

# Algoritmusok és alapfogalmak

- **Algoritmus:** Egy feladat megoldására szolgáló lépések sorozata, amely **véges számú lépésből áll, egyértelmű, és minden lépése végrehajtható.**
  - **Iteráció:** Egy ciklus egy teljes végrehajtási köre.
  - **Feltételes elágazás (if-else szerkezet):** Az elágazás egy logikai feltétel vizsgálatán alapul, amely meghatározza, hogy a program melyik utasítást hajtja végre.
  - **Ciklusok:** Egy adott kódrészlet ismételt végrehajtására szolgálnak. Fő típusai:
    - **Számlálós ciklus:** Meghatározott számú ismétlést hajt végre.
    - **Feltételes ciklus:** Addig ismétlődik, amíg egy feltétel igaz.
    - **Végtelen ciklus:** Egy olyan ciklus, amelynek nincs kilépési feltétele, ezért folyamatosan fut.
  - **Pszudokód:** Egy egyszerű, ember által olvasható formátum az algoritmusok leírására, amely nem kötött egy adott programozási nyelvhez.
  - **Flowgorithm:** Egy vizuális programozási eszköz, amely lehetővé teszi algoritmusok folyamatábrákban való megadását és tesztelését.
- 

## Változók és adattípusok

- **Változó:** Egy olyan névvel ellátott memóriahely, amely egy adott értéket tárol. Az értéke a program futása során megváltozhat.
  - **Változók típusai Flowgorithm-ban:**
    - **Egész szám (Integer)**
    - **Valós szám (Float)**
    - **Szöveg (String)**
    - **Logikai érték (Boolean - igaz vagy hamis)**
  - **Változó deklarációja:** A változó használata előtt meg kell adni a nevét és típusát.
  - **Inicializálás:** Egy változó kezdőértékének beállítása.
- 

## Matematikai és logikai műveletek

- **Osztási maradék (% vagy mod):** Kiszámítja egy szám osztási maradékát (például  $10 \bmod 3 = 1$ ).
  - **Összefűzés (&):** Két szöveg összekapcsolása (például "Hello" & " World" → "Hello World").
  - **Logikai műveletek:**
    - **AND (&& vagy and)** – Mindkét feltételnek igaznak kell lennie.
    - **OR (|| vagy or)** – Legalább egy feltétel igaz.
    - **NOT (! vagy not)** – Az igaz állítás hamisra változik, és fordítva.
-

## Vezérlési szerkezetek és ciklusok

- **Elágazás (if-else szerkezet):** Feltételes végrehajtást tesz lehetővé, ahol a program egyik vagy másik utasítást hajtja végre a feltétel eredményétől függően.
  - **Számlálós ciklus (for loop):** A ciklus adott számú ismétlést hajt végre.
  - **Feltételes ciklus (while loop):** Addig fut, amíg a megadott feltétel igaz marad.
  - **Lépésenkénti végrehajtás:** A program egyes utasításait külön-külön hajtja végre, hogy a hibák könnyebben azonosíthatók legyenek.
- 

## Függvények és eljárások

- **Függvény:** Egy adott művelet elvégzésére szolgáló kódrészlet, amelynek lehet **visszatérési értéke**.
- **Paraméter:** A függvény bemenő adata, amely befolyásolja a működését.
- **Visszatérési érték:** A függvény által visszaadott érték. Példa:

```
plaintext
MásolásSzerkesztés
Nagyobbegylenlő(x, y) → x vagy y
```

Ha x nagyobb, akkor x-et ad vissza, ha y nagyobb, akkor y-t, ha egyenlőek, akkor x-et.

