

Programozás C nyelven (6. ELŐADÁS)

Sapientia EMTE

2020-21



ELJÁRÁSOK: void-függvények

- Olvassunk be n számot a billentyűzetről, és írassuk ki a négyzeteiket a képernyőre.

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

```
3
7
49
11
121
5
25
—
```

ELJÁRÁSOK: void-függvények

```
void be_szamsor_ki_negyzetsor(int);
int main(){
    int n;
    scanf("%i", &n);
    be_szamsor_ki_negyzetsor(n);
    return 0;
}
void be_szamsor_ki_negyzetsor(int db){
    int i, szam;
    for( i=1 ; i <= db ; ++i){
        scanf("%i", &szam);
        printf("%i\n", szam*szam);
    }
}
```

```
3
7
49
11
121
5
25
—
```

ISMÉTLÉS: iteratív vs. rekurzív



PÉLDÁNYOK

1.

2.

3.

4.

5.

Kiosztott feladatok

5 fát!



4 fát!



3 fát!



2 fát!



1 fát!



ISMÉTLÉS: iteratív vs. rekurzív

- Ha egy függvény azt a feladatot kapja, hogy ismételten végezzen el bizonyos műveleteket, akkor két variáns közül választhat:
 - Ciklus révén felvállalja a sokszori végrehajtást (iteratív módszer);
 - Csak egyszeri végrehajtást vállal fel, és a többit átruházza egy „klónjára” (rekurzív módszer).
 - átruházza egy „klónjára” \Leftrightarrow **meghívja önmagát**

MIÉRT?

Hanoi tornyai: iteratív vs. rekurzív



```
void tohIterative(int num_of_disks, struct Stack *src, struct Stack *aux,
struct Stack *dest){
    int i, total_num_of_moves;
    char s = 'S', d = 'D', a = 'A';
    if (num_of_disks % 2 == 0){
        char temp = d;
        d = a;
        a = temp;
    }
    total_num_of_moves = pow(2, num_of_disks);
    for (i = num_of_disks; i >= 1; i--)
        push(src, i);
    for (i = 1; i <= total_num_of_moves; i++)
        if (i % 3 == 1)
            moveDisksBetweenTwoPoles(src, dest, s, d);
        else if (i % 3 == 2)
            moveDisksBetweenTwoPoles(src, aux, s, a);
        else if (i % 3 == 0)
            moveDisksBetweenTwoPoles(aux, dest, a, d);
}
```

```
void move(int n, int from, int to, int via){
    if (n > 0) {
        move(n - 1, from, via, to);
        Printf("%i >> %i\n", from, to);
        move(n - 1, via, to, from);
    }
}
```

ELEGÁNS

NEHÉZKES

```
void moveDisksBetweenTwoPoles(struct Stack *src,
struct Stack *dest, char s, char d){
    int pole1TopDisk = pop(src);
    int pole2TopDisk = pop(dest);
    if (pole1TopDisk == INT_MIN){
        push(src, pole2TopDisk);
    }
    moveDisk(s, d, pole1TopDisk);
    if (pole2TopDisk == INT_MIN){
        push(dest, pole1TopDisk);
    }
    moveDisk(s, d, pole2TopDisk);
}
```

Ültess fákat: iteratív

```
void ultess_fakat(int);  
int main(){  
    int n;  
    scanf("%i", &n);  
    ultess_fakat(n);  
    return 0;  
}
```



