

Függvények

Készítette: Vastag Atila

2017

Program írásakor az a célunk, hogy minél rövidebb, gyorsabb, karbantarthatóbb és átláthatóbb kódot írjunk.

Ha munkánk során egy logikai egészet (ugyanazokat a lépéseket) egynél többször használunk, akkor ezt a logikai egészet függvényekbe kell tenni.

A metódusok rövidebb, átláthatóbb és karbantarthatóbb egészekre bontják a programunkat (ha a logikai folyamat változtatásra szorul, akkor csak egy helyen kell kijavítani) így könnyítve azok tesztelését is, mert a függvények külön – külön tesztelhetők, hisz mindegyikük egy logikailag egész folyamatért kell, hogy felelős legyen.

A függvények az adatokon (paraméterek) számítási műveleteket végeznek, és előállítanak egy új, de nem kötelezően (a függvénynek nem kötelező, hogy visszatérési adata legyen). Ezen új értéket hívjuk a **függvény visszatérési értékének**. Ennek típusát deklarálni kell. A függvény visszatérési típusa gyakorlatilag bármilyen lehet. Elképzelhető egyszerű adattípusok (logikai, karakter, valós, egész, szöveg, felsorolás, ...), de akár összetett adattípusok is (rekord, lista, tömb, objektum, ...) is.

A függvény visszatérési típusát meg kell adni a függvény definíciós részében.

A függvény definíciós részének általános alakja:

```
[láthatóság] <visszatérő érték típusa> <függvénynév> (<formális paraméterlista>)  
{  
    utasítástörzs utasításai  
}
```

Vegyünk egy egyszerű példát egy függvényen melynek az a feladata, hogy összeadjon két számot !

```
class Program
{
    static void Main(string[] args)
    {
        int x = 5;
        int y = 10;

        int eredmeny = Osszeadas(x, y);

        Console.WriteLine($"Az összeg: {eredmeny}");
    }
}
```

Az **Osszeadas** olyan függvény, amely **int** típusú értéket ad vissza.

Paraméterként két **int** típusú adatot fogad.

```
private static int Osszeadas(int elsoOsszeadando, int masodikOsszeadanto)
{
    int osszeg = elsoOsszeadando + masodikOsszeadanto;

    return osszeg;
}
```

Feladata két szám összeadása !

```
class Program
{
    static void Main(string[] args)
    {
        int x = 5;
        int y = 10;

        int eredmeny = Osszeadas(x, y);

        Console.WriteLine("Az osszeg: {0}", eredmeny);
    }
}
```

Függvény
meghívása

visszatérő
érték

láthatóság

függvény
neve

paraméterek

```
private static int Osszeadas(int elsoOsszeadando, int masodikOsszeadanto)
{
    int osszeg = elsoOsszeadando + masodikOsszeadanto;

    return osszeg;
}
```

Paraméter
típusai

```
class Program
```

```
{
```

```
    static void Main(string[] args)
```

```
    {
```

```
        int x = 5;
```

```
        int y = 10;
```

Az x és y változók értéket kapnak

1

```
        int eredmény = Osszeadas(x, y);
```

Meghívásra kerül az **Osszeadas** függvény

2

```
        Console.WriteLine("Az összeg: {0}", eredmény);
```

6

```
    }
```

a program kiírja az eredményt: 15

A függvény megkapja paraméterként az x és y értékét, ahol **elsőOsszeadando** = x-el és a **masodikOsszeadando** = y-al

```
    private static int Osszeadas(int elsőOsszeadando, int masodikOsszeadando)
```

```
    {
```

```
        int osszeg = elsőOsszeadando + masodikOsszeadando;
```

```
        return osszeg;
```

```
    }
```

```
}
```

3

A függvény elvégzi a két változó összeadását, majd a **return** kulcsszó segítségével visszatér arra a programrészre ahonnan meg lett hívva egy **int** típusú értékkel (**osszeg**), mivel **int** típusú a visszatérő érték.

4

az **eredmény** változó felveszi a függvény visszatérő érték értékét (osszeg = 15)

5

Vegyünk egy egyszerű példát egy függvényen melynek az a feladata, hogy kiírjon valamilyen szöveget !

```
class Program
{
    static void Main(string[] args)
    {
        string nev = "Smuci";

        UdvozloSzovegKiirasa(nev);
    }

    private static void UdvozloSzovegKiirasa(string text)
    {
        Console.WriteLine($"Udvozlom {text}");
    }
}
```

Az `UdvozloSzovegKiirasa` olyan függvény, amelynek nincs visszatérő értéke (**void**).

Feladata valami kiírás!

