

# Python Programozás Vizsga

3. félév – Gyakorló feladatsor

Logiscool

Név:	
Dátum:	
Időtartam:	90 perc
Összpontszám:	100 pont
Minimum pontszám:	51 pont (>50%)

## Útmutató:

- A vizsga 15 programozási feladatból áll.
- Minden feladatot Pythonban kell megoldani.
- Figyelj a helyes szintaxisra és behúzásokra!
- Részpontszám jár a részben helyes megoldásokért.
- A megoldásokat külön lapon vagy a számítógépen kell beadni.

## I. rész: Alapműveletek és operátorok (20 pont)

### 1. feladat – Számológép (6 pont)

Írj programot, amely bekér két számot a felhasználótól, majd kiírja:

- az összegüket
- a különbségüket
- a szorzatukat
- az egész osztás eredményét
- a maradékos osztás eredményét
- az első szám második hatványát

### Példa kimenet

```
Add meg az elso szamot: 17
Add meg a masodik szamot: 5
Osszeg: 22
Kulonbseg: 12
Szorzat: 85
Egesz osztas: 3
Maradek: 2
Hatvany: 289
```

**2. feladat – Pénzváltó (7 pont)**

Írj programot, amely bekér egy összeget forintban, majd kiszámítja, hogy az hány darab 10000-es, 5000-es, 2000-es, 1000-es, 500-as, 200-as, 100-as, 50-es, 20-as, 10-es és 5-ös bankjegyre/érmére váltható fel a lehető legkevesebb darabszámmal!

*Tipp: Használd az egész osztást (//) és a maradékos osztást (%)!*

**Példa kimenet**

```
Add meg az osszeget: 28745
10000 Ft: 2 db
5000 Ft: 1 db
2000 Ft: 1 db
1000 Ft: 1 db
500 Ft: 1 db
200 Ft: 1 db
100 Ft: 0 db
50 Ft: 0 db
20 Ft: 2 db
10 Ft: 0 db
5 Ft: 1 db
```

**3. feladat – Tudományos számformátum (7 pont)**

Írj programot, amely bekéri egy szám mantisszáját és kitevőjét (pl.  $3.5 \times 10^{-2}$ ), majd:

- Kiszámítja a tényleges értéket
- Kiírja normál és tudományos formátumban is
- Meghatározza, hogy a szám pozitív, negatív vagy nulla

**Példa kimenet**

```
Add meg a mantisszat: 3.5
Add meg a kitevot: -2
Tenyleges ertek: 0.035
Tudomanyos formatum: 3.5e-02
A szam pozitiv.
```

## II. rész: Vezérlési szerkezetek (25 pont)

### 4. feladat – Osztályzat meghatározása (5 pont)

Írj programot, amely bekér egy pontszámot (0-100), és kiírja a megfelelő osztályzatot:

- 90-100: Jeles (5)
- 75-89: Jó (4)
- 60-74: Közepes (3)
- 40-59: Elégséges (2)
- 0-39: Elégtelen (1)

Ha a pontszám nem 0 és 100 között van, írjon ki hibaüzenetet!

#### Példa kimenet

```
Add meg a pontszamot: 73
Osztalyzat: Kozepes (3)
```

### 5. feladat – Prímszám ellenőrzés (6 pont)

Írj programot, amely bekér egy pozitív egész számot, és eldönti, hogy prímszám-e! A program addig kérjen be számot, amíg pozitív egész számot nem kap!

*Tipp: Egy szám prím, ha csak 1-gyel és önmagával osztható.*

#### Példa kimenet

```
Add meg a szamot: -5
Kerem, pozitiv szamot adj meg!
Add meg a szamot: 17
A 17 primszam.

Add meg a szamot: 24
A 24 nem primszam.
```

### 6. feladat – Szorzótábla (6 pont)

Írj programot, amely kiírja az  $n \times n$ -es szorzótáblát, ahol  $n$ -t a felhasználó adja meg! A számok legyenek szépen igazítva oszlopokba!

*Tipp: Használj egymásba ágyazott ciklusokat és formázott kiírást!*

**Példa kimenet**

```
Add meg a méretet: 5
 1  2  3  4  5
 2  4  6  8 10
 3  6  9 12 15
 4  8 12 16 20
 5 10 15 20 25
```

**7. feladat – Számkitaláló játék (8 pont)**

Írj programot, amely generál egy véletlen számot 1 és 100 között, majd a felhasználónak ki kell találnia! Minden tipp után a program mondja meg, hogy a keresett szám nagyobb vagy kisebb! Számold meg, hány tippből sikerült kitalálni!

Ha a felhasználó eltalálta, írd ki a próbálkozások számát! Használd **for-else** vagy **while-else** szerkezetet: ha 10 próbálkozáson belül nem találja el, írd ki, hogy veszített!

