

Programozás C nyelven (5. ELŐADÁS)

Sapientia EMTE

2022-23



for – ciklus



OH NO!
I FORGOT ...
SOMETHING ...
... BUT WHAT ?

```
int n, i;  
scanf("%i", &n);  
for ( i=1 ; i<=n ; ++i ){  
    printf("( %i)", i);  
}
```

```
5  
(1)(2)(3)(4)(5)_
```

```
int n;  
scanf("%i", &n);  
for ( ; n ; --n ){  
    printf("( %i)", n);  
}
```

```
5  
(5)(4)(3)(2)(1)_
```

```
int n, i, j;  
scanf("%i", &n);  
for ( i=0, j=n-1; i<j ; ++i, --j ){  
    printf("( %i) ( %i) \n", i, j);  
}
```

```
7  
(0)(6)  
(1)(5)  
(2)(4)  
_
```

```
int n, i;  
scanf("%i", &n);  
for ( i=0; i<n ; ++i ){  
    printf("( %i) ( %i) \n", i, n-1-i);  
}
```

```
5  
(0)(4)  
(1)(3)  
(2)(2)  
(3)(1)  
(4)(0)  
_
```

for – ciklus



```
int n, i, j;
scanf("%i", &n);
for ( i=0; i<n ; ++i ){
    printf("(%i)", i);
}
printf("\n");
for ( j=0; j<n ; ++j ){
    printf("(%i)", j);
}
printf("\n");
```

```
5
(0)(1)(2)(3)(4)
(0)(1)(2)(3)(4)
—
```



```
int n, i, j;
scanf("%i", &n);
for ( i=0; i<n ; ++i ){
    for ( j=0; j<n ; ++j ){
        printf("(%i,%i)", i, j);
    }
    printf("\n");
}
```

```
5
(0,0)(0,1)(0,2)(0,3)(0,4)
(1,0)(1,1)(1,2)(1,3)(1,4)
(2,0)(2,1)(2,2)(2,3)(2,4)
(3,0)(3,1)(3,2)(3,3)(3,4)
(4,0)(4,1)(4,2)(4,3)(4,4)
—
```

Egymásba ágyazott ciklusok

Jelenítsünk meg a képernyőn $n \times n$ méretű csillag-mátrixot!

5

```
*****  
*****  
*****  
*****  
*****
```

```
int main(){  
    int n, i;  
    scanf("%i", &n);  
    for( i=0 ; i < n*n ; ++i){  
        if (i%n == 0){  
            printf("\n"); //n-szer  
        }  
        printf("*"); //n*n-szer  
    }  
    return 0;  
}
```

```
int main(){  
    int n, i, j;  
    scanf("%i", &n);  
    for( i=1 ; i <= n ; ++i){  
        for( j=1 ; j <= n ; ++j){  
            printf("*"); //n*n-szer  
        }  
        printf("\n"); //n-szer  
    }  
    return 0;  
}
```

5

```
*****  
*****  
*****  
*****  
*****
```

$i \leftarrow 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25$

Egymásba ágyazott ciklusok

```
int main(){
    int n, i, j;
    scanf("%i", &n);
    for( i=0 ; i < n ; ++i){
        if (i%2){
            for( j=0 ; j < n ; ++j){
                printf("-");
            }
        }
        else{
            for( j=0 ; j < n ; ++j){
                printf("+");
            }
        }
        printf("\n");
    }
    return 0;
}
```

```
5
+++++
-----
+++++
-----
+++++
-----
```

```
int main(){
    int n, i, j;
    scanf("%i", &n);
    for( i=1 ; i <= n ; ++i){
        for( j=1 ; j <= n ; ++j){
            if (i<j){
                printf("-");
            }
            else if(i>j){
                printf("+");
            }
            else{
                printf("o");
            }
        }
        printf("\n");
    }
    return 0;
}
```

```
5
o----
+o---
++o--
+++o-
++++o
5
```