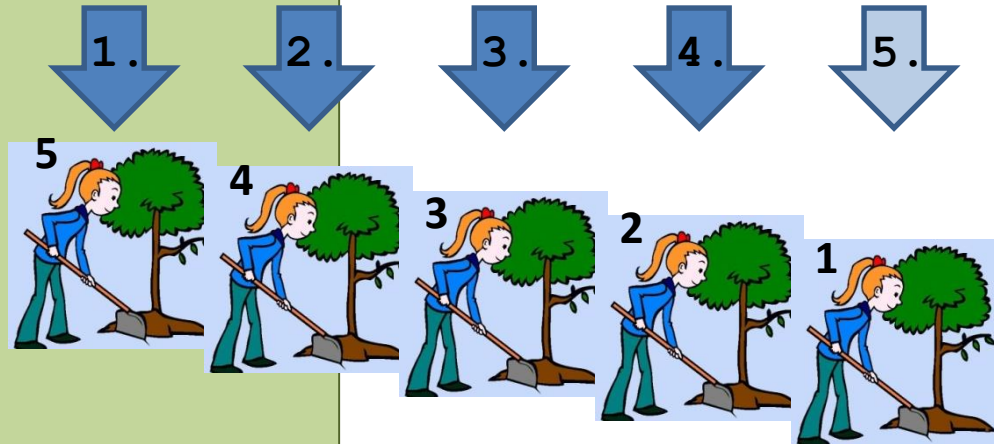
Hányszor
hívódik
meg a
függvény

```
void ultess_fakat(int db){  
    int i;  
    for( i=1 ; i <= db ; ++i){  
        <ültess el egy fát>  
    }  
}
```

Hány db/i -nek
történik
helyfoglalás

Ültess fákat: rekurzív

```
void ultess_fakat(int);  
int main(){  
    int n;  
    scanf("%i", &n);  
    ultess_fakat(n);  
    return 0;  
}
```



Hányszor hívódik meg a függvény

```
void ultess_fakat(int db){  
    if (db > 1){  
        <ültess el egy fát>  
        ultess_fakat(db-1);  
    }  
    else {<ültess el egy fát>}  
}
```

Hány db -nek történik helyfoglalás

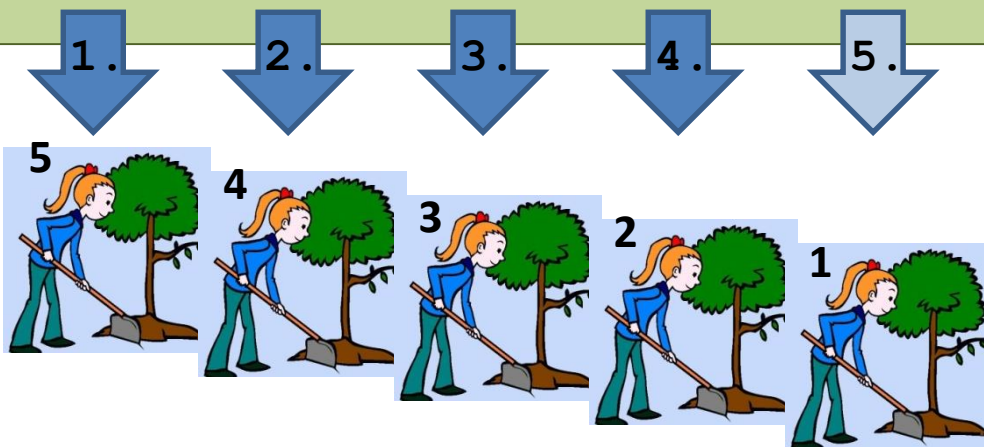
Rekurzív eljárások: SÉMA (1)

```
void R_E(<kapott_feladat_paramétereik>) {  
    if (<kapott_feladat_NEM_triviális>) {  
        <old_meg_a_felvallalt_sajat_részt>  
        R_E(<átruházott_feladat_paramétereik>);  
    }  
    else {  
        <old_meg_a_triviális_feladatot>  
    }  
}
```

