Struktúrák

Mi az a struktúra?

A struktúra (structure) egy felhasználó által definiált adattípus, amely különböző típusú változókat (adatokat) képes egy név alatt összefogni. Ez hasonló az osztályhoz más nyelvekben, csak egyszerűbb, mert nincs benne metódus.

Célja: összetett adatok (pl. hallgató, autó, pont, dátum stb.) logikai egységbe foglalása.

Struktúra definiálása

A struktúra típust a struct kulcsszóval hozzuk létre:

```
struct Szemely {
    char nev[50];
    int eletkor;
    float magassag;
};
```

Ezután létrehozhatunk belőle változókat:

```
struct Szemely ember1, ember2;
```

VAGY egyből definiálhatjuk és példányosíthatjuk:

```
struct Szemely {
    char nev[50];
    int eletkor;
    float magassag;
} ember1, ember2;
```

Típusnév (typedef)

A typedef segítségével egyszerűbb nevet adhatunk a struktúrának:

```
typedef struct {
    char nev[50];
    int eletkor;
    float magassag;
} Szemely;
```

```
Szemely ember1, ember2;
//Így már nem kell mindenhol struct kulcsszót írni.
```

Struktúratagok elérése

A tagokat a pont (.) operátorral érjük el:

```
strcpy(ember1.nev, "Kovacs Bela");
ember1.eletkor = 25;
ember1.magassag = 1.82;
```

Ha mutatóval dolgozunk, akkor a nyíl (→) operátort használjuk:

```
Szemely *ptr = &ember1;
printf("%s %d %.2f", ptr->nev, ptr->eletkor, ptr->magassag);
```

Struktúrák átadása függvényeknek

Érték szerint:

```
void kiir(Szemely s) {
    printf("%s (%d ev)\n", s.nev, s.eletkor);
}
```

Mutatóval (másképpen):

```
void kiir(const Szemely *s) {
    printf("%s (%d ev)\n", s->nev, s->eletkor);
}
```

Beágyazott struktúrák

Egy struktúra tartalmazhat másik struktúrát is:

```
typedef struct {
    int ev, honap, nap;
} Datum;

typedef struct {
    char nev[50];
    Datum szuletes;
} Hallgato;
```

Struktúra adatainak bekérése és kiírása:

```
#include <stdio.h>

typedef struct{
    char nev[30];
    int kor;
    double magassag;
}EMBER;

int main()
{
    EMBER e;
    int osszeg = 0;
    double atlag = 0;
    double max = 0;
    for(int i = 0; i < 3; i++){
        printf("Adja meg az ember nevét: ");
        fgets(e.nev, sizeof(e.nev), stdin);
        printf("Adja meg a korát: ");
        scanf("%d", &e.kor);
        printf("Magassága: ");
        scanf("%lf", &e.magassag);
        osszeg += e.kor;
        if(max < e.magassag){
            max = e.magassag;
        }
    }
    atlag = (double)osszeg / 3;
    printf("\n\nA korok átlaga: %lf", atlag);
    printf("\nLegnagyobb magassag: %lf", max);
}
```

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

struct DIAK{
    int id;
    char nev[50];
    double atlag;
};

int main()
{
   struct DIAK d1;
```

```
   d1.id = 1;
   strcpy(d1.nev, "Asd Feri");
   d1.atlag = 4.8;
   //d1 = {0,"sdg asg", 5.0};
   struct DIAK diakok[2] = {
       {2, "Jkl Éva", 4.4},
       {3, "Qwerty Béla", 3.6}
   };
   printf("ID: %d\n", d1.id);
   printf("Név: %s\n", d1.nev);
   printf("Átlag: %lf\n\n", d1.atlag);
   for(int i = 0; i < 2; i++){
        printf("ID: %d\n", diakok[i].id);
        printf("Név: %s\n", diakok[i].nev);
        printf("Átlag: %lf\n", diakok[i].atlag);
   }
}
```

Gyakorlás

1. Készíts egy struktúrát, ami egy pontot (x, y) tárol, és írja ki a koordinátáit!

```
#include <stdio.h>

struct Pont {
    int x;
    int y;
};

int main() {
    struct Pont p = {3, 7};
    printf("A pont koordinátai: (%d, %d)\n", p.x, p.y);
    return 0;
}
```

2.Kérj be 3 személy nevét és életkorát, majd írd ki őket!

```
#include <stdio.h>

typedef struct {
    char nev[50];
    int eletkor;
} Szemely;

int main() {
    Szemely emberek[3];
```

```
    for (int i = 0; i < 3; i++) {
        printf("%d. szemely neve: ", i+1);
        scanf(" %[^\n]", emberek[i].nev);
        printf("%d. szemely eletkora: ", i+1);
        scanf("%d", &emberek[i].eletkor);
    }

    printf("\n--- Adatok ---\n");
    for (int i = 0; i < 3; i++) {
        printf("%s (%d ev)\n", emberek[i].nev, emberek[i].eletkor);
    }
    return 0;
}
```

3. Írj függvényt, ami egy személy adatait kiírja!

```
#include <stdio.h>

typedef struct {
    char nev[50];
    int kor;
} Szemely;

void kiir(const Szemely *s) {
    printf("%s - %d ev\n", s->nev, s->kor);
}

int main() {
    Szemely ember = {"Kiss Anna", 22};
    kiir(&ember);
    return 0;
}
```

4. Készíts struktúrát, ahol egy hallgató neve és születési dátuma van eltárolva!

```
#include <stdio.h>

typedef struct {
    int ev, honap, nap;
} Datum;

typedef struct {
    char nev[50];
    Datum szuletes;
} Hallgato;

int main() {
```

```
    Hallgato h = {"Nagy Peter", {2002, 4, 15}};
    printf("%s szuletett: %d.%02d.%02d\n",
           h.nev, h.szuletes.ev, h.szuletes.honap, h.szuletes.nap);
    return 0;
}
```

5. Adj meg 3 személyt, és írd ki, ki a legidősebb!

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

typedef struct {
    char nev[50];
    int kor;
} Szemely;

int main() {
    Szemely emberek[3] = {
        {"Kovacs Bela", 34},
        {"Kiss Anna", 28},
        {"Toth Eva", 41}
    };

    int maxIndex = 0;
    for (int i = 1; i < 3; i++) {
        if (emberek[i].kor > emberek[maxIndex].kor)
            maxIndex = i;
    }

    printf("A legidosebb: %s (%d ev)\n",
           emberek[maxIndex].nev, emberek[maxIndex].kor);
    return 0;
}
```

6. Készíts programot, ami 5 diák 3 tantárgyból kapott jegyeit tárolja és minden diáknak kiszámolja az átlagát!

```
#include <stdio.h>

#define N 5
#define M 3

typedef struct {
    char nev[30];
    int jegyek[M];
    float atlag;
} Diak;
```

```
int main() {
    Diak diakok[N] = {
        {"Anna", {4, 5, 5}},
        {"Bela", {3, 4, 4}},
        {"Csilla", {5, 5, 5}},
        {"David", {2, 3, 4}},
        {"Erika", {4, 4, 3}}
    };

    for (int i = 0; i < N; i++) {
        int osszeg = 0;
        for (int j = 0; j < M; j++)
            osszeg += diakok[i].jegyek[j];
        diakok[i].atlag = (float)osszeg / M;
    }

    printf("--- Diakok atlaga ---\n");
    for (int i = 0; i < N; i++) {
        printf("%s: %.2f\n", diakok[i].nev, diakok[i].atlag);
    }

    return 0;
}
```