Adott egy $n \times m$ méretű **mátrix** a be.txt állományban.
Írassuk ki a képernyőre az **elemek összegét**.

```
int main(){
    freopen("be.txt", "r", stdin);
    int szam, n, m, i, s;
    scanf("%i%i", &n, &m);
    s = 0;
    for( i=1 ; i <= n*m ; ++i){
        scanf("%i", &szam); // (n*m)-szer
        s += szam; // (n*m)-szer
    }
    printf("%i", s);
    return 0;
}
```

be.txt

```
2 3
44 5 13
7 -10 11
```

```
int main(){
    freopen("be.txt", "r", stdin);
    int szam, n, m, i, j, s = 0;
    scanf("%i%i", &n, &m);
    for( i=1 ; i <= n ; ++i){
        for( j=1 ; j <= m ; ++j){
            scanf("%i", &szam); // (n*m)-szer
            s += szam; // (n*m)-szer
        }
    }
    printf("%i", s);
    return 0;
}
```

be.txt

```
2 3 44 5 13 7 -10 11
```

Adott egy $n \times m$ méretű **mátrix** a be.txt állományban.
Írassuk ki az elemek **soronkénti összegeit**.

```
int main(){
    freopen("be.txt", "r", stdin);
    int szam, n, m, i, s;
    scanf("%i%i", &n, &m);
    s = 0;
    for( i=0 ; i < n*m ; ++i){
        scanf("%i", &szam);
        // (n*m)-szer
        s += szam;
        if (i%m == m-1){
            printf("%i", s); s = 0;
            //n-szer
        }
    }
    return 0;
}
```

be.txt							
2	3	44	5	13	7	-10	11

be.txt	
2	3
44	5 13
7	-10 11

```
int main(){
    freopen("be.txt", "r", stdin);
    int szam, n, m, i, j, s;
    scanf("%i%i", &n, &m);
    for( i=1 ; i <= n ; ++i){
        s = 0; //n-szer
        for( j=1 ; j <= m ; ++j){
            scanf("%i", &szam);
            // (n*m)-szer
            s += szam; // (n*m)-szer
        }
        printf("%i\n", s); //n-szer
    }
    return 0;
}
```

Adott egy **$n \times n$** méretű **mátrix** a **be.txt** állományban.
Írassuk ki a **főátló feletti** elemek összegeit.

```
int main(){
    freopen("be.txt", "r", stdin);
    int szam, n, k, i, j, s;
    scanf("%i", &n);
    s = 0;
    for( k=0 ; k < n*n ; ++k){
        scanf("%i", &szam);
        i = k/n; j = k%n;
        if (i < j){
            s += szam;
        }
    }
    printf("%i", s);
    return 0;
}
```

```
k←0,1,2,3,4,5,6,7,8,9
i←0,0,0,1,1,1,2,2,2
j←0,1,2,0,1,2,0,1,2
```

be.txt
3
44 5 13
7 -10 11
6 7 1

```
int main(){
    freopen("be.txt", "r", stdin);
    int szam, n, i, j, s = 0;
    scanf("%i", &n);
    for( i=1 ; i <= n ; ++i){
        for( j=1 ; j <= n ; ++j){
            scanf("%i", &szam);
            if (i < j){
                s += szam;
            }
        }
    }
    printf("%i", s);
    return 0;
}
```

be.txt
3 44 5 13 7 -10 11 6 7 1

Függvények

```
#include <stdio.h>
#include <math.h>
int main(){
    double x;
    scanf("%lf", &x);
    printf("Gyoke: %lf\n", sqrt(x));
    return 0;
}
```

Könyvtári
függvények



```
#include <iostream>
int main(){
    int a, b;
    scanf("%i%i", &a, &b);
    printf("LNKO: %i\n", loko(a,b));
    return 0;
}
```

SAJÁT
függvények





main függvény



visszatérített
eredmény típusa

név /
azonosító

paraméter-lista
<típus> <azonosító>

```
int main () {
```

. . .

```
return 0;
```

```
}
```

függvény
törzse / magva

Az operációs rendszer hívja meg,
és ennek téríti vissza a 0 hibakódot

Saját (valódi)függvények

Függvény DEKLARÁCIÓ

<típus> **<azonosító>** (**<típus₁>**, ..., **<típus_N>**);

<típus> **<azonosító>** (**<típus₁>** **<név₁>**, ...,
<típus_n> **<név_n>**) {

○ ○ ○

return **<eredmény>**;

}

Függvény DEFINÍCIÓ



```
#include <stdio.h>
```

```
int lnko(int, int);
```

```
int main() {
```

```
    int a, b, c;
```

```
    scanf("%i%i", &a, &b);
```

```
    c = lnko(a, b);
```

```
    printf("%i", c);
```

```
    return 0;
```

```
}
```

```
int lnko(int x, int y) {
```

```
    while (x != y) {
```

```
        if (x < y) { y -= x; }
```

```
        else { x -= y; }
```

```
    }
```

```
    return x;
```

```
}
```

deklaráció

függvény
hívás

d
e
f
i
c
i
ó

```
#include <stdio.h>
```

```
int lkkt(int, int);
```

```
int main() {
```

```
    int a, b;
```

```
    scanf("%i%i", &a, &b);
```

```
    printf("%i", lkkt(a, b));
```

```
    return 0;
```

```
}
```

```
int lkkt(int x, int y) {
```

```
    int sx = x, sy = y;
```

```
    while (sx != sy) {
```

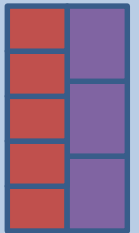
```
        if (sx < sy) {sx += x;}
```

```
        else {sy += y;}
```

```
    }
```

```
    return sx;
```

```
}
```



Inko - lépésenként

```

#include <stdio.h>
int lngo(int,int);
int main() {
    int a, b, c;
    scanf("%i%i",&a,&b);
    c = lngo(a,b);
    printf("%i", c);
    return 0;
}

int lngo(int x, int y) {
    while (x != y) {
        if (x < y) {y -= x;}
        else { x -= y; }
    }
    return x;
}

```

színészek

effectív/aktuális
paraméterek

`c = lngo(12,21);`

`c = 3;`

main

lngo

formális
paraméterek

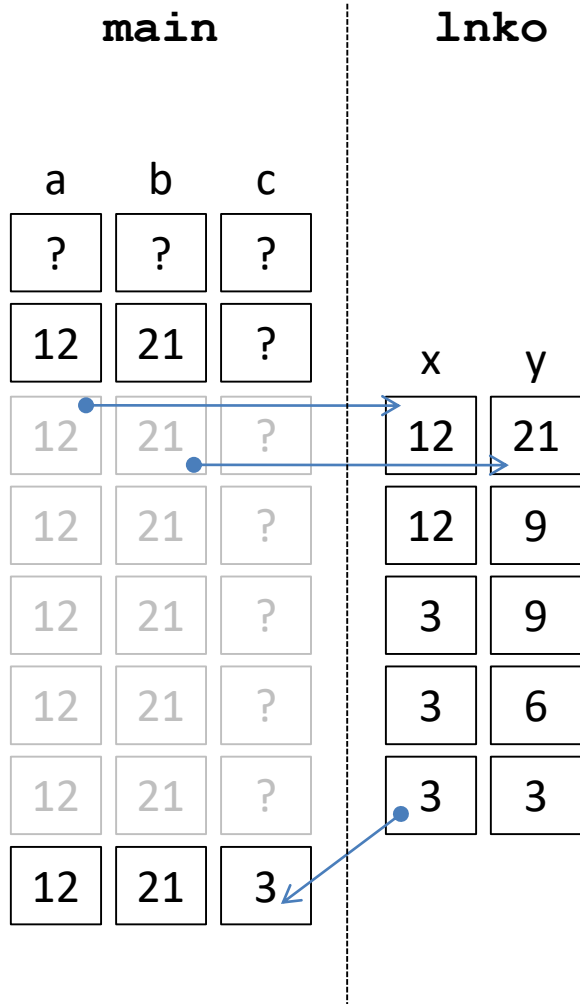
szeplők/
dublörök

12 21_

12 21
3_



Inko - lépésenként



```
#include <stdio.h>
int lnko(int,int);
int main(){
    int a, b, c;
    scanf("%i%i",&a,&b);
    c = lnko(a,b);
    printf("%i", c);
    return 0;
}
int lnko(int x, int y){
    while (x != y){
        if (x < y) { y -= x; }
        else { x -= y; }
    }
    return x;
}
```