- **Függvény hívása:** Egy függvény végrehajtása a program egy adott pontján.
  - **Beépített függvények a Flowgorithm-ban:**
    - **random(n)** – 0 és n-1 közötti véletlenszám generálása.
    - **sqrt(x)** – Egy szám négyzetgyökének kiszámítása.
    - **abs(x)** – Egy szám abszolút értékének kiszámítása.
- 

## Típusalgoritmusok

- **Megszámlálás algoritmus:** Egy adott tulajdonságú elemek számát határozza meg egy halmazban.

```
plaintext
MásolásSzerkesztés
DB := 0
Ciklus i = 1-től N-ig
    Ha adott tulajdonság teljesül, akkor DB := DB + 1
Ciklus vége
```

- **Maximumkiválasztás algoritmus:** Meghatározza a legnagyobb értékű elemet egy adott halmazból.
-

## Példák és felhasználási módok

- **Fej vagy írás algoritmus:**
    - Véletlenszám-generálás `random()` függvénnyel.
    - Elágazás használata az eredmény kiírására (F vagy I).
    - Számlálós ciklus a több dobás megvalósítására.
  - **Chatbot algoritmus:**
    - Felhasználói bevitelt kezel.
    - Ha-akkor szerkezetekkel válaszokat ad.
    - Szöveg kiírását és beolvasását végzi változókkal.
- 

# ALGORITMUSOK ÉS PROGRAMOZÁS ALAPFOGALMAK

## 1. Algoritmus

Egy algoritmus egy **probléma megoldásának pontos, egyértelmű és véges lépésekből álló leírása**, amely biztosítja a kívánt eredmény elérését.

- **Tulajdonságai:**
  - **Véges:** Az algoritmusnak véges számú lépésből kell állnia.
  - **Egyértelmű:** Minden lépésnek egyértelműnek és meghatározottnak kell lennie.
  - **Hatékony:** Az algoritmus optimális esetben a legkevesebb erőforrást használja fel.
  - **Bemenet és kimenet:** Az algoritmus bemeneteket kap, és eredményként kimenetet ad vissza.

## 2. Algoritmus leírási módjai

Az algoritmusokat különböző módokon lehet leírni:

- **Szöveges leírás:** A műveleteket ember által érthető módon írjuk le.
  - **Folyamatábra:** Grafikus ábrázolás, amelyben különböző szimbólumokat használunk a műveletek megjelenítésére.
  - **Pszudokód:** Egy formális, de még nem programozási nyelvhez kötött leírás.
-

# VEZÉRLÉSI SZERKEZETEK

## 3. Szekvencia (sorozatos végrehajtás)

Az utasításokat **sorban egymás után** hajtja végre a program.

### Példa szekvenciális végrehajtásra:

```
plaintext
MásolásSzerkesztés
Kiírás: "Add meg a számot!"
Beolvasás: x
Kiírás: "A beírt szám: ", x
```

## 4. Elágazás (feltételes végrehajtás)

Az elágazás során a program egy feltételt vizsgál meg, és annak eredményétől függően dönt, hogy melyik ágon halad tovább.

- **Egyszerű elágazás:**

```
plaintext
MásolásSzerkesztés
Ha x > 0 akkor
    Kiírás: "Pozitív szám"
elágazás vége
```

- **Kétirányú elágazás (if-else szerkezet):**

```
plaintext
MásolásSzerkesztés
Ha x > 0 akkor
    Kiírás: "Pozitív"
különben
    Kiírás: "Negatív vagy nulla"
elágazás vége
```

- **Többirányú elágazás (nested if, switch-case szerkezet)**

Ha több lehetőség közül kell választani, egymásba ágyazott elágazásokat használunk.

---

## 5. Ciklusok (ismétlés)

A ciklusok segítségével egy utasítássorozatot **többször végrehajthatunk**.

- **Számlálós ciklus (for loop):** Egy adott számú ismétlés végrehajtására használjuk.

```
plaintext
MásolásSzerkesztés
Ciklus i = 1-től 10-ig
    Kiírás: i
Ciklus vége
```

- **Feltételes ciklus (while loop):** A ciklus addig ismétlődik, amíg a megadott feltétel igaz marad.

```
plaintext
MásolásSzerkesztés
i = 1
Amíg i < 10
    Kiírás: i
    i = i + 1
Ciklus vége
```

- **Végtelen ciklus:** Ha nincs kilépési feltétel, a ciklus örökké fut.

---

## VÁLTOZÓK ÉS ADATTÍPUSOK

### 6. Változók

Egy változó egy névvel ellátott memóriahely, amelyben **értékeket tárolhatunk és módosíthatunk** a program futása során.

