átadott pointer nem változtatható meg (az általa mutatott elemek igen).

```
#include <stdio.h>

void stringmasolo_1(char cel[], char forras[]){
    int i;
    for(i=0;forras[i]!=0;i++)cel[i]=forras[i];
    cel[i]=0;//A stringet lezáró 0-t is be kell tenni
}

void stringmasolo_2(char * cel, char * forras){
    int i;
    for(i=0;cel[i]=forras[i];i++);
}

void stringmasolo_3(char * cel, char * forras){
    while(*cel++=*forras++);
}

void hozzafuz(char * cel, char * forras){
    while(*cel++);
    stringmasolo_1(cel-1,forras);
}

void megfordit(char * s){
    char * p=s;
    while(*p++);
    for(p-=2;p>s;p--,s++){char c=*p;*p=*s;*s=c;}
}
```

```
void fesul(char * cel, char * s1,char * s2){
    int i,j;
    for(i=0;s1[i]!=0&&s2[i]!=0;i++){
        cel[2*i]=s1[i]; cel[2*i+1]=s2[i];
    }
    if(s1[i]!=0)for(j=0;cel[2*i+j]=s1[i+j];j++);
    else        for(j=0;cel[2*i+j]=s2[i+j];j++);
}

int main(){
    char s1[100]="Egy ludnyak tobb tiz tyuknyaknal.",s2[100],s3[100];
    s2[0]=0;                printf("s2 ures: %s\n",s2);
    stringmasolo_1(s2,s1);  printf("s2 teli: %s\n",s2);
    s2[0]=0;                printf("s2 ures: %s\n",s2);
    stringmasolo_2(s2,s1);  printf("s2 teli: %s\n",s2);
    s2[0]=0;                printf("s2 ures: %s\n",s2);
    stringmasolo_3(s2,s1);  printf("s2 teli: %s\n",s2);
    hozzafuz(s2,s1);        printf("s2 dupla: %s\n",s2);
    megfordit(s2);          printf("s2 ford: %s\n",s2);
    stringmasolo_1(s1,"02468");
    stringmasolo_1(s2,"13579ABC");
    fesul(s3,s1,s2);        printf("s2 ford: %s\n",s3);
   }
```

A fennmaradó időt fordítsuk az eddigi anyag gyakorlására, ismétlésre, házi feladat ellenőrzésre! (A szombati csoportban vélhetőleg nem lesz fennmaradó idő.)