12 21_

12 21
3_

ÖSSZEFOGLALÓ

- **(valódi) FÜGGVÉNYEK**

- deklaráció / definíció
- formális/effektív paraméterek
- paraméter-átadás / `return` (STACK)
- hívó/hívott függvény

- **Lnko / Lkkt számítás technikája**



Egymásba ágyazott ciklusok és függvények

Adott n természetes szám. Hány közülük prím?

```
int n,i,j,szam,k;
scanf("%i", &n);
k = 0;
for( i=1 ; i<=n ; ++i ){
    scanf("%i", &szam);
    if ( szam == 0 || szam == 1 ) {continue;}
    bool prim = true;
    for( j=2 ; j<=sqrt(szam) ; ++j ){
        if ( szam%j == 0 ){
            prim = false; break;
        }
    }
    if ( prim ) { ++k; }
}
printf("%i", k);
```



TOP-DOWN programozás



```
int main(){
    int n,i,szam,k;
    scanf("%i", &n);
    k = 0;
    for( i=1 ; i<=n ; ++i){
        scanf("%i", &szam);
        if ( prim_e(szam) ) { ++k; }
    }
    printf("%i", k);
    return 0;
}
```

```
bool prim_e (int);
```

```
bool prim_e (int x){
    int i;
    if (x==0 || x==1) {return false;}
    for( i=2 ; i<=sqrt(x) ; ++i ){
        if ( x%i == 0 ){
            return false;
        }
    }
    return true;
}
```

Természetes számokat olvasunk be 0 végjelig. Hány közülük szuper-prím?

```
int main(){
  int szam,k;
  k = 0;
  for( ; ; ){
    scanf("%i", &szam);
    if ( !szam ) { break; }
    if ( szuperprim_e(szam) ) { ++k; }
  }
  printf("%i", k);
  return 0;
}
```

```
bool prim_e (int);
bool szuperprim_e (int);
```

```
bool szuperprim_e (int x){
  while ( x ) {
    if ( !prim_e(x) ){
      return false;
    }
    x /= 10;
  }
  return true;
}
```

```
bool prim_e (int y){
  int i;
  if (y==0 || y==1) {return false;}
  for( i=2 ; i<=sqrt(y) ; ++i ){
    if ( y%i == 0 ){
      return false;
    }
  }
  return true;
}
```

2333

233

23

2



Generáljunk n véletlen számot a [0,10000) intervallumból. Hánynak prím a számjegyösszege?

```
int main(){
    int szam,k=0,n,i;
    scanf("%i", &n);
    srand(time(0)); // <stdlib.h>, <time.h>
    for( i=1 ; i<=n ; ++i){
        szam = rand() % 10000;
        if ( prim_e( szamjegyosszeg (szam) ) ) { ++k; }
    }
    printf("%i", k);
    return 0;
}
```

```
bool prim_e (int);
int szamjegyosszeg(int);
```

```
int szamjegyosszeg (int x){
    int s =0;
    while ( x ) {
        s += x%10;
        x /= 10;
    }
    return s;
}
```

```
bool prim_e (int y){
    int i;
    if (y==0 || y==1) {return false;}
    for( i=2 ; i<=sqrt(y) ; ++i ){
        if ( y%i == 0 ){
            return false;
        }
    }
    return true;
}
```





ÖSSZEFOGLALÓ

- Egymásba-ágyazott ciklusok
- TOP-DOWN programozás

- Szám-sorozatok:
 - generáljuk
 - szabály
 - véletlen
 - beolvassuk
 - billentyűzetről
 - állományból
 - ismert elemszám / végjelig
- Számpár-sorozatok