Paraméterként **string** típusú adatot fogad

```
class Program
```

```
{
```

```
    static void Main(string[] args)
```

```
    {
```

```
        string nev = "Smuci";
```

```
        UdvozloSzovegKiirasa(nev);
```

```
        private static void UdvozloSzovegKiirasa(string text)
```

```
        {
```

```
            Console.WriteLine($"Udvozlom {text}");
```

```
        }
```

```
    }
```

Függvény
meghívása

láthatóság

visszatérő
érték
NINCS

függvény
neve

paraméter

Paraméter
típusai


```
class Program
```

```
{
```

```
    static void Main(string[] args)
```

```
    {
```

Az **nev** változó értékét kap

```
        string nev = "Smuci";
```

1

Meghívásra kerül az **UdvozloSzovegKiirasa** függvény

2

```
        UdvozloSzovegKiirasa(nev);
```

```
    }
```

4

A függvény megkapja paraméterként
a **text** változó megkapja a **nev** értékét,
vagyis **text = nev**

```
    private static void UdvozloSzovegKiirasa(string text)
```

3

```
    {
```

```
        Console.WriteLine($"Udvozlom {text}");
```

```
    }
```

```
}
```

A függvény elvégzi a kiírást, majd a **return** kulcsszó segítségével, mert nincs visszatérő érték, visszatér arra a programrészre ahonnan meg lett hívva

```
class Program
{
    static void Main(string[] args)
    {
        int x = 5;
        int y = 10;

        int eredmeny = Osszeadas(x, y);

        Console.WriteLine("Az összeg: {0}", eredmeny);
    }
}
```

Ez azt jelenti, hogy ebben az osztályban, vagyis a fő programban nem használhatók változókat melyeket **elsőOsszeadando** illetve **masodikOsszeadanto** neveznék el

Az **elsőOsszeadando** és **masodikOsszeadanto** változók csak a függvény testében léteznek, a függvény végrehajtása után megszűnnek létezni

```
private static int Osszeadas(int elsőOsszeadando, int masodikOsszeadanto)
{
    int osszeg = elsőOsszeadando + masodikOsszeadanto;

    return osszeg;
}
```

Ügyelni kell arra is, hogy a programunkban ugyanolyan elnevezésű változó globálisan (látható az egész programban / osztályban ne létezzen mint a paraméterekben megadott változók nevei

Függvényírás szabályai:

- mindig meg kell adni a láthatóságot
- mindig definiálni kell a visszatérő érték típusát
- mindig definiálni kell a függvény nevét. Igyekezzünk olyan nevet adni a függvénynek, amely leírással bír arról, hogy a függvénynek mi is a feladata
- a paraméterek típusát kötelező megadni
- ha a függvénynek van visszatérő típusa definiálva, akkor kötelesek vagyunk **return** kulcsszó segítségével olyan típusú adattal visszatérni a meghívás helyére
- **két, vagy több egyforma elnevezésű függvényt csak akkor írhatunk, ha azok legalább egy valamiben különböznek (paraméterek száma, paraméterek típusa)**

```
class Program
{
    static void Main(string[] args)
    {
        int x = 5;
        int y = 10;

        int eredmeny = Osszeadas(x, y);

        Console.WriteLine($"Az összeg: {eredmeny}");
    }

    private static int Osszeadas(int szam1, int szam2)
    {
        int osszeg = szam1 + szam2;

        return osszeg;
    }

    private static int Osszeadas(int szam1, int szam2, int szam3)
    {
        int osszeg = szam1 + szam2 + szam3;

        return osszeg;
    }
}
```

A program nem fog hibát jelezni, mert annak ellenére, hogy mindkét függvénynek ugyanaz a neve, ugyanolyan a visszatérési adata, különbözik a paraméterek számában.

FÜGGVÉNY PARAMÉTER ALAPÉRTELMEZETT ÉRTÉKE

```
public double Fizetes(int oraber, int oraszam = 40)
{
    return oraszam * oraber;
}

double tuloraFizetes = Fizetes(1200, 60);
double normalFizetes = Fizetes(1200);
```

Hogy lehetséges az, hogy a **normalFizetes** függvény meghívható, hisz a **Fizetes** függvény két paramétert kér?

Ez úgy lehetséges, hogy az **oraszam** paraméternek be van állítva egy alapértelmezett érték (**40**) és ha a **Fizetes** függvényt csak egy paraméterrel hívjuk meg (**normalFizetes**), akkor az alapértelmezett értéket veszi, ha két paraméterrel, akkor az **oraszam** paraméter átveszi a függvény meghívásában megadott értéket (**60**).

```
Console.WriteLine($"{tuloraFizetes}"); //72.000
```

```
Console.WriteLine($"{normalFizetes}"); //48.000
```

KITERJESZTŐ FÜGGVÉNYEK

extension methods

```
public struct Jatekos  
{  
    public string Nev;  
  
    public int Magassag;  
  
    public int EviPontszam;  
}
```

```
Jatekos[] jatekosok = new Jatekos[10];  
double atlagMagassag = jatekosok.AtlagMagassag();
```

Hol a hiba?

