Paraméterek beolvasása

Kormányos Andor

Komplex Rendszerek Fizikája Tanszék

2022. szeptember 20.

Emlékeztető: az első program

Másodfokú egyenlet megoldóképlete

$$x_{1,2} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

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

int main() {
  double a, b, c;
  double d, r1, r2;
  a = 1.0;
  b = 3.0;
  c = 2.0;
  d = b * b - 4 * a * c;
  d = sqrt(d);
  r1 = (-b + d) / (2 * a);
  r2 = (-b - d) / (2 * a);
  printf("r1 = %f, r2 = %f\n", r1, r2);
  return 0;
}
```

Az előző program egy kicsit tömörebben

```
1   #include <stdio.h>
2   #include <math.h>
3
4   int main() {
5     double a = 1.0;
6     double b = 3.0;
7     double c = 2.0;
8     double d = sqrt(b * b - 4 * a * c);
9     double r1 = (-b + d) / (2 * a);
10    double r2 = (-b - d) / (2 * a);
11    printf("r1 = %f, r2 = %f\n", r1, r2);
12    return 0;
13  }
```

- a változók deklarálása összevonható az értékadással

Az előző program egy kicsit tömörebben

```
1   #include <stdio.h>
2   #include <math.h>
3
4   int main() {
5     double a = 1.0;
6     double b = 3.0;
7     double c = 2.0;
8     double d = sqrt(b * b - 4 * a * c);
9     double r1 = (-b + d) / (2 * a);
10    double r2 = (-b - d) / (2 * a);
11    printf("r1 = %f, r2 = %f\n", r1, r2);
12    return 0;
13  }
```

- a változók deklarálása összevonható az értékadással
- a függvények kifejezéseket is kaphatnak paraméterként
  - ilyenkor a kifejezés előbb kiértékelődik
  - a függvény a kapott eredményt kapja meg paraméterként

Az első program fő problémái

1. Az egyenlet együtthatói nem paraméterek, hanem konstansok
   - ez úgy mondjuk, hogy az értékek “hard code”-olva vannak
   - az ilyet mindig kerülni kell, kivéve ha valóban konstansokról van szó

Az első program fő problémái

1. Az egyenlet együtthatói nem paraméterek, hanem konstansok
   - ez úgy mondjuk, hogy az értékek "hard code"-olva vannak
   - az ilyet mindig kerülni kell, kivéve ha valóban konstansokról van szó

2. Hogyan lehetne az együtthatókat paraméterként beadni?
   - parancssori argumentumként
   - billentyűzetről beolvasva
   - fájlból beolvasva

Parancssori paraméterek beolvasása

A parancssori argumentumokat a konzolablakban szeretnénk megadni futtatáskor, pl.:

```
1  $ ./roots 1 3 2
```

Parancssori paraméterek beolvasása

A parancssori argumentumokat a konzolablakban szeretnénk megadni futtatáskor, pl.:

```
1  $ ./roots 1 3 2
```

Ezeket az argumentumokat a `main` függvény paramétereiként kapjuk meg

- vigyázat: minden paraméter szövegként van kezelve
- át kell alakítani a szöveget számmá
- egyelőre itt a kész megoldás: `atof` függvény

Parancssori paraméterek beolvasása

A parancssori argumentumokat a konzolablakban szeretnénk megadni futtatáskor, pl.:

```
$ ./roots 1 3 2
```

Ezeket az argumentumokat a `main` függvény paramétereiként kapjuk meg

- vigyázat: minden paraméter szövegként van kezelve
- át kell alakítani a szöveget számmá
- egyelőre itt a kész megoldás: `atof` függvény

```c
#include <stdlib.h>
#include <stdio.h>
#include <math.h>

int main(int argc, char* argv[]) {
  double a = atof(argv[1]);
  double b = atof(argv[2]);
  double c = atof(argv[3]);
  double d = sqrt(b * b - 4 * a * c);
  double r1 = (-b + d) / (2 * a);
  double r2 = (-b - d) / (2 * a);
  printf("r1 = %f, r2 = %f\n", r1, r2);
  return 0;
}
```

- Az `argc` paraméter a parancssori paraméterek számát tartalmazza
- Az `argv` paraméterben szövegként kapjuk meg a paramétereket
  - az első (0. indexű) paraméter mindig a futó program neve + path

Parancssori paraméterek beolvasása

A parancssori argumentumokat a konzolablakban szeretnénk megadni futtatáskor, pl.:

```
$ ./roots 1 3 2
```

Ezeket az argumentumokat a `main` függvény paramétereiként kapjuk meg

- vigyázat: minden paraméter szövegként van kezelve
- át kell alakítani a szöveget számmá
- egyelőre itt a kész megoldás: `atof` függvény

```c
#include <stdlib.h>
#include <stdio.h>
#include <math.h>

int main(int argc, char* argv[]) {
  double a = atof(argv[1]);
  double b = atof(argv[2]);
  double c = atof(argv[3]);
  double d = sqrt(b * b - 4 * a * c);
  double r1 = (-b + d) / (2 * a);
  double r2 = (-b - d) / (2 * a);
  printf("r1 = %f, r2 = %f\n", r1, r2);
  return 0;
}
```

- Az `argc` paraméter a parancssori paraméterek számát tartalmazza
- Az `argv` paraméterben szövegként kapjuk meg a paramétereket
  - az első (0. indexű) paraméter mindig a futó program neve + path

Látni fogjuk, hogy az `argv`-t úgy a helyes használni, ha előbb leellenőriztük az `argc` értékét!