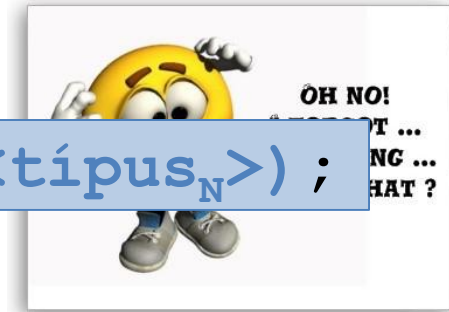
FÜGGVÉNYEK

- Mátrixok

- Prím-tesztelés technikája

Függvény DEKLARÁCIÓ

`<típus> <azonosító> (<típus1>, ..., <típusN>) ;`



HÍVÓ Függvény

Függvény HÍVÁS

`<változó> = <azonosító> (<kif1>, ..., <kifN>) ;`

`<típus> <azonosító> (<típus1> <név1>, ..., <típusn> <névn>) {`

o o o „forgatókönyv” o o o

`return <eredmény>;`

`}`

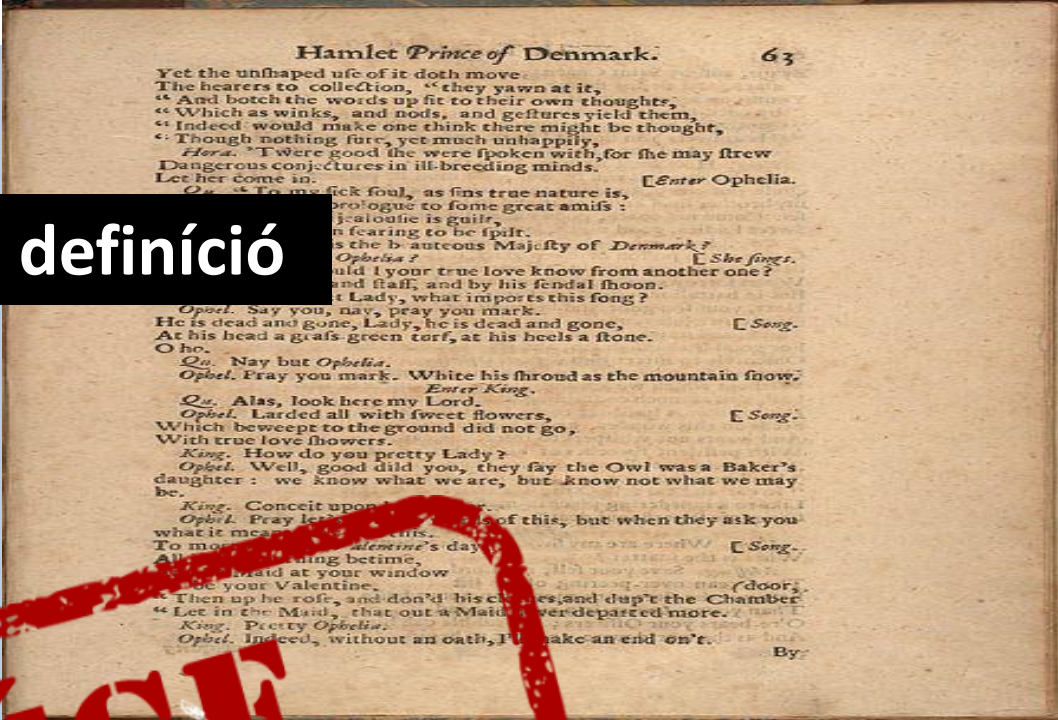
Függvény DEFINÍCIÓ



deklaráció

- szereplők
 - Claudius
 - Hamlet
 - Polonius
 - Horatio
 - Laertes
 - Gertrude
 - Ophelia
 - ...

definíció



VÉGE

- színészek
 - Makranczi Zalán
 - Szabó Kimmel Tamás
 - Rába Roland
 - Mátyássy Bence



dublőr