Házi feladat

```
X ****************************************************************************
C Enum, tömb, struktúra, sztringek
X ****************************************************************************
----------------------------------------------------------------------------
2. Adott az alábbi változó-értékadás:
int a[] = {23,29,31,11,31,5};
Mi az értéke az &a[1]-a kifejezésnek?

----------------------------------------------------------------------------
3. Mit ír ki az alábbi függvény, ha az adott gépen egy pointer 4 karakternyi?
void syzeof(char t[17]){printf("%u",sizeof(t);}

----------------------------------------------------------------------------
4. Mit ír ki az alábbi függvény, ha az adott gépen egy pointer 4 karakternyi?
void syzeof(){char t[23];printf("%u",sizeof(t);}

----------------------------------------------------------------------------
5. Mennyi d értéke az alábbi felsorolt típusban?
enum {a = 7, b = 14, c = 21, d};
[ ] 28, mert az következik a sorozatban.
[ ] Nincs inicializálva, így nem lehet megmondani.
[ ] 22, mert ha másként nincs megadva, akkor eggyel nagyobb egy konstans értéke,
mint az őt megelőző.
----------------------------------------------------------------------------
6. Hány darab és milyen elemei lesznek az alábbi tömbnek?
double t[] = {1.2, 2.4, 3.6};
[ ] Három darab eleme lesz, azok, amiket felsoroltunk.
[ ] Hibás a tömb definíció, mert nem adtuk meg a tömb méretét.
----------------------------------------------------------------------------
7. Érvényes-e az alábbi struktúra-definíció és ha igen, milyen adatszerkezetet
valósít meg?
struct Konyvek {
   char szerzo[20];
   char cim[50];
   int kiadas_eve;
   struct Konyvek kovetkezo;
   struct Konyvek elozo;
};
[ ] Helyes a definíció. Az elnevezéstől függetlenül megvalósítható vele két
irányban láncolt lista és bináris fa is.
[ ] Nem jó a definíció, mert rekurzív a szerkezet.
[ ] Helyes definíció, egy láncolt lista alapelemét írja le.
----------------------------------------------------------------------------
8. Adott az alábbi változó-értékadás:
int a[] = {1,2,3,4};
Okoz-e fordítási idejű hibát az a[-1] kifejezés?
[ ] Igen, mert a tömbök indexelése nullával kezdődik.
[ ] Igen, mert a tömbök indexelése eggyel kezdődik.
[ ] Nem.
----------------------------------------------------------------------------
9. Adottak az alábbi változó-értékadások:
int a[] = {1,2,3,4};
int* p = a;
Okoz-e fordítási idejű hibát a p[6] kifejezés?
[ ] Igen, mert túlcímzés történik.
[ ] Igen, mert mutató típusú változóhoz egy tömb típusú változót rendelünk
értékként, illetve a [] operátorral csak tömbök indexelhetőek.
[ ] Nem.
----------------------------------------------------------------------------
10. Mit ír ki a következő kód?: char *s="abcdef"; printf("%d", sizeof(s));
[ ] 7, mert hat betű, és a lezáró nulla karakter a sztring mérete.
[ ] A számítógép típusától függ.
[ ] 6, mert hat betűt tartalmaz a sztring.
----------------------------------------------------------------------------
11. Teljesül a feltétel az alábbi kódban?
char s[] = "Hello!", t[] = "Hello!";
if(s == t) printf("A két sztring egyforma!");
[ ] Igen.
[ ] Nem.
----------------------------------------------------------------------------
12. Teljesül a feltétel az alábbi kódban?
char s[] = "Hello!", t[] = "Hello!";
if(strcmp(s, t)) printf("A két sztring egyforma!");
[ ] Igen.
[ ] Nem.
----------------------------------------------------------------------------
13. 10 darab, képernyőn pattogó labda koordinátáit szeretnénk tárolni. Melyik
a helyes kódolási technika?
A: int golyo_x[10], golyo_y[10];
B: struct golyo{ int x; int y; };
   struct golyo golyok[10];
[ ] Mindkettő.
[ ] Az A, mert sokkal egyszerűbb.
[ ] A B, mert kifejezi a koordináták összetartozását.
----------------------------------------------------------------------------
14. Melyik állítás igaz?
[ ] Függvények hívásakor mindig fel kell tüntetni a paraméterek típusát.
[ ] A függvények a struktúrákat érték szerint veszik át.
[ ] A függvények a tömböket érték szerint veszik át.
----------------------------------------------------------------------------
15. Válassza ki a hamis állítást!
[ ] Az 'a' tömb i. elemét *(a+i) módon is elérhetjük
[ ] C: Ha 'a' egy tömb akkor az a++ művelet a 2. elem címét adja vissza.
[ ] Az 'a' tömb i. elemének címe a+i
----------------------------------------------------------------------------
16. Rekurzív struktúrát akarunk létrehozni. Mit kell a ??? helyére írni?
typedef  struct s { ??? p; int e; } str;
[ ] str *
[ ] struct s *
[ ] struct * s
----------------------------------------------------------------------------
17. Mit ír ki az alábbi program?
#include <stdio.h>
typedef enum {hetfo, kedd, szerda, csutortok, pentek, szombat, vasarnap} napok;
napok nap = 22 % 7;
void main(void) {nap = (nap +  kedd) % 7; printf("%d", nap);}

----------------------------------------------------------------------------
18. Mit tartalmaz az s = "Nem értem" sztring változó, ha meghívtuk a shuf(s)
függvényt?
void shuf(char *trs) {
    int i;
   for (i = 0; trs[i] && trs[i+1]; i += 2) {char c = trs[i]; trs[i]=trs[i+1];
trs[i+1]=c;}
}
[ ] Beugratós a feladat: a változó értéke nem változik meg, hiszen a C-ben a
függvények nem tudják
megváltoztatni a paraméterként kapott változók értékét.
[ ] Minden második betű helyére az őt követőt teszi: "ee  éttm"
[ ] Felcseréli az egymást követő betűket: "eN mréetm"
----------------------------------------------------------------------------
19. Mit ír ki a program?
```

```
int a[] = {2, 0, 0, 8};
void main(void) {printf("%d", %u1 % 2 ? a[1] : a[3]);}
```