Rekurzív-ág

Triviális-ág

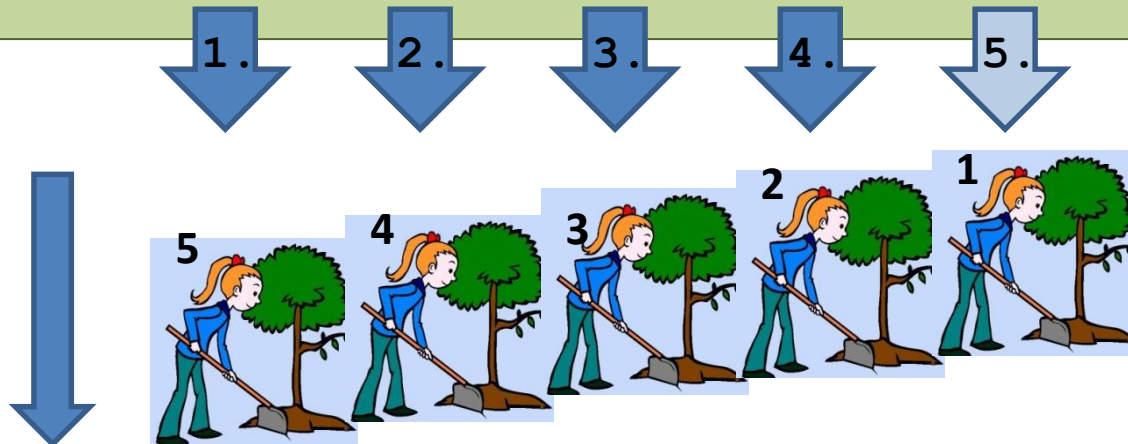
Hasonló,
egyszerűbb;
konvergál a
triviális
feladathoz



Melyik
példány ültet
először fát?

Rekurzív eljárások: SÉMA (2)

```
void R_E(<kapott_feladat_paramétereik>) {  
    if (<kapott_feladat_NEM_triviális>) {  
        R_E(<átruházott_feladat_paramétereik>);  
        <old_meg_a_felvállalt_saját_részt>  
    }  
    else {  
        <old_meg_a_triviális_feladatot>  
    }  
}
```



Melyik példány ültet először fát?

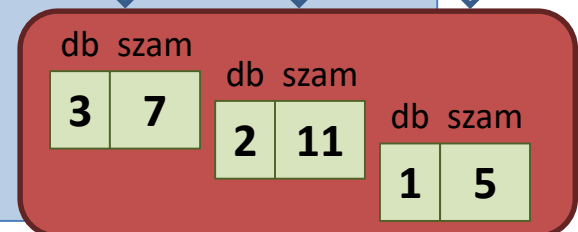
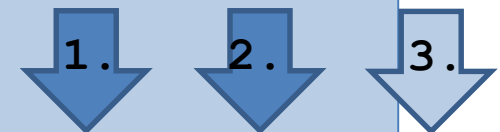
Rekurzív void-függvények (1)

```
void REKURZIV_be_szamsor_ki_negyzetsor(int);
int main(){
    int n;
    scanf("%i", &n);
    REKURZIV_be_szamsor_ki_negyzetsor(n);
    return 0;
}
void REKURZIV_be_szamsor_ki_negyzetsor(int db){
    int szam;
    if (db > 1){
        scanf("%i",&szam); printf("%i\n",szam*szam);
        REKURZIV_be_szamsor_ki_negyzetsor(db-1);
    }
    else {
        scanf("%i", &szam);
        printf("%i\n", szam*szam);
    }
}
```

3
7
49
11
121
5
25
–

Hányszor hívódik meg a függvény

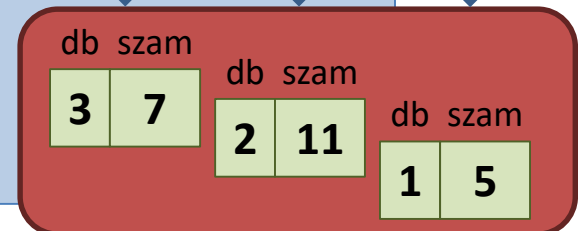
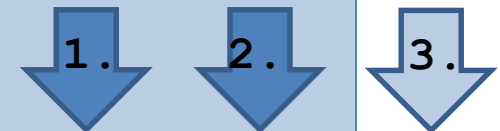
Hány db/szam-nak történik helyfoglalás



Rekurzív void-függvények (2)

```
void REKURZIV_be_szamsor_ki_negyzetsor(int) ;
int main() {
    int n;
    scanf("%i", &n);
    REKURZIV_be_szamsor_ki_negyzetsor(n);
    return 0;
}
void REKURZIV_be_szamsor_ki_negyzetsor(int db) {
    int szam;
    if (db > 1) {
        scanf("%i", &szam);
        REKURZIV_be_szamsor_ki_negyzetsor(db-1);
        printf("%i\n", szam*szam);
    }
    else {
        scanf("%i", &szam);
        printf("%i\n", szam*szam);
    }
}
```

