Változók hatálya

Kormányos Andor

Komplex Rendszerek Fizikája Tanszék

2020. szeptember 14.

Változók hatálya

A C nyelvben a változók hatályát a kapcsos zárójelek jelölik ki

- a függvényeken belül létrehozott változók *lokálisak*
- csak a függvényből látszanak
- ha a függvény egy másik függvényt hív, a lokális változót *át kell neki adni*

Ugyanez igaz a függvényparaméterekre

- a paraméterek csak a függvényen belülről érhetők el
- ugyanúgy működnek, mint a lokális változók

Változók deklarálhatók függvényeken kívül is

- *globális változók*
- ezek minden függvényből látszanak
- nem kell őket expliciten átadni függvényhíváskor
- a használatukat érdemes kerülni

Példa globális változóra

```
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  // some global variables
5  int a = 12, b = 23;
6
7  void fun1() {
8      a += 2;
9  }
10
11 void fun2() {
12     printf("%d\n", a + b);
13 }
14
15 int main()
16 {
17     fun1();
18     fun2();
19     return 0;
20 }
```

Globális változók

- a függvényeken kívül deklaráljuk

Példa globális változóra

```
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  // some global variables
5  int a = 12, b = 23;
6
7  void fun1() {
8      a += 2;
9  }
10
11 void fun2() {
12     printf("%d\n", a + b);
13 }
14
15 int main()
16 {
17     fun1();
18     fun2();
19     return 0;
20 }
```

Globális változók

- a függvényeken kívül deklaráljuk
- tetszőleges függvényből hivatkozhatók
- Figyelem!
  - általában utasítás nem szerepelhet függvényen kívül
  - kivéve globális változó inicializálása

Változók hatálya és a kapcsos zárójelek

```
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  int main()
5  {
6      int a = 5;
7      for (int i = 0; i < 10; i++)
8      {
9          int a = i + 1;
10         int c = i + 2;
11         printf("%d %d\n", a, c);
12     }
13     printf("%d\n", a);
14     printf("%d\n", c);
15     return 0;
16 }
```

A kapcsos zárójelek új hatályt jelölnek ki

- a {} blokkon belül új változók deklarálhatók
- nem látszanak a blokkon kívülről
- felüldefiniálják a külső változókat

Változók hatálya és a kapcsos zárójelek

```
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  int main()
5  {
6      int a = 5;
7      for (int i = 0; i < 10; i++)
8      {
9          int a = i + 1;
10         int c = i + 2;
11         printf("%d %d\n", a, c);
12     }
13     printf("%d\n", a);
14     printf("%d\n", c);
15     return 0;
16 }
```

A kapcsos zárójelek új hatályt jelölnek ki

- a {} blokkon belül új változók deklarálhatók
- nem látszanak a blokkon kívülről
- felüldefiniálják a külső változókat

Fontos észrevenni:

- két külön változó a néven!

Változók hatálya és a kapcsos zárójelek

```
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  int main()
5  {
6      int a = 5;
7      for (int i = 0; i < 10; i++)
8      {
9          int a = i + 1;
10         int c = i + 2;
11         printf("%d %d\n", a, c);
12     }
13     printf("%d\n", a);
14     printf("%d\n", c);
15     return 0;
16 }
```

A kapcsos zárójelek új hatályt jelölnek ki

- a {} blokkon belül új változók deklarálhatók
- nem látszanak a blokkon kívülről
- felüldefiniálják a külső változókat

Fontos észrevenni:

- két külön változó a néven!
- a c változó kívülről nem látszik

Változók hatálya és a kapcsos zárójelek

```
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  int main()
5  {
6      int a = 5;
7      for (int i = 0; i < 10; i++)
8      {
9          int a = i + 1;
10         int c = i + 2;
11         printf("%d %d\n", a, c);
12     }
13     printf("%d\n", a);
14     printf("%d\n", c);
15     return 0;
16 }
```

A kapcsos zárójelek új hatályt jelölnek ki

- a {} blokkon belül új változók deklarálhatók
- nem látszanak a blokkon kívülről
- felüldefiniálják a külső változókat

Fontos észrevenni:

- két külön változó a néven!
- a c változó kívülről nem látszik
- ezért a 13. sor fordítási hibát okoz