7. Definiálj egy Matrix struktúrát, amely tárol egy 3×3-as mátrixot, és készíts függvényt, ami a mátrix transzponáltját számolja ki!

```
#include <stdio.h>

#define SIZE 3

typedef struct {
    int adat[SIZE][SIZE];
} Matrix;

Matrix transzponalt(Matrix m) {
    Matrix t;
    for (int i = 0; i < SIZE; i++)
        for (int j = 0; j < SIZE; j++)
            t.adat[i][j] = m.adat[j][i];
    return t;
}

void kiir(Matrix m) {
    for (int i = 0; i < SIZE; i++) {
        for (int j = 0; j < SIZE; j++)
            printf("%d ", m.adat[i][j]);
        printf("\n");
    }
```

```
}

int main() {
    Matrix A = {{{1,2,3}, {4,5,6}, {7,8,9}}};
    printf("Eredeti mátrix:\n");
    kiir(A);
    printf("\nTranszponált:\n");
    Matrix T = transzponalt(A);
    kiir(T);
    return 0;
}
```

8. Tárolj 5 darab 2D pontot egy struktúra tömbben, és határozd meg, melyik két pont van egymástól legmesszebb!

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

#define N 5

typedef struct {
    float x, y;
} Pont;

float tav(Pont a, Pont b) {
    return sqrt((a.x - b.x)*(a.x - b.x) + (a.y - b.y)*(a.y - b.y));
}

int main() {
    Pont pontok[N] = {
        {0, 0}, {3, 4}, {5, 1}, {2, 8}, {9, 9}
    };

    float maxTav = 0;
    int p1 = 0, p2 = 1;

    for (int i = 0; i < N; i++) {
        for (int j = i + 1; j < N; j++) {
            float d = tav(pontok[i], pontok[j]);
            if (d > maxTav) {
                maxTav = d;
                p1 = i;
                p2 = j;
            }
        }
    }

    printf("Legnagyobb tavolsag: %.2f a (%g,%g) es (%g,%g) kozott.\n",
           maxTav, pontok[p1].x, pontok[p1].y, pontok[p2].x, pontok[p2].y);
```

```
    return 0;
}
```

9. Tölts fel 4 diák adatait 4 jeggyel (mátrix formában), majd minden diákhoz írd ki a legjobb és legrosszabb jegyet!

```
#include <stdio.h>

#define N 4
#define M 4

typedef struct {
    char nev[30];
    int jegyek[M];
} Diak;

int main() {
    Diak diakok[N] = {
        {"Anna", {5, 4, 3, 5}},
        {"Bela", {2, 3, 4, 4}},
        {"Csilla", {5, 5, 5, 4}},
        {"David", {3, 2, 3, 2}}
    };

    for (int i = 0; i < N; i++) {
        int min = diakok[i].jegyek[0];
        int max = diakok[i].jegyek[0];
        for (int j = 1; j < M; j++) {
            if (diakok[i].jegyek[j] < min) min = diakok[i].jegyek[j];
            if (diakok[i].jegyek[j] > max) max = diakok[i].jegyek[j];
        }
        printf("%s: legjobb %d, legrosszabb %d\n", diakok[i].nev, max, min);
    }

    return 0;
}
```

10. Készíts programot, ami két 3×3-as mátrixot ad össze és kiírja az eredményt.

```
#include <stdio.h>

#define N 3

typedef struct {
    int adat[N][N];
} Matrix;
```

```
int main() {
    Matrix A = {{{1,2,3},{4,5,6},{7,8,9}}};
    Matrix B = {{{9,8,7},{6,5,4},{3,2,1}}};
    Matrix C; // eredmény mátrix

    // Mátrixok összeadása
    for (int i = 0; i < N; i++) {
        for (int j = 0; j < N; j++) {
            C.adat[i][j] = A.adat[i][j] + B.adat[i][j];
        }
    }

    // Eredmény kiírása
    printf("Eredmeny:\n");
    for (int i = 0; i < N; i++) {
        for (int j = 0; j < N; j++) {
            printf("%d ", C.adat[i][j]);
        }
        printf("\n");
    }

    return 0;
}
```

From:
https://edu.iit.uni-miskolc.hu/ - **Institute of Information Science - University of Miskolc**

Permanent link:
https://edu.iit.uni-miskolc.hu/tanszek:oktatas:szamitastechnika:strukturak

Last update: **2025/11/18 18:26**