3
7
11
5
25
121
49
—



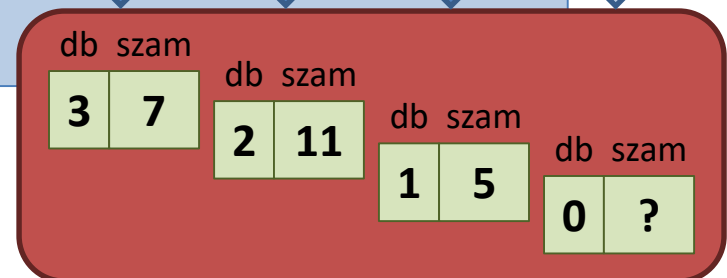
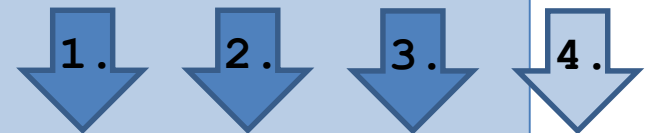
Rekurzív void-függvények (3)

```
void REKURZIV_be_szamsor_ki_negyzetsor(int);  
int main(){  
    int n;  
    scanf("%i", &n);  
    REKURZIV_be_szamsor_ki_negyzetsor(n);  
    return 0;  
}  
void REKURZIV_be_szamsor_ki_negyzetsor(int db){  
    int szam;  
    if (db > 0){  
        scanf("%i",&szam); printf("%i\n",szam*szam);  
        REKURZIV_be_szamsor_ki_negyzetsor(db-1);  
    }  
    else {;}  
}
```

```
3  
7  
49  
11  
121  
5  
25  
—
```

Hányszor hívódik meg a függvény

Hány db/szam-nak történik helyfoglalás



Rekurzív valódi-függvények (1)

Írj rekurzív függvényt, amely visszatéríti a paraméterként kapott természetes szám számjegyei szorzatát!

1. KÉRDÉS:

Hogyan vezethető vissza a feladat hasonló, egyszerűbb részfeladatra?

x számjegyszorzata
(52437)



x/10 számjegyszorzata
(5243)

Rekurzív valódi-függvények (2)

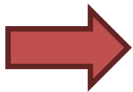
Írj rekurzív függvényt, amely visszatéríti a paraméterként kapott természetes szám számjegyei szorzatát!

2. KÉRDÉS:

Mikor tekintem úgy, hogy a feladat triviálissá redukálódott?

Ha elfogyott a szám

(52437)



(5243)



(524)



(52)



(5)



()

Rekurzív valódi-függvények (3)

Írj rekurzív függvényt, amely visszatéríti a paraméterként kapott természetes szám számjegyei szorzatát!

```
int szamjegyszorzat(int x) {  
    if (x > 0) {  
        ooo  
    }  
    else{ ooo }  
}
```

Rekurzív-ág

Triviális-ág

Rekurzív valódi-függvények (4)

Írj rekurzív függvényt, amely visszatéríti a paraméterként kapott természetes szám számjegyei szorzatát!

```
int szamjegyszorzat(int x) {  
    if (x > 0) {  
        ooo  
    }  
    else{ return 1; }  
}
```

Triviális
részfeladat
nyilvánvaló
megoldásának
return-elése

Rekurzív valódi-függvények (5)

Írj rekurzív függvényt, amely visszatéríti a paraméterként kapott természetes szám számjegyei szorzatát!

```
int szamjegyszorzat(int x) {  
    if (x > 0) {  
        int t = szamjegyszorzat(x/10);  
        ooo  
    }  
    else{ return 1; }  
}
```

Rekurzív hívás
révén kérem
tálcán az
átruházott rész
megoldását

Rekurzív valódi-függvények (6)

Írj rekurzív függvényt, amely visszatéríti a paraméterként kapott természetes szám számjegyei szorzatát!

```
int szamjegyszorzat(int x) {  
    if (x > 0) {  
        int t = szamjegyszorzat(x/10);  
        return t * (x%10);  
    }  
    else{ return 1; }  
}
```