Vajon a jatekosok tömböknek van olyan AtlagMagassag függvénye???

Nincs!!!!

megoldás: kiterjesztő függvény, mely csak egy bizonyos adattípuson (`Jatekos[]`) hívható meg !!!

```
public static class JatekosExtension
{
    /// <summary>
    /// A jatekosok magassaganak atlagat szamitsa ki
    /// </summary>
    /// <param name="jatekosok">Jatekos[]</param>
    /// <returns></returns>
    public static double AtlagMagassag(this Jatekos[] jatekosok)
    {
        double atlag = 0;

        foreach(Jatekos jatekos in jatekosok)
        {
            atlag += jatekos.Magassag;
        }

        return atlag / jatekosok.Length;
    }
}
```

FELADATOK:

1 – Írj egy mértékegység konvertáló programot. Az átalakítást Celsius-ból függvényrel oldjuk meg amely két paramétert kap! Az első egy hőmérséklet érték, a második paraméter pedig azt jelzi milyen mértékegységre akarjuk átkonvertálni ('F': Fahrenheit, 'K': Kelvin).

($K = C + 273,15$; $F = 9/5 * C + 32$).

Egyes logikai egészeket alkotó műveleteket függvényekkel oldjuk meg.

2 - Írj egy programot mely bekéri két pont koordinátáját (X,Y) koordinátáit és visszaadja a két pont távolságát!

Egyes logikai egészeket alkotó műveleteket függvényekkel oldjuk meg.

3 – Deviza konvertáló programot kell írunk, mely a megadott HUF forintot japán jen, dollár és svájci frankba konvertálja kiválasztás alapján. Az árfolyamokat konstans változóba helyezzük el. Minden konverzió után jelenítsük meg az értéket EUR-ban is, ha tudjuk, hogy $1 \text{ JPY} = 0.75 \text{ EUR}$, $1 \text{ USD} = 0.8 \text{ EUR}$ és $1 \text{ CHF} = 0.55 \text{ EUR}$.

HUF és az EUR árfolyamát véletlen számmal állítsuk elő 350 – 400 közt.

Egyes logikai egészeket alkotó műveleteket függvényekkel oldjuk meg.

4 - Virág és Jázmin számkitalálós játékot készített, a következő módom:
Virág gondolt két számra, az első 0 és 9 közt, a második 40 és 50 közt, majd a számítógép egy véletlen számot (**rnd**) generált titokban e két érték közt, amit Jázmin nem láthatott.

Jázmin feladata az volt, hogy a véletlen számot kitalálja.

A program eredményként Jázmin próbálkozásainak számát írta ki miután kitalálta a számítógép által választott számot. Minden beírt szám után a program Jázmint értesítette, hogy nagyobb vagy kisebb a kitalálendő szám mint amit ő beírt.

5 – Egy kis céget vezetünk, ahol 5 munkás dolgozik nekünk. Minden hét végén el kell számolnunk az alkalmazottak munkabérét. Egy hét 40 munkaórából áll, ez a normál munkidő amit 1000Ft / órában fizetünk ki. Minden túlórárt 1500Ft / óra értékben kell kifizetni. Számítsuk ki és írjuk ki a képernyőre, hogy a héten ki hány órát dolgozott, mennyi az össz keresete. A dolgozók nevét és a heti ledolgozott órák számát konzolról kérjük be.

6 - A program bekér egy dolgozat pontszámot (x) a felhasználótól és kiír egy érdemjegyet az alábbiak szerint! : 1: $x < 50$; 2: $50 \leq x < 60$; 3: $60 \leq x < 70$; 4: $70 \leq x < 85$; 5: $x \geq 85$.

7 - Tegyük fel, hogy szeretnéd a szobád egyik falát kékre festeni!

- Mennyi festéket kell vened, ha 1 m² lefestéséhez 0,15 liter festék kell?

- Ha 1 liter festék ára 930 Ft, akkor mennyit kell fizetni a szükséges festékért?

8 – Az önkormányzat a 20 méternél keskenyebb és 30 méternél rövidebb telkekre 25%-os kedvezményt ad a teljes adóból. Írjunk programot, mely a telek szélessége és a hossza alapján kiszámítja a teljes adó mértékét!