Feladat 1

Írjon egy “int oszthato(int*, int)” nevű függvényt, amely átvesz egy n elemű tömböt és méretét majd visszaadja hány darab 3-mal osztható van benne.

```
int oszthato(int *tomb, int n) {
    int db = 0;
    for (int i = 0; i < n; i++) {
        if (tomb[i] % 3 == 0) db++;
    }
    return db;
}
```

Feladat 2

Írjon egy atlag() nevű függvényt, amely a paraméterében megadott két float típusú paraméter értékének átlagát adja vissza.

```
float atlag(float a, float b) {
    return (a + b) / 2;
}
```

Feladat 3

Írjon egy atlag() nevű függvényt, amely egy 10 elemű int vektor átlagát adja vissza.

```
float atlag(int v[10]) {
    int osszeg = 0;
    for (int i = 0; i < 10; i++) osszeg += v[i];
    return osszeg / 10.0;
}
```

Feladat 4

Írjon egy max() nevű függvényt, amely egy 10 elemű int vektor legnagyobb elemének értékét adja vissza.

```
int max(int v[10]) {
    int m = v[0];
    for (int i = 1; i < 10; i++)
        if (v[i] > m) m = v[i];
    return m;
}
```

Feladat 5

Írjon egy min() nevű függvényt, amely egy 10 elemű int vektor legkisebb elemének értékét adja vissza.

```
int min(int v[10]) {
    int m = v[0];
    for (int i = 1; i < 10; i++)
        if (v[i] < m) m = v[i];
    return m;
}
```

Feladat 6

Írjon egy min() nevű függvényt, amely egy 10 elemű int vektor legkisebb elemének indexét adja vissza.

```
int min(int v[10]) {
    int idx = 0;
    for (int i = 1; i < 10; i++)
        if (v[i] < v[idx]) idx = i;
    return idx;
}
```

Feladat 7

Írjon egy max() nevű függvényt, amely egy 10 elemű int vektor legnagyobb elemének indexét adja vissza.

```
int max(int v[10]) {
    int idx = 0;
    for (int i = 1; i < 10; i++)
        if (v[i] > v[idx]) idx = i;
    return idx;
}
```

Feladat 8

Írjon egy fordit() nevű függvényt, amely egy 5 elemű int vektor elemeit megfordítja.

```
void fordit(int v[5]) {
    for (int i = 0; i < 5/2; i++) {
```

```
        int tmp = v[i];
        v[i] = v[4-i];
        v[4-i] = tmp;
    }
}
```

Feladat 9

Írjon egy “int* osszead(int*, int*, int)” nevű függvényt, amely átvesz két azonos méretű vektort és a méretüket, majd visszaadja a két vektor összegét tartalmazó vektort.

```
#include <stdlib.h>
int* osszead(int *a, int *b, int n) {
    int *c = (int*)calloc(n, sizeof(int));
    for (int i = 0; i < n; i++) {
        c[i] = a[i] + b[i];
    }
    return c;
}
```

Feladat 10

Írjon egy “int negyzetszam(int)” nevű függvényt, amely átvesz egy számot és ha az négyzetszám akkor 1-et ad vissza, egyébként 0-t.

```
#include <math.h>
int negyzetszam(int x) {
    if (x < 0) return 0;
    int gyok = (int)sqrt(x);
    return gyok * gyok == x;
}
```

Feladat 11

Írjon egy “void csere(int*, int*)” nevű függvényt, amely átvesz két számot és megcseréli az értéküket.

```
void csere(int *a, int *b) {
    int tmp = *a;
    *a = *b;
    *b = tmp;
}
```

Feladat 12

Írjon egy “int szorzat(int*, int)” nevű függvényt, amely átvesz egy vektort és a méretét, majd az elemeinek szorzatát adja vissza.

```
int szorzat(int *t, int n) {
    int s = 1;
    for (int i = 0; i < n; i++) s *= t[i];
    return s;
}
```

Feladat 13

Írjon egy “int* negyzet(int*, int)” nevű függvényt, amely átvesz egy vektort és a méretét, majd az elemeinek négyzetét tartalmazó vektort adja vissza.

```
#include <stdlib.h>
int* negyzet(int *t, int n) {
    int *res = (int*)calloc(n, sizeof(int));
    for (int i = 0; i < n; i++) {
        res[i] = t[i] * t[i];
    }
    return res;
}
```

Feladat 14

Írjon egy “int csak_paros(int*, int)” nevű függvényt, amely átvesz egy vektort és a méretét. Ha csak páros számokat tartalmaz a vektor, akkor 1-et, egyébként 0-t ad vissza.

```
int csak_paros(int *t, int n) {
    for (int i = 0; i < n; i++) {
        if (t[i] % 2 != 0) return 0;
    }
    return 1;
}
```

Feladat 15

Írjon egy “int csak_paratlan(int*, int)” nevű függvényt, amely átvesz egy vektort és a méretét. Ha csak páratlan számokat tartalmaz a vektor, akkor 1-et, egyébként 0-t ad vissza.

```
int csak_paratlan(int *t, int n) {
    for (int i = 0; i < n; i++) {
        if (t[i] % 2 == 0) return 0;
    }
    return 1;
}
```

Feladat 16

Írjon egy “int osszeg_paratlan(int*, int)” nevű függvényt, amely visszaadja a paraméterként kapott tömb páratlan elemeinek összegét.

```
int osszeg_paratlan(int *t, int n) {
    int sum = 0;
    for (int i = 0; i < n; i++) {
        if (t[i] % 2 != 0) sum += t[i];
    }
    return sum;
}
```

Feladat 17

Írjon egy “int elso_negativ_index(int*, int)” nevű függvényt, amely visszaadja az első negatív elem indexét, ha nincs ilyen, akkor -1-et.

```
int elso_negativ_index(int *t, int n) {
    for (int i = 0; i < n; i++) {
        if (t[i] < 0) return i;
    }
    return -1;
}
```

Feladat 18

Írjon egy “void feltolt_szammal(int*, int, int)” nevű függvényt, amely a paraméterként kapott tömb minden elemét ugyanarra az értékre állítja.

```
void feltolt_szammal(int *t, int n, int ertek) {
    for (int i = 0; i < n; i++) {
        t[i] = ertek;
    }
}
```

Feladat 19

Írjon egy “double paros_atlag(const int*, int)” nevű függvényt, amely visszaadja a tömb páros elemeinek átlagát (ha nincs páros elem, az átlag legyen 0).

```
double paros_atlag(const int *t, int n) {
    int sum = 0, db = 0;
    for (int i = 0; i < n; i++) {
        if (t[i] % 2 == 0) {
            sum += t[i];
            db++;
        }
    }
    if (db == 0) return 0;
    return (double)sum / db;
}
```

Feladat 20

Írjon egy “void duplaz(int*, int)” nevű függvényt, amely a tömb minden elemét megduplázza.

```
void duplaz(int *t, int n) {
    for (int i = 0; i < n; i++) {
        t[i] *= 2;
    }
}
```

Feladat 21

Írjon egy “int monoton_novekvo(const int*, int)” nevű függvényt, amely 1-et ad vissza, ha a tömb szigorúan növekvő, egyébként 0-t.

```
int monoton_novekvo(const int *t, int n) {
    for (int i = 1; i < n; i++) {
        if (t[i] <= t[i-1]) return 0;
    }
    return 1;
}
```

Feladat 22

Írjon egy “int legnagyobb_kulonbseg(const int*, int)” nevű függvényt, amely visszaadja a tömb

legnagyobb és legkisebb eleme közötti különbséget.

```
int legnagyobb_kulonbseg(const int *t, int n) {
    int min = t[0], max = t[0];
    for (int i = 1; i < n; i++) {
        if (t[i] < min) min = t[i];
        if (t[i] > max) max = t[i];
    }
    return max - min;
}
```

Feladat 23

Írjon egy “int tartalmaz(const int*, int, int)” nevű függvényt, amely 1-et ad vissza, ha a keresett érték megtalálható a tömbben, különben 0-t.

```
int tartalmaz(const int *t, int n, int ertek) {
    for (int i = 0; i < n; i++) {
        if (t[i] == ertek) return 1;
    }
    return 0;
}
```

Feladat 24

Írjon egy “void fordit_szoveg(char*)” nevű függvényt, amely megfordítja a paraméterként kapott szöveget.

```
#include <string.h>
void fordit_szoveg(char *s) {
    int h = strlen(s);
    for (int i = 0; i < h/2; i++) {
        char tmp = s[i];
        s[i] = s[h-1-i];
        s[h-1-i] = tmp;
    }
}
```

Feladat 25

Írjon egy “int osszeg_tartomany(const int*, int n, int min, int max)” nevű függvényt, amely visszaadja a tömb azon elemeinek összegét, amelyek a megadott zárt intervallumba esnek.

```
int osszeg_tartomany(const int *t, int n, int also, int felso) {
    int sum = 0;
```

```
    for (int i = 0; i < n; i++) {
        if (t[i] >= also && t[i] <= felso) sum += t[i];
    }
    return sum;
}
```