*Tipp: `import random, majd random.randint(1, 100)`*

**Példa kimenet**

```
Gondoltam egy számra 1 es 100 kozott!
Tipped: 50
Nagyobbra gondoltam!
Tipped: 75
Kisebbre gondoltam!
Tipped: 62
Gratulalok! Eltalaltad 3 probalkozasbol!
```

### III. rész: Listák és tuple-ök (20 pont)

#### 8. feladat – Lista statisztikák (6 pont)

Írj programot, amely bekér számokat a felhasználótól (amíg 0-t nem ír be), eltárolja őket egy listában, majd kiírja:

- A lista elemeit
- Az elemek számát
- Az elemek összegét
- Az átlagot
- A legnagyobb és legkisebb elemet
- A listát növekvő sorrendben

#### Példa kimenet

```
Adj meg számokat (0 = vege): 5
Adj meg számokat (0 = vege): 12
Adj meg számokat (0 = vege): 3
Adj meg számokat (0 = vege): 8
Adj meg számokat (0 = vege): 0
Lista: [5, 12, 3, 8]
Elemek szama: 4
Osszeg: 28
Atlag: 7.0
Legnagyobb: 12
Legkisebb: 3
Novekvo sorrendben: [3, 5, 8, 12]
```

#### 9. feladat – Mátrix műveletek (8 pont)

Írj programot, amely létrehoz egy  $3 \times 3$ -as mátrixot a felhasználó által megadott értékekkel, majd:

- Kiírja a mátrixot táblázatos formában
- Kiszámítja a főátló elemeinek összegét
- Kiszámítja minden sor összegét
- Megkeresi a legnagyobb elemet és annak pozícióját

**Példa kimenet**

Add meg a matrix elemeit soronként:

1. sor: 1 2 3  
2. sor: 4 5 6  
3. sor: 7 8 9

Matrix:

1 2 3  
4 5 6  
7 8 9

Foatlo osszege: 15

Sorok osszege: [6, 15, 24]

Legnagyobb elem: 9 (pozicio: 2. sor, 2. oszlop)

**10. feladat – Tuple vs Lista (6 pont)**

Írj programot, amely:

1. Létrehoz egy tuple-t 5 diák nevével
2. Létrehoz egy listát a jegyeikkel
3. Kiírja párosítva a neveket és jegyeket
4. Megpróbálja módosítani a tuple egyik elemét (try-except)
5. A listában kicseréli a legrosszabb jegyet 5-re
6. Kiírja az új jegy-listát

**Példa kimenet**

Diakok: ('Anna', 'Bela', 'Csilla', 'David', 'Eva')

Jegyek: [4, 3, 5, 2, 4]

Anna: 4

Bela: 3

Csilla: 5

David: 2

Eva: 4

A tuple nem modositható! (TypeError)

Uj jegyek: [4, 3, 5, 5, 4]

## IV. rész: Dictionary és stringek (20 pont)

### 11. feladat – Telefonkönyv (7 pont)

Írj programot, amely egy telefonkönyvet kezel dictionary segítségével! A program menüvezérelt legyen:

1. Új kontakt hozzáadása
2. Kontakt keresése név alapján
3. Kontakt törlése
4. Összes kontakt listázása
5. Kilépés

A program addig fusson, amíg a felhasználó ki nem lép!

### Példa kimenet

```
--- Telefonkonyv ---
1. Uj kontakt
2. Kereses
3. Torles
4. Listazas
5. Kilepes
Valasztas: 1
Nev: Kiss Peter
Telefonszam: 06301234567
Kontakt hozzaadva!

Valasztas: 4
Kiss Peter: 06301234567

Valasztas: 2
Keresett nev: Kiss Peter
Telefonszam: 06301234567
```

### 12. feladat – Szógyakoriság számláló (7 pont)

Írj programot, amely bekér egy többsoros szöveget a felhasználótól (üres sor = vége), majd:

- Megszámolja, hány szó van összesen
- Dictionary-ben tárolja, melyik szó hányszor fordul elő
- Kiírja a 3 leggyakoribb szót
- Kiírja a leghosszabb szót

*Tipp: A szavakat alakítsd kisbetűsre az összehasonlításhoz!*

**Példa kimenet**

Írj be szöveget (üres sor = vége):

A kutya és a macska játszik

A kutya szeret játszani

A macska alszik

Összes szó: 12

Szógyakoriság: {'a': 4, 'kutya': 2, 'es': 1, 'macska': 2, ...}

Top 3 leggyakoribb: a (4), kutya (2), macska (2)

Leghosszabb szó: játszani (8 betű)

**13. feladat – Jelszó ellenőrző (6 pont)**

Írj programot, amely bekér egy jelszót és ellenőrzi, hogy megfelel-e a követelményeknek:

- Legalább 8 karakter hosszú
- Tartalmaz legalább egy nagybetűt
- Tartalmaz legalább egy kisbetűt
- Tartalmaz legalább egy számot
- Tartalmaz legalább egy speciális karaktert (!@#\$%^&\*)

Minden feltételnél írd ki, hogy teljesül-e vagy sem!

**Példa kimenet**

Add meg a jelszót: Abc123!

Jelszo ellenorzes:

[X] Legalabb 8 karakter - NEM TELJESUL (7 karakter)

[] Tartalmaz nagybetut

[] Tartalmaz kisbetut

[] Tartalmaz szamot

[] Tartalmaz specialis karaktert

A jelszo NEM felel meg a kovetelmenyeknek!

## V. rész: Függvények és kivételkezelés (15 pont)

### 14. feladat – Rekurzív Fibonacci (7 pont)

Írj egy rekurzív függvényt, amely kiszámítja a Fibonacci sorozat  $n$ -edik elemét!

Írj egy másik függvényt, amely kiírja a Fibonacci sorozat első  $n$  elemét!

A főprogramban kérd be  $n$  értékét, és kezeld a hibás bemeneteket (negatív szám, nem szám) try-except blokkal!

*Fibonacci sorozat: 0, 1, 1, 2, 3, 5, 8, 13, 21, ...*

### Példa kimenet

Hanyadik Fibonacci számot számoljam? 10

A 10. Fibonacci szám: 55

Az első 10 elem: 0, 1, 1, 2, 3, 5, 8, 13, 21, 34

Hanyadik Fibonacci számot számoljam? -5

Hiba: Pozitív számot adj meg!

Hanyadik Fibonacci számot számoljam? abc

Hiