2. hét

Az óra témája

A laboron bemutatásra kerül:

- `char, unsigned char` típus,
- bit operátorok (**~**, **&**, **|**, **^**, **<<**, **>>**),
- elágazások (if-else, switch-case)
- `int scanf(const char *format,...);`

Mintafeladat

A következő példaprogram a `char, unsigned char` típusokkal végez műveleteket bit- és logikai operátorokkal.

Forráskód

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

int main()
{
    char c, b, a, d1, d2, d3, d4, d5;
    unsigned char uc;

    a = 65;
    b = 'A';
    c = b + 1;

    printf ( "char size = %d\n", sizeof(c) );  // char tipus merete
    printf ( " %d %d\n", a, b );
    printf ( " %c %c\n", a, b );

    printf ( " %c %d\n\n", c, c );

    d1 = 80;
    d2 = 'e';
    d3 = 't';
    d4 = 'e';
    d5 = 'r';

    printf ( " %c%c%c%c%c\n", d1, d2, d3, d4, d5 );

    uc = 128;
    c = 128;

    printf( "c = %d\n", c, c );
    printf( "uc = %d\n\n", uc, uc );

    // ~    egyes komplemens kepzo operator
```

```
    c = 1;
    printf( "c = %d \t ~c = %d\n", c, ~c );

    // <<   jobbra eltolas operator (shifteles)
    a = 0b00000001;
    printf( " a = %d\n", a );
    a = a << 3;
    printf( " a << 3 = %d\n", a );

    // >>   balra eltolas operator (shifteles)
    printf( " a = %d\n", a );
    a = a >> 3;
    printf( " a >> 3 = %d\n", a );

    // &    bitenkenti es kapcsolat
    printf("\n");
    a = 0b00000100;
    b = 0b00110100;
    printf ( " a & b = %d\n", a & b );
    printf ( " a && b = %d\n", a && b ); // logikai es kapcsolat

    // |    bitenkenti vagy kapcsolat
    printf("\n");
    a = 0b00000100;
    b = 0b00000011;
    printf ( " a | b = %d\n", a | b );
    printf ( " a || b = %d\n", a || b ); // logikai vagy kapcsolat

    // ? :  felteteles ertekadas
    printf("\n");
    b = 45;
    c = b & 0b000000001;
    printf ( " %d %s \n", b, (c ? "paratlan" : "paros") );
    b = 44;
    c = b & 0b000000001;
    printf ( " %d %s \n", b, (c ? "paratlan" : "paros") );

    return 0;
}
```

Mintafeladat

A következő példaprogram az elágazást mutatja be. Egy változóba beolvasunk egy számot az `int scanf(const char *format,...);` függvény segítségével, majd egy `if-else` szerkezettel megvizsgáljuk, hogy a beolvasott szám értéke nagyobb-e, mint 3. Figyelem! A program nem kezeli le, ha szám helyett mást (például betűket vagy írásjeleket) viszünk be.

Forráskód

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

int main()
{
```

```
    int a;

    printf( "Adjon be egy szamot: ");
    scanf( "%d", &a );
    printf( "\n" );

    if ( a > 3 )
    {
        printf( "%d nagyobb mint 3.\n", a );
    }
    else
    {
        if ( a < 3 )
        {
            printf( "%d kisebb mint 3.\n", a );
        }
        else
        {
                printf( "%d egyenlo 3-mal.\n", a );
        }
    }

    return 0;
}
```

Magyarázat
A kódban látható, hogy a `scanf( "%d", &a );` függvénynél nem az `a` változót, hanem annak memóriacímét adtuk át a címképző operátor segítségével (`&a`). A `%d scanf` *specifier* mondja meg, hogy egy előjeles decimális egészt szeretnénk beolvasni.

Mintafeladat
Készítsük el az előző programot `switch-case` szerkezettel.

Forráskód

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

int main()
{
    int a;

    printf( "Adjon be egy szamot (0-9): ");
    scanf( "%d", &a );
    printf( "\n" );

    switch ( a )
    {
        case 0 :
        case 1 :
        case 2 :
            printf( "%d kisebb mint 3.\n", a );
            break;
        case 3 :
            printf( "%d egyenlo 3-mal.\n", a );
```

```
            break;

        default :
            printf( "%d nagyobb mint 3.\n", a );
    }

    return 0;
}
```

Magyarázat

A `switch-case` szerkezet jellemzője, hogy az első igaz ág után következő valamennyi ág végrehajtódik. Ennek elkerülése érdekében valamennyi `case` ágat a `break` utasítással szoktunk lezárni, így kerülve el, hogy az utána következő ágak is végrehajtódjanak. A fenti példában azonban kihasználjuk a C nyelv ezen tulajdonságát hiszen bármely háromnál kisebb `case` ág igaz, a `printf( "%d kisebb mint 3.\n", a );` kiiratás fog végrehajtódni, ha a értéke kisebb háromnál. Ha egyetlen `case` feltétel sem teljesül, akkor a default ág hajtódik végre, amennyiben létezik.