- **Deklaráció:** A változót használat előtt meg kell határozni.
- **Inicializálás:** A változó első értékének beállítása.

### 7. Adattípusok

- **Egész szám (Integer)** – pl. 5, -10, 200
- **Valós szám (Float, Real)** – pl. 3.14, -2.5
- **Szöveg (String)** – pl. "Hello", "Programozás"
- **Logikai érték (Boolean)** – igaz (True) vagy hamis (False)

---

## MATEMATIKAI ÉS LOGIKAI MŰVELETEK

### 8. Matematikai műveletek

- **Összeadás (+), kivonás (-), szorzás (\*), osztás (/)**
- **Osztási maradék (mod, %)** → például  $10 \bmod 3 = 1$

### 9. Logikai műveletek

- **És (&& vagy and)** – mindkét feltételnek igaznak kell lennie.
- **Vagy (|| vagy or)** – legalább az egyik feltételnek igaznak kell lennie.
- **Nem (! vagy not)** – igazból hamisat, hamisból igazat csinál.

# FÜGGVÉNYEK ÉS ELJÁRÁSOK

## 10. Függvények

Egy függvény egy **ismétlődő műveletsorozat**ot tartalmaz, amely lehet paraméterezett és visszatérési értékkel rendelkezhet.

- **Példa egy függvényre:**

```
plaintext
MásolásSzerkesztés
Függvény négyzet(x)
    visszatérési érték: x * x
Függvény vége
```

- **Függvényhívás:**

```
plaintext
MásolásSzerkesztés
eredmény = négyzet(5)
Kiírás: eredmény
```

## 11. Paraméterek

A függvények bemenő adatai, amelyeket a híváskor adunk meg.

## 12. Visszatérési érték

A függvény által visszaadott eredmény.

---

# TÍPUSALGORITMUSOK

## 13. Megszámlálás

Adott tulajdonságú elemek számolása egy listában.

```
plaintext
MásolásSzerkesztés
DB := 0
Ciklus i = 1-től N-ig
    Ha elem tulajdonsága teljesül, akkor DB := DB + 1
Ciklus vége
```

## 14. Maximumkiválasztás

A legnagyobb elem kiválasztása egy listából.

```
plaintext
MásolásSzerkesztés
max = első elem
Ciklus i = 2-től N-ig
    Ha lista[i] > max, akkor max := lista[i]
Ciklus vége
```

---

Ezek a **legfontosabb definíciók és algoritmusleírások** az **algoritmizálás** témakörében. Ha szeretnéd, részletezhetem valamelyik részt, vagy további példákat adhatok! 😊

**Ezt mondtad:**

kérek további példákat

A ChatGPT ezt mondta:

Íme **további példák** az algoritmizálás, vezérlési szerkezetek, változók, függvények és típusalgoritmusok témájában, a könyv alapján.

---

## 1. Alapvető algoritmusok példái

### 1.1 Egyszerű kiírás

Feladat: Írjunk ki egy üdvözlő üzenetet a képernyőre.

**Pszudokód:**

```
plaintext
MásolásSzerkesztés
Kiírás: "Helló, világ!"
```

**Flowgorithm folyamatábra:**

- Egy **kiírás (Output)** blokkba írjuk a "Helló, világ!" szöveget.
-

## 1.2 Egyszerű beolvasás és kiírás

Feladat: Kérjük be a felhasználó nevét, majd írjuk ki egy üdvözlő üzenetben.

### Pszudokód:

```
plaintext
MásolásSzerkesztés
Kiírás: "Add meg a neved:"
Beolvasás: név
Kiírás: "Helló, " & név & "!"
```

### Magyarázat:

- A Beolvasás utasítás lehetővé teszi, hogy a felhasználó adjon meg egy értéket.
  - Az & operátorral összefűzzük a szövegeket.
- 

## 2. Vezérlési szerkezetek példái

### 2.1 Egyszerű elágazás (if szerkezet)

Feladat: Kérjünk be egy számot, és írjuk ki, hogy **pozitív, negatív vagy nulla**.