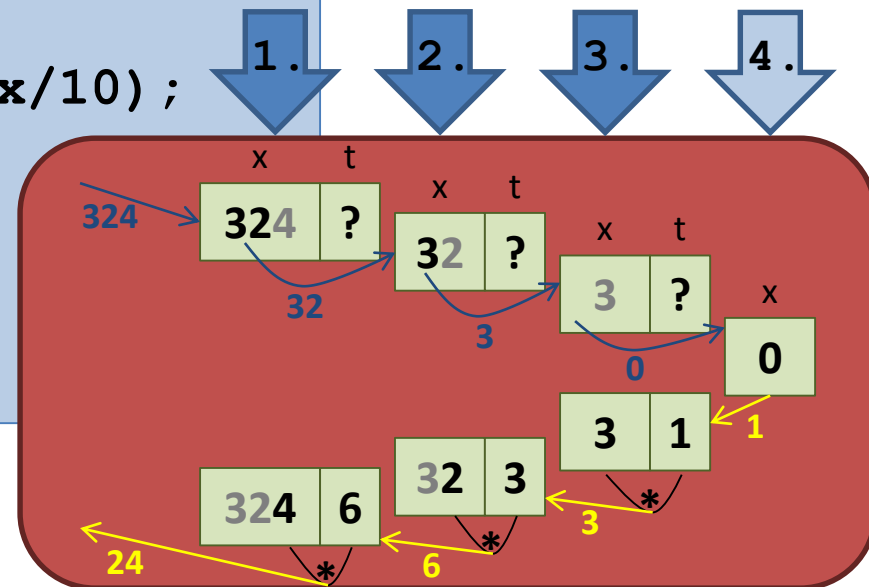
A bevállalt szorzás

Rekurzív valódi-függvények

```
int szamjegyszorzat(int);
int main(){
    int szam;
    scanf("%i", &szam);
    printf("%i", szamjegyszorzat(szam));
    return 0;
}
int szamjegyszorzat(int x){
    if (x > 0){
        int t = szamjegyszorzat(x/10);
        return t * x%10;
    }
    else{ return 1; }
}
```

324

24_





Összefoglalás (1)

- Minden jelentősebb feladat megoldása részfeladatok ismételt elvégzését feltételezi.
 - Ha ciklus-utasítás révén valósítjuk meg az ismétlést, akkor iteratív implementációról beszélünk.
 - Egy másik lehetőség a rekurzív megvalósítás.
- A rekurzív eljárások/függvények sajátossága, hogy meghívják önmagukat.
 - Olyan, mintha klónoznának magukból további példányokat.
 - Úgy is mondhatnánk, hogy a rekurzivitás mechanizmusa révén, az illető eljárásból/függvényből példányok sora jön létre, amelyek között szétesztődik a megoldandó feladat.



Összefoglalás (2)

- Mindegyik példány bevállal egy adott részt a kapott feladatból, a maradék részfeladatot pedig egy következő példányra ruházza át.
 - Mivel az átruházás egy klónra történik, ezért a kapott feladatnak és az átruházott részfeladatnak hasonlóknak kell lennie.
- A rekurzív implementálás kulcskérdése tehát az, hogy miként vezethető vissza a kapott feladat megoldása hasonló, egyre egyszerűbb részfeladatok megoldására.
 - E redukálási folyamat révén végül annyira leegyszerűsödik a feladat, hogy a sorvégi példány egészében be tudja vállalni.



Összefoglalás (3)

- Minden eljárás/függvény-példány hasonló feladatot kap, hogy megoldjon;
- A kapott feladat méretétől függ, hogy a kurrens példány rekurzívan fog viselkedni, vagy bevállalja a teljes megoldást;
 - Rekurzívan viselkedni azt jelenti, hogy önmaga meghívása által, a kapott feladat orozlánrészét (hasonló, egyszerűbb részfeladatot) átruházza egy következő példányra (a fennmaradt részt bevállalja);
- Egyszerűsített, általános forgatókönyv a kurrens példányra megfogalmazva.

Gyakori hibák (ID)

The screenshot shows the Code::Blocks IDE interface. The main editor window displays the following C++ code:

```
1 #include <iostream>
2 using namespace std;
3 int main()
4 {
5     for(int i = 0; i < 10; i++)
6     {
7         cout << endl;
8         for(int j = 0; j < 10; j++)
9         {
10            if (i < j)
11                cout << i + j;
12            else
13                cout << i * j;
14            //missing: }
15        }
16    return 0;
17    //expected '}' at end of input|
18 }
```

The error message in the 'Logs & others' panel is:

```
==== Build: Debug in gyakorihibak (compiler: GNU GCC Compiler) ====
C:\Users\dauid... In function 'int main()':
C:\Users\dauid... 17 error: expected '}' at end of input
```

The error message indicates a missing closing curly brace at the end of the input on line 17.

Gyakori hibák (ID)

The screenshot shows the Code::Blocks IDE interface. The main editor window displays the following C++ code:

```
1  #include <iostream>
2  using namespace std;
3  int main()
4  {
5      //a function-definition is not allowed here before '{' token
6      bool prim_e(unsigned szam)
7      {
8          return true;
9      }
10
11     cout << "Hello world!" << endl;
12     return 0;
13 }
14
```

The error message is visible in the "Build messages" window at the bottom:

```
=== Build: Debug in gyakorihibak (compiler: GNU GCC Compiler) ===
C:\Users\david... In function 'int main()':
C:\Users\david... 8 error: a function-definition is not allowed here before '{' token
=== Build failed: 1 error(s), 0 warning(s) (0 minute(s), 0 second(s)) ===
```

The IDE interface includes a menu bar (File, Edit, View, Search, Project, Build, Debug, Fortran, wxSmith, Tools, Tools+, Plugins, DoxyBlocks, Settings, Help), a toolbar, a project manager on the left, and a status bar at the bottom showing the current file path, window title, and system tray information.

Gyakori hibák (ID)

The screenshot shows the Code::Blocks IDE with a C++ program named `*main.cpp` open. The program defines a function `kisebb` that is intended to return the larger of two integers, `x` and `y`. However, the function contains a common error: it returns `0` instead of the larger value. The `main` function calls `kisebb(8, -4)`, which would output `0` due to the error.

```
1  #include <iostream>
2  using namespace std;
3  //function used as procedure. prints result,
4  //instead of returning it
5  int kisebb(int x, int y)
6  {
7      cout << "a kisebb szam: ";
8      if (x < y)
9          cout << x;
10     else
11         cout << y;
12     cout << endl;
13     return 0;
14 }
15 int main()
16 {
17     cout << kisebb(8, -4) << endl;
18     return 0;
19 }
```

The IDE interface includes a menu bar (File, Edit, View, Search, Project, Build, Debug, Fortran, wxSmith, Tools, Tools+, Plugins, DoxyBlocks, Settings, Help), a toolbar, and a status bar at the bottom showing the current file path, window title, and system tray information.

Gyakori hibák (ID)

The screenshot shows the Code::Blocks IDE interface. The main editor window displays the following C++ code:

```
1 #include <iostream>
2 using namespace std;
3
4 int main()
5 {
6     int a = 5, b = 3, c;
7     c = a + b
8     //expected ';' before 'cout' |
9     cout << c;
10    return 0;
11 }
12
```

The error message in the 'Build messages' window is:

```
=== Build: Debug in gyakorihibak (compiler: GNU GCC Compiler) ===
C:\Users\dauid... In function 'int main()':
C:\Users\dauid... 8 error: expected ';' before 'cout'
=== Build failed: 1 error(s), 0 warning(s) (0 minute(s), 0 second(s)) ===
```

The error message indicates a semicolon is missing before the `cout` statement on line 9.

Gyakori hibák (ID)

The screenshot shows the Code::Blocks IDE interface. The main editor window displays the following C++ code:

```
1  #include <iostream>
2  using namespace std;
3
4  int main()
5  {
6      int a, b;
7      cin >> a >> b;
8      if (a<b); //<---- ; after if
9          cout << a;
10     // 'else' without a previous 'if'
11     else
12         cout << b;
13
14     return 0;
15 }
16
```

The IDE's status bar at the bottom indicates the current position: Line 8, Column 14. The 'Logs & others' panel at the bottom shows the following error message:

```
==== Build: Debug in gyakorihibak (compiler: GNU GCC Compiler) ====
C:\Users\david... In function 'int main()':
C:\Users\david... 10 error: 'else' without a previous 'if'
==== Build failed: 1 error(s), 0 warning(s) (0 minute(s), 0 second(s)) ====
```

The Windows taskbar at the bottom shows the system tray with the date and time: 11:02 PM, 10/28/2015.

Gyakori hibák (ID)

The screenshot shows the Code::Blocks IDE with a C++ program in `*main.cpp`. The code is as follows:

```
1  #include <iostream>
2  using namespace std;
3
4  int main()
5  {
6      for (int i = 1; i < 10; i++); //<--- ; after for
7      {
8          //name lookup of 'i' changed for ISO 'for' scoping
9          int i2 = i*i;
10         cout << i2+1;
11     }
12
13     return 0;
14 }
15
```

The compilation log shows the following error:

```
=== Build: Debug in gyakorihibak (compiler: GNU GCC Compiler) ===
C:\Users\david... In function 'int main()':
C:\Users\david... 8 error: name lookup of 'i' changed for ISO 'for' scoping [-fpermissive]
C:\Users\david... 8 note: (if you use '-fpermissive' G++ will accept your code)
=== Build failed: 1 error(s), 0 warning(s) (0 minute(s), 0 second(s)) ===
```

The error message indicates that the variable `i` is used in the for loop condition but is not yet declared at that point due to C++11's 'for' scoping rules. The variable `i` is only declared inside the loop body, which is why the compiler reports a name lookup error.

Gyakori hibák (ID)

The screenshot shows the Code::Blocks IDE with a C++ program in `main.cpp`. The program includes `<iostream>` but does not use the `using namespace std;` directive. The `main` function contains two errors: `cin` is not declared in the scope, and `cout` is not declared in the scope. The IDE's error log at the bottom displays the following messages:

```
File          Line  Message
C:\Users\da... 9      error: 'cin' was not declared in this scope
C:\Users\da... 9      note: suggested alternative:
d:\program fil... 60     note: 'std::cin'
C:\Users\da... 12     error: 'cout' was not declared in this scope
C:\Users\da... 12     note: suggested alternative:
```

```
1  #include <iostream>
2  //missing: using namespace std;
3
4  int main()
5  {
6      int a;
7      //'cin' was not declared in this scope|
8      //suggested alternative: std::cin |
9      cin >> a;
10     //'cout' was not declared in this scope|
11     //suggested alternative: std::cout |
12     cout << 2*a;
13     return 0;
14 }
15
```

Gyakori hibák (ID)

The screenshot shows the Code::Blocks IDE with a C++ program named `main.cpp` open. The program contains several errors highlighted in red. The errors are:

- Line 7: `//variable 'std::ifstream fin' has initializer but incomplete type`
- Line 10: `error: 'sqrt' was not declared in this scope`

The program code is as follows:

```
1  #include <iostream>
2  //missing: #include <fstream>
3  //missing: #include <math.h>
4  using namespace std;
5  int main()
6  {
7      //variable 'std::ifstream fin' has initializer but incomplete type
8      ifstream fin("be.txt");
9      double d;
10     fin >> d;
11     //'sqrt' was not declared in this scope|
12     cout << sqrt(d);
13     return 0;
14 }
15
```

The bottom panel shows the 'Build messages' window with the following output:

```
=== Build: Debug in gyakorihibak (compiler: GNU GCC Compiler) ===
C:\Users\dauid... In function 'int main()':
C:\Users\dauid... 7 error: variable 'std::ifstream fin' has initializer but incomplete type
C:\Users\dauid... 10 error: 'sqrt' was not declared in this scope
=== Build failed: 2 error(s), 0 warning(s) (0 minute(s), 0 second(s)) ===
```