---

20. Mit ír ki a program?

```
union {int Int; struct {char Lowbyte; char Highbyte;} Char;} Unio;
void main(void) {
    Unio.Int = 1 + 31 + 9;
   printf("%d", Unio.Char.Lowbyte);
}
```

---

21. Mit ír ki az alábbi program?

```
#define SIZE 29
void main(){
  double x[SIZE], *fp; int i;
  for (i = 0; i < SIZE; i++){ x[i]=0.5*(double)i; fp=x; }
  printf("%f", *(fp+i-1));
}
```

---

22. Milyen típusú lesz az var1 változó utolsó értéke az alábbi programban?

```
#include <stdio.h>
int main() {
  char Cond=0x55 & 0xaa; int k=8;
  if (Cond != 0) {
    int var1;
    if (k != 4) { double var1; var1 = 6.27; }
    var1 = 3 * 2 ;
  }
  else char var1[20] = "X123456789";
  return 0;
} /* end main() */
```

[ ] int
[ ] char
[ ] double

---

23. Helyesen működik-e az alábbi program?

```
#include <stdio.h>
void main(){ double x=3.14, y; int *p;  p = &x; y = *p;  printf("%f",y);}
```

[ ] Nem
[ ] Igen

---

24. Mit ír ki az alábbi program?

```
main(){
  char word[20]; word[0] = 'B'; word[1] = 'Y';
  word[2] = 'E'; word[3] = 0; word[4] = 'B';
  printf("The contents of word[] is -->%s\n", word);
}
```

[ ] The contents of word[] is --> BYE
[ ] The contents of word[] is --> BYE0B
[ ] The contents of word[] is --> BYE B

---

25. Mit ír ki az alábbi program?

```
main(){
  char word[20];
  word[0] = 'B'; word[0]++;
  word[1] = 'Y';
  word[2] = 'E'; --word[2];
  word[3] = 0;
  word[4] = 'B';
  printf("The contents of word[] is -->%s\n", word);
}
```

[ ] The contents of word[] is --> CYD
[ ] The contents of word[] is --> BYD
[ ] The contents of word[] is --> BYE

---

26. Mennyi lesz a következő enumban c értéke?

```
enum {a = 0, b = 4, c};
```

---

27. Helyes-e az alábbi kódrészlet?

```
int i;
double t[5];
for( i = 1; i <=5; i++ ){
   printf("\nt[%d]", i );
   scanf("%lf", &t[i]);
}
```

[ ] Igen.
[ ] Nem, mert kiindexelünk a tömbből

---

28. Mennyi lesz x értéke?

```
int t[] = {7, 62, 22, 16, 32, 157}
int *p = t + 2;
int x = p[1];
```

---

29. Helyes-e a következő változó definiálás és értékadás?

```
     char t[] = "Hello";
```

[ ] Nem
[ ] Igen

---

30. Helyes-e a következő változó definiálás és értékadás?

```
     char t1[10]; t1 = "Hello";
```

[ ] Nem
[ ] Igen

---

31. Mit ír ki a következő programrészlet?

```
   char str[] = "qXeu327";
   char *p = str;
   while (*p){
       if (*(p+1)) *p = *(p+1);
       p++;
   }
   printf("%s", str);
```

---

32. Adott az alábbi változó-értékadás: char s[]="Hello!";
Okoz-e fordítási idejű hibát a s[6] kifejezés?

[ ] Nem, nincs ellenőrzés a túlcsordulásra.
[ ] Igen, mert túlcímzés történik.
[ ] Nem, futásidőben a lezáró \0 karakter kapjuk vissza.

---

33. Mi történik, ha a konzolon a "Juliska" szöveget gépeljük be, miután elindítottuk az alábbi programot?

```
int main(){ char nev[7]; scanf("%s", nev); printf("%s", nev); return 0; }
```

[ ] Juliska íródik ki.
[ ] Julisk íródik ki.
[ ] A sztring nem fér bele a tömbbe, ami a program elszállását vagy hibás működését is okozhatja, de a hiba nem feltétlenül jelentkezik.

---

34. Hány darab és milyen elemei lesznek az alábbi tömbnek?

```
int k[3]={1,2,3,4};
```

[ ] Hibás a tömb definíció, mert az több inicializálási értéket adtunk meg mint

```
a tömb elemszáma
[ ] Négy darab eleme lesz, azok, amiket felsoroltunk.
[ ] Három darab eleme lesz, az első három a felsoroltak közül.
--------------------------------------------------------------------------------
35. Mekkora egy struktúra mérete?
[ ] Pontosan akkora, mint a benne lévő adatok összmérete.
[ ] Legalább akkora, mint a benne lévő adatok összmérete.
[ ] Nagyobb, mint a benne lévő adatok összmérete, mert a rendszer eltárolja a
struktúra leírását is a struktúrában.
--------------------------------------------------------------------------------
36. Mi az strlen(s) == strlen(&s[0]) kifejezés értéke, ha char *s = "sztring";
[ ] Hamis
[ ] Igaz
--------------------------------------------------------------------------------
37. Van-e különbség az alábbi két sor között?
char str[]="alma";
char *str="alma";
[ ] Van különbség az str változó kiíratásában.
[ ] Nincs különbség.
[ ] Van különbség a memória kezelésében.
--------------------------------------------------------------------------------
38. Mit lesz a size változó értéke?
char *msg = "Hello";
size_t size = sizeof(msg)/sizeof(char);
[ ] rendszerfüggő
[ ] 5
[ ] 6
--------------------------------------------------------------------------------
39. Ha egy tömböt adunk át paraméterként egy függvénynek, és a függvényben
szeretnénk változtatni a tömb tartalmán, akkor szükséges-e az & operátor?
[ ] Igen, hiszen mindig kell az & operátor ha egy változót meg akarunk
változtatni egy függvényen belül.
[ ] A tömb tartalmát nem lehet megváltoztatni a paraméterlistán keresztül.
[ ] Nem, mert a tömb neve a tömb első elemére mutató pointer.
--------------------------------------------------------------------------------
40. Mikor adja meg a sizeof operátor a tömb méretét?
[ ] Ahol deklaráltuk attól kezdve mindig, függvényekben is
[ ] Csak abban a blokkban, ahol deklarálva van (hacsak nem globális)
[ ] Mindig
--------------------------------------------------------------------------------
41. Lehet-e egy függvény visszatérrési értéke egy struktúra?
[ ] Igen
[ ] Nem, a visszatérési érték csak alaptípus lehet
[ ] Nem, csak a rá mutató pointer
--------------------------------------------------------------------------------
42. Adott az alábbi változó-értékadás:
int t[] = {31, 21,  3,  2, 22, 25};
Mi az értéke a (t+2)[t[3]+1] kifejezésnek?

--------------------------------------------------------------------------------
43. Hogyan viszonyul egymáshoz sizeof(struct polip1) és sizeof(struct polip2)?
struct polip1{ char kar[8]; double szem; };
struct polip2{ char kar1[6]; double szem; char kar2[2]; };
[ ] Egyenlők, hiszen ugyanakkorák külön-külön az adattagok mindkét struktúrában.
[ ] Rendszer- ill. fordítófüggő; eltérhetnek.
[ ] sizeof(struct polip2) 1-gyel nagyobb, mert ott két karaktertömbhöz is
tartozik egy lezáró '\0'.
--------------------------------------------------------------------------------
44. Mit mondhatunk a következő kódrészletről?
       char str[]="alma"; char* p; p = str;
[ ] A p pointer az "alma" sztringre fog mutatni.
[ ] A p = str értékadás hibás, mert az egyik pointer, a másik tömb.
[ ] A p által mutatott területre átmásolja az "alma" sztringet,
de mivel a p még nem kapott értéket, ezért a program valószínűleg
el fog szállni.
--------------------------------------------------------------------------------
45. Mit eredményez az alábbi program?
#include <stdio.h>
typedef enum {januar=1, februar, marcius,telho=1, teluto, tavaszelo} honapok;
void main(void){
    honapok kedvencem=marcius;
    printf("februar a %d. honap, tavaszelo a %d., kedvencem a %d.\n",
      februar, tavaszelo, kedvencem);
    fflush(stdin); getchar(); return;
}
[ ] februar a 2. honap, tavaszelo a 3., kedvencem a 3.
[ ] februar a februar. honap, tavaszelo a tavaszelo., kedvencem a marcius.
[ ] Nem fordul le: két különböző szimbólum azonos sorszámhoz a felsorolásban.
--------------------------------------------------------------------------------
46. Péter a következő kódrészben a 'b' tömbbe át akarja másolni az 'a' tömb
tartalmát, addig amíg nullát nem talál (a nullát is), de maximum csak annyit,
ami a 'b' tömbbe belefér. A kapkodásban a while ciklus logikai kifejezésébe
&& helyett & került.
Mit mondhatunk a következő kódrészletről?
      int a[10]={8,9,10,11,12,13,14,15,16,0}, b[5], i=0;
      while( i<5 & ( (b[i]=a[i]) != 0) ) i++;
[ ] A program nem fordul le, mert a bitenkénti ÉS kapcsolat
csak egész típusokra értelmezett.
[ ] Nem baj, így is jól fog működni a ciklus, mert az & jobb és bal oldalán is
logikai kifejezés áll, és a ciklusba maradás feltétele IGAZ & IGAZ érték is
IGAZ, ha pedig valamelyik oldal értéke hamis, akkor a teljes kifejezés
értéke is hamis.
[ ] A kód nem működik helyesen, mert i==5 esetén is átmásolja az a[i] értékét a
b[i]-be, tehát túlcímzi a 'b' tömböt.
```

Példák

1. Írjunk függvényt, mely paraméterként kap egy stringet, és benne minden kisbetűt nagybetűvé alakít!
2. Írjunk függvényt, mely a paraméterként kapott stringből eltávolítja a whitespace karaktereket! Pl.: bemenet: „Minden egér szereti a sajtot.” kimenet: „Mindenegérszeretiasajtot.” (Abban a tömbben adja vissza, amelyben kapta!)
3. Írjon függvényt, mely paraméterként kap egy szöveget tartalmazó karaktertömböt, és úgy alakítja át, hogy a képernyőn megjeleníthető legyen, azaz legfeljebb 80 karakterenként beiktat egy újsor karaktert (’\n’). Az újsor karaktereket kizárólag szóköz karakterek helyére teheti! Ha egymás után nyolcvannál több nemszóköz karakter érkezik, akkor az ezt követő első szóközt kell helyettesíteni, az egybefüggő részt nem bántjuk! Ügyeljen arra, hogy a sorok a lehető leghosszabbak legyenek, azaz minden sorban a 80. karakter előtti utolsó szóközt helyettesítse!