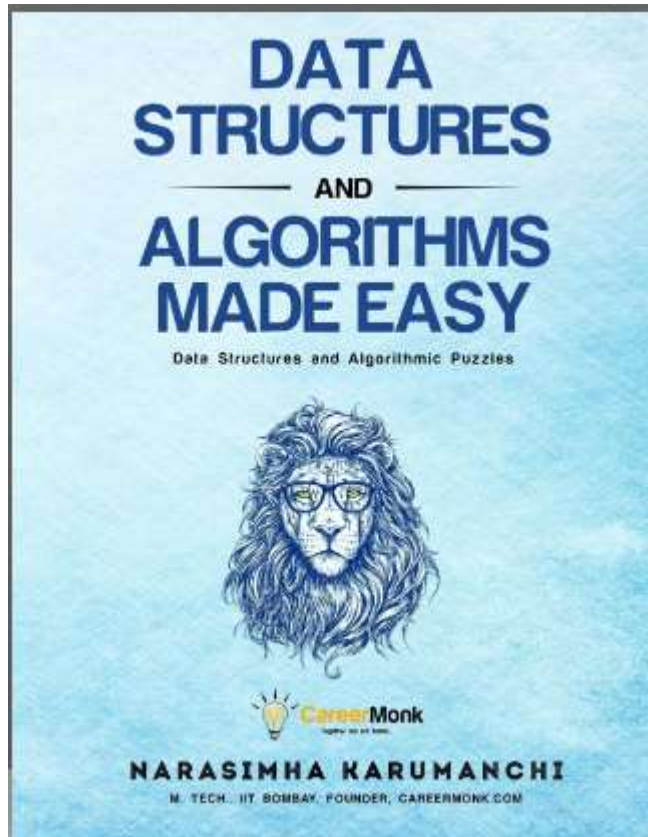


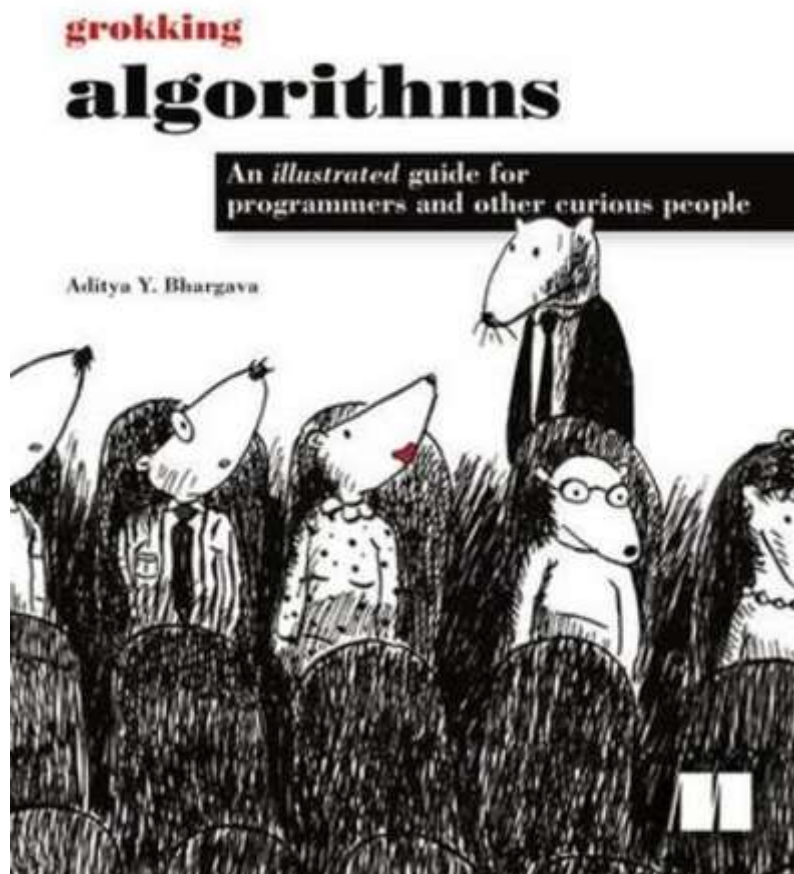
Algoritmusok 1

- A szoftverfejlesztés alapjait az adatstruktúrák és az algoritmusok jelentik.
- Az adatstruktúrák és algoritmusok tanulmányozása sok **kritikus gondolkodást** és **agyi edzést** tesz szükségessé.
- Számos hatalmas IT-cég, mint például a Google, az Amazon és a Microsoft, valamint a fiatal, feltörekvő startupok, mint például a Linktree és a StackBlitz készít anyagokat az adatstruktúrákra és az algoritmusokra összpontosítva.
- Fontos, hogy tisztában legyünk a tanulmányozásukhoz szükséges **legjobb forrásokkal**.
- Bizonyos **könyvek** segítenek mélyebben megérteni az algoritmusok világát, és **kiváló alapot** nyújtanak a fejlesztői problémamegoldó készségek fejlesztéséhez. → lásd következő lap



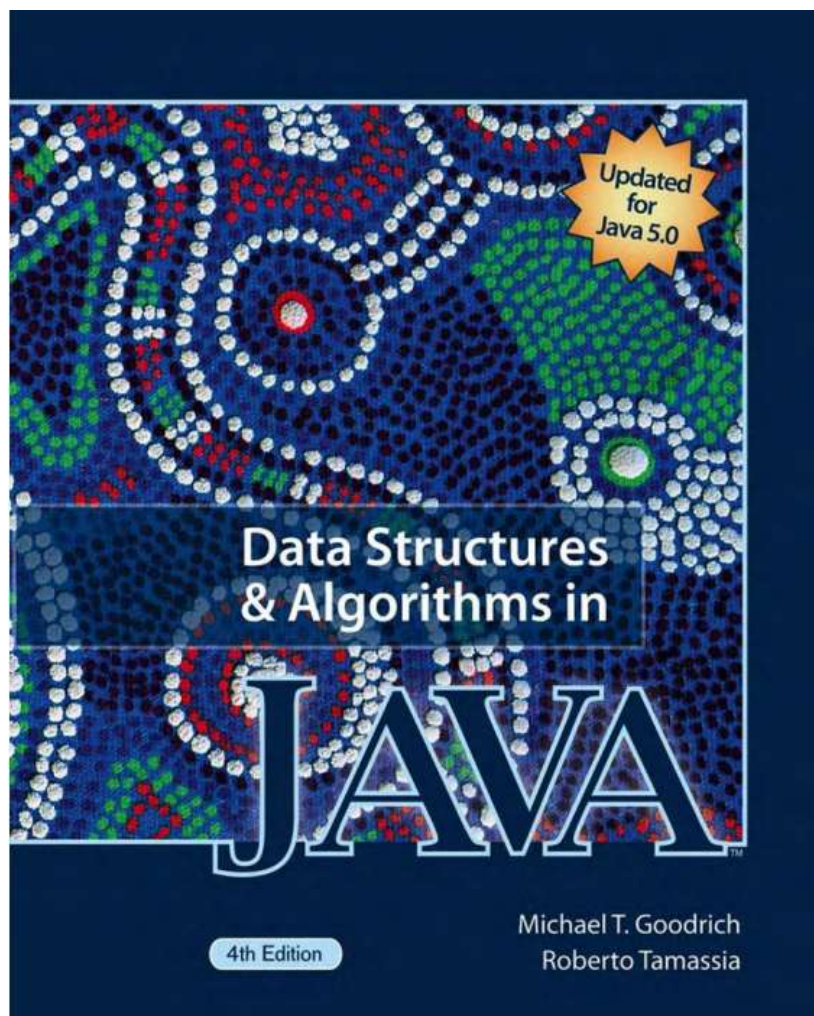
Adatstruktúrák és algoritmusok egyszerűen:

- Ez a könyv bemutatja az adatstruktúrák és algoritmusok alapjait.
- Megismerheted a tömböket, karakterláncokat és adatfákat, valamint az algoritmusok működését.
- A gyakorlatok C/C++ kódot használnak, így ha ismered ezt a nyelvet, fantasztikus hely lehet a kezdőknek is.

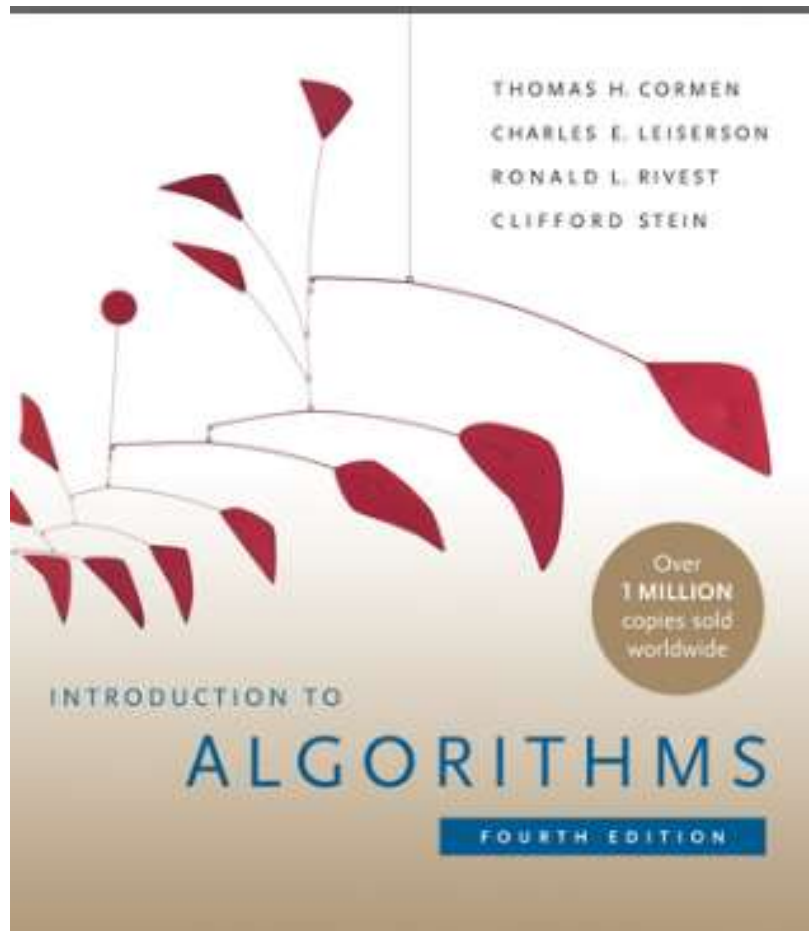


Grokking algoritmusok:

- Ez a könyv a valós algoritmusokkal kapcsolatos problémákat járja körül.
- A Python nyelvet használja, és több mint 400 kép segíti az adatszerkezetek és algoritmusok megértését.
- Nagyon szájbarágósan és képekkel illusztrálva mutatja be a könyv az alap adatszerkezeteket és algoritmusokat.



- **Adatstruktúrák és algoritmusok JAVA-ban:**
- Ez a könyv részletesen ismerteti az adatstruktúrákat és algoritmusokat, különös tekintettel a Java nyelvre.
- A matematikai elemzések igényesek, de érthetőek az olvasók széles köre számára.
- A könyv számos illusztrációt tartalmaz az adatstruktúrákról és azok működéséről.



Kezdőkönyv az algoritmusokról:

- Ez a könyv az algoritmusok alapos ismeretét célozza meg.
- Segít a hatékony programozásban és a memóriatakarékos megoldásokban.

Az algoritmusok tanulásához számos [online forrás áll rendelkezésre](#), amelyek segíthetnek mélyebben megérteni ezeket a kulcsfontosságú témákat.

Itt van néhány ajánlás:

- [\[Szegedi Tudományegyetem Online Algoritmusok\]](https://www.inf.u-szeged.hu/~cimreh/Online_algorithmusok.pdf)(https://www.inf.u-szeged.hu/~cimreh/Online_algorithmusok.pdf):

Ez a dokumentum részletesen bemutatja az online algoritmusokat, a síbérlestől a gépköltséges ütemezésig. Tartalmaz példákat és matematikai elemzéseket is.

- [\[Coursera\]](https://www.coursera.org/)(<https://www.coursera.org/>):

A Coursera online kurzusokat kínál, amelyek az algoritmusokról és a gépi tanulásról szólnak. Olyan egyetemek oktatói tartanak itt kurzusokat, mint a Stanford, a Princeton és a Michigan.

- [\[GeeksforGeeks\]](https://www.geeksforgeeks.org/)(<https://www.geeksforgeeks.org/>):

A GeeksforGeeks egy hatalmas online forrás, amely részletesen bemutatja az algoritmusokat, adatszerkezeteket és programozási technikákat. Rengeteg cikket, példát és gyakorlati feladatot találsz itt.

Algoritmusok elsajátítása:

Lépések:

- 1. A probléma pontos meghatározása:** Világosan fogalmazd meg a problémát.
- 2. Felbontás kisebb részekre:** Bontsd le a problémát egyszerűbb részfeladatokra.
- 3. Minden részfeladat megoldásának definiálása:** Határozd meg a megoldást minden részfeladatra.
- 4. Megvalósítás:** Implementáld a megoldást.
- 5. Hatékonyság optimalizálása:** Végül törekedj a hatékony megoldásra.

Ne feledd, az algoritmusok gyakorlása időt és kitartást igényel, de a folyamatos gyakorlás hatékonyabbá tesz a problémamegoldásban!

Gyakorlás és ismétlés:

- Oldj meg minél több algoritmusos feladatot. Használj online platformokat, mint a LeetCode vagy a HackerRank.
- Ismételd át a már megoldott feladatokat, hogy rögzítsd a tanultakat.

Könyvek és online források:

- Olvass könyveket az algoritmusokról.
- Használj online kurzusokat, például a Coursera-t vagy a Khan Academy-t.

Gyakorlás papíron:

- Írd le az algoritmusokat és a megoldásokat papíron. Ez segít jobban megérteni a lépéseket.

Összehasonlítások és példák:

- Helyezd át az algoritmusokat mindennapi élethelyzetekbe. Például, hogyan rendeznéd a zoknikat egy kupacból vagy keresnél egy szót a szótárban.

Mesterséges intelligencia és gépi tanulás:

- Tanulj meg algoritmusokat a gépi tanulás és a mesterséges intelligencia területén. Ez segít mélyebben megérteni az algoritmusok működését.

Hogyan írjunk algoritmust?

A Cambridge-i etegyemen készült egyik tanulmány szerint a szövegekben nem található olyan jelölés, amely megkülönböztetné az első és az utolsó betűt a szövegben, ha a többi a leggyakoribb özeássivsszszg a jlezmlő, a szöveg aokr is tleejs mrtébéekn ovasalthó mraad. A jneleésg mgáayzrataa az, hogy az emberi agy nem egyedgi btűeket, hneam tleejs szakvat ovals. Íme a bzoinytéik!

COPILOT használata



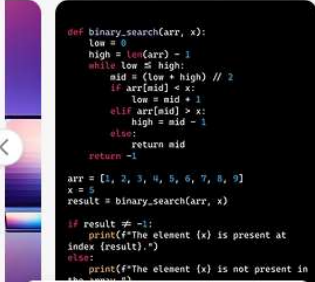
Microsoft Copilot

<https://copilot.microsoft.com> · Oldal lefordítása


Microsoft Copilot: Your everyday AI companion

Copilot


Az Ön mindennapi AI-segítőtársa



Python-szkript írása bináris keresés végrehajtásához



Készítsen plakátot egy szabadtéri zenei fesztiválra, retro stílusban, szöveg nélkül



Készítsen egy evite-t egy babaváráshoz, amely ajándékköteteket tartalmaz

A Copilot mesterséges intelligenciát használ. Hibák keresése. [Kifejezések](#) | [Adatvédelem](#) | [GYIK](#)

Beszélgetési stílus kiválasztása

További Kreatív **További Kiegyensúlyozott** További Precíz

Kérdezzen valamit...

0/2000

Itt van egy egyszerű pszeudo kód az algoritmus számára, amely megőrzi a szavak első és utolsó karaktereit, miközben a többi karaktert összekeveri:

Függvény szavakOsszekeverese(szoveg):

szavak = szoveg.szavakraBontasa() # A szöveg szavakra bontása

ujSzoveg = üresLista # Üres lista létrehozása az új szövegnek

minden szó a szavak között:

ha a szó hossza > 3:

elsőKarakter = szo[0]

utolsóKarakter = szo[-1]

középsőKarakterek = szo[1:-1]

összekevertKözepso = középsőKarakterek.osszekeverese()

ujSzo = elsőKarakter + összekevertKözepso + utolsóKarakter

különben:

ujSzo = szo # Ha a szó hossza <= 3, hagyjuk változatlanul

ujSzoveg.hozzaad(ujSzo) # Az új szót hozzáadjuk az új szöveghez

visszatérés ujSzoveg # Visszaadjuk az új szöveget

Példa használat:

bemenetiSzoveg = "Ez egy példa mondat."

ujSzoveg = szavakOsszekeverese(bemenetiSzoveg)

kiir(ujSzoveg)

Ez a pszeudo kód egy egyszerű megközelítést mutat be arra, hogyan lehet megőrizni a szavak olvashatóságát, miközben az első és utolsó karaktereket változatlanul hagyjuk. Természetesen a pontos implementáció a választott programozási nyelvtől függ, de ez a kód segíthet elindulni!



Copilot

Persze, itt van egy egyszerű Python program, amely megvalósítja ezt a feladatot:

Python

```
def osszekever_szavak(text):
    # Szavak szétválasztása a szóközők mentén
    szavak = text.split()

    # Minden szó első és utolsó karakterének megtartása, többi karakter összekeverése
    uj_szavak = []
    for szo in szavak:
        if len(szo) > 2:
            elso_karakter = szo[0]
            utolso_karakter = szo[-1]
            koztes_karakterek = szo[1:-1]
            osszekevert_koztes_karakterek = ''.join(random.sample(koztes_karakterek, len(koztes_karakterek)))
            uj_szo = elso_karakter + osszekevert_koztes_karakterek + utolso_karakter
            uj_szavak.append(uj_szo)
        else:
            uj_szavak.append(szo)

    # Összekevert szavak visszaillesztése a szövegbe
    uj_text = ' '.join(uj_szavak)
    return uj_text

# Példa használat:
bemeneti_szoveg = "Ez egy példa mondat, amit összekeverünk."
osszekevert_szoveg = osszekever_szavak(bemeneti_szoveg)
print(osszekevert_szoveg)
```

Mesterséges intelligencia által létrehozott kód. Ellenőrizze és körültekintően használja. További információ a GYIK-ről.

Ez a program a bemeneti szöveget szavakra bontja, majd minden szó első és utolsó karakterét megtartja, a többi karaktert pedig összekeveri. A példa bemenet "Ez egy példa mondat, amit összekeverünk." esetén az alábbi kimenetet adhatja: