Feltételes elágazások

Kormányos Andor

Komplex Rendszerek Fizikája Tanszék

2022. szeptember 20.

Feltételes elágazások, az if utasítás

```
1   #include <stdio.h>
2   #include <stdlib.h>
3   #include <math.h>
4
5   int main(int argc, char* argv[]) {
6     if (argc < 4) {
7       printf("Not enough arguments.\n");
8       exit(-1);
9     }
10
11    double a = atof(argv[1]);
12    double b = atof(argv[2]);
13    double c = atof(argv[3]);
14    double d = b * b - 4 * a * c;
15
16    if (d < 0) {
17      printf("No real solution.\n");
18      exit(-1);
19    } else {
20      d = sqrt(d);
21      double r1 = (-b - d) / 2 / a;
22      double r2 = (-b + d) / 2 / a;
23      printf("r1 = %f, r2 = %f\n", r1, r2);
24    }
25
26    return 0;
27  }
```

- Az if (...) utasítás zárójelében egy feltétel szerepel
  - logikai vagy aritmetikai kifejezés

Feltételes elágazások, az `if` utasítás

```
1   #include <stdio.h>
2   #include <stdlib.h>
3   #include <math.h>
4
5   int main(int argc, char* argv[]) {
6     if (argc < 4) {
7       printf("Not enough arguments.\n");
8       exit(-1);
9     }
10
11    double a = atof(argv[1]);
12    double b = atof(argv[2]);
13    double c = atof(argv[3]);
14    double d = b * b - 4 * a * c;
15
16    if (d < 0) {
17      printf("No real solution.\n");
18      exit(-1);
19    } else {
20      d = sqrt(d);
21      double r1 = (-b - d) / 2 / a;
22      double r2 = (-b + d) / 2 / a;
23      printf("r1 = %f, r2 = %f\n", r1, r2);
24    }
25
26    return 0;
27  }
```

- Az `if (...)` utasítás zárójelében egy feltétel szerepel
  - logikai vagy aritmetikai kifejezés
- Ha feltétel teljesül (értéke nem nulla), akkor az `if` utáni `{ ... }` block fut le

Feltételes elágazások, az `if` utasítás

```
1   #include <stdio.h>
2   #include <stdlib.h>
3   #include <math.h>
4
5   int main(int argc, char* argv[]) {
6     if (argc < 4) {
7       printf("Not enough arguments.\n");
8       exit(-1);
9     }
10
11    double a = atof(argv[1]);
12    double b = atof(argv[2]);
13    double c = atof(argv[3]);
14    double d = b * b - 4 * a * c;
15
16    if (d < 0) {
17      printf("No real solution.\n");
18      exit(-1);
19    } else {
20      d = sqrt(d);
21      double r1 = (-b - d) / 2 / a;
22      double r2 = (-b + d) / 2 / a;
23      printf("r1 = %f, r2 = %f\n", r1, r2);
24    }
25
26    return 0;
27  }
```

- Az `if (...)` utasítás zárójelében egy feltétel szerepel
  - logikai vagy aritmetikai kifejezés
- Ha feltétel teljesül (értéke nem nulla), akkor az `if` utáni `{ ... }` block fut le
- Ha a feltétel nem teljesült, az `else` ág fut le
  - az `else` ág nem kötelező

Feltételes elágazások, az `if` utasítás

```
1   #include <stdio.h>
2   #include <stdlib.h>
3   #include <math.h>
4
5   int main(int argc, char* argv[]) {
6     if (argc < 4) {
7       printf("Not enough arguments.\n");
8       exit(-1);
9     }
10
11    double a = atof(argv[1]);
12    double b = atof(argv[2]);
13    double c = atof(argv[3]);
14    double d = b * b - 4 * a * c;
15
16    if (d < 0) {
17      printf("No real solution.\n");
18      exit(-1);
19    } else {
20      d = sqrt(d);
21      double r1 = (-b - d) / 2 / a;
22      double r2 = (-b + d) / 2 / a;
23      printf("r1 = %f, r2 = %f\n", r1, r2);
24    }
25
26    return 0;
27  }
```

- Az `if (...)` utasítás zárójelében egy feltétel szerepel
  - logikai vagy aritmetikai kifejezés
- Ha feltétel teljesül (értéke nem nulla), akkor az `if` utáni `{ ... }` block fut le
- Ha a feltétel nem teljesült, az `else` ág fut le
  - az `else` ág nem kötelező
- `exit(-1);` függvényhívás
  - ez egy speciális függvény, ami azonnal kilép a programból
  - a nem nulla argumentummal hibát lehet jelezni az operációs rendszer felé

Mi történik itt?

Az előbbi program egy részlete:

```
1
2   int main(int argc, char* argv[]) {
3     if (argc < 4) {
4       printf("Not enough arguments.\n");
5       exit(-1);
6     }
```

- Összesen 3 bemenő paraméterre van szükségünk (a, b, c)
- Azt szeretnénk, ha a program ellenőrizné futáskor, hogy van-e elég bemenő paraméter
- a programnak átadott paraméterek száma az *argc*-ben tárolódik
- *argc* értéke minimum 1 (a program neve)
- ha nincs elég bemenő paraméter, akkor a futás leáll `exit(-1)` hatására

Milyen sorrendben hajtódik végre a program?

1. Az utasítások felülről lefelé hajtódnak végre
2. A program futása a `main` függvény első sorával indul
   - ekkor a parancssori argumentumok már fel lettek dolgozva, az `argv` értéke fel van töltve
3. Ha a program `if`-hez ér, akkor három dolog történhet
   - ha a feltétel teljesül, akkor a főág fut le
   - ha a feltétel nem teljesül, de van `else` ág, akkor az fut le
   - ha a feltétel nem teljesül, és nincsen `else` ág, akkor a program az `if` főágát átugorja, és az `if` utáni első utasításnál folytatódik

Többágú elágazás egymásba ágyazott `if`-ekkel

```
1   #include <stdio.h>
2   #include <stdlib.h>
3   #include <math.h>
4
5   int main(int argc, char* argv[]) {
6     if (argc < 4) {
7       printf("Not enough arguments.\n");
8       return -1;
9     }
10
11    double a = atof(argv[1]);
12    double b = atof(argv[2]);
13    double c = atof(argv[3]);
14    double d = b * b - 4 * a * c;
15
16    if (d < 0) {
17      printf("No real solution.\n");
18      return -1;
19    } else if (d == 0) {
20      double r = -b / 2 / a;
21      printf("r = %f\n", r);
22    } else {
23      d = sqrt(d);
24      double r1 = (-b - d) / 2 / a;
25      double r2 = (-b + d) / 2 / a;
26      printf("r1 = %f, r2 = %f\n", r1, r2);
27    }
28
29    return 0;
30  }
```

- Az `if(...)` és az `else` után egy-egy utasításblokk jön
- de lehet egy másik `else if` utasítás is