### Pszudokód:

```
plaintext
MásolásSzerkesztés
Kiírás: "Adj meg egy számot:"
Beolvasás: szám
Ha szám > 0 akkor
    Kiírás: "A szám pozitív."
különben ha szám < 0 akkor
    Kiírás: "A szám negatív."
különben
    Kiírás: "A szám nulla."
Elágazás vége
```

### Flowgorithm implementáció:

- Egy **bemenet (Input)** blokk a szám beolvasására.
  - Egy **elágazás (Decision)** blokk a szám értékének vizsgálatához.
-



## 2.2 Számlálás ciklus (for loop)

Feladat: Írjuk ki az első 10 számot.

### Pszeudokód:

```
plaintext
MásolásSzerkesztés
Ciklus i = 1-től 10-ig
    Kiírás: i
Ciklus vége
```

### Magyarázat:

- A ciklus **1-től 10-ig** halad, és minden lépésben kiírja az aktuális *i* értéket.
- 

## 2.3 Feltételes ciklus (while loop)

Feladat: Addig kérjünk be egy számot, amíg az pozitív.

### Pszeudokód:

```
plaintext
MásolásSzerkesztés
Beolvasás: szám
Amíg szám > 0
    Kiírás: "Adj meg egy másik számot:"
    Beolvasás: szám
Ciklus vége
```

### Flowgorithm implementáció:

- Egy **bemenet (Input)** blokk a szám beolvasására.
  - Egy **while ciklus**, amely ismétlést biztosít, amíg a szám pozitív.
-

## 3. Függvények példái

### 3.1 Egyszerű függvény visszatérési értékkel

Feladat: Írjunk egy függvényt, amely egy szám négyzetét adja vissza.

#### Pszeudokód:

```
plaintext
MásolásSzerkesztés
Függvény négyzet(x)
    Visszatérési érték: x * x
Függvény vége
```

Kiírás: "Adj meg egy számot:"

Beolvasás: szám

Kiírás: "A szám négyzete: " & négyzet(szám)

#### Magyarázat:

- A  $\text{négyzet}(x)$  függvény egy számot kap bemenetként és visszaadja annak négyzetét.
- 

### 3.2 Függvény két szám közül a nagyobbát adja vissza

Feladat: Hozzunk létre egy függvényt, amely két szám közül visszaadja a nagyobbat.

#### Pszeudokód:

```
plaintext
MásolásSzerkesztés
Függvény maximum(a, b)
    Ha a > b akkor
        Visszatérési érték: a
    különben
        Visszatérési érték: b
Függvény vége
```

Kiírás: "Adj meg két számot:"

Beolvasás: x, y

Kiírás: "A nagyobb szám: " & maximum(x, y)

#### Flowgorithm implementáció:

- Egy **függvény (Function)** blokk, amely a és b változókat kap paraméterként.
  - Egy **elágazás (Decision)** a nagyobb érték kiválasztására.
-

## 4. Típusalgoritmusok példái

### 4.1 Megszámlálás algoritmusa

Feladat: Számoljuk meg, hány 5-tel osztható szám van 1 és 50 között.

#### Pszudokód:

```
plaintext
MásolásSzerkesztés
DB := 0
Ciklus i = 1-től 50-ig
    Ha i mod 5 = 0 akkor
        DB := DB + 1
Ciklus vége
Kiírás: "Az 5-tel osztható számok száma: " & DB
```

#### Magyarázat:

- $\text{mod } 5 = 0$  azt vizsgálja, hogy egy szám osztható-e 5-tel.
  - A DB változó számolja az osztható számokat.
- 

### 4.2 Maximum kiválasztása egy listából

Feladat: Egy lista legnagyobb elemének megkeresése.

#### Pszudokód:

```
plaintext
MásolásSzerkesztés
Kiírás: "Add meg az elemek számát:"
Beolvasás: N
Kiírás: "Add meg az első számot:"
Beolvasás: max

Ciklus i = 2-től N-ig
    Kiírás: "Add meg a következő számot:"
    Beolvasás: szám
    Ha szám > max akkor
        max := szám
Ciklus vége

Kiírás: "A legnagyobb szám: " & max
```

#### Magyarázat:

- Az első számot `max` változóként eltároljuk.
  - A ciklus összehasonlítja az új számokat a `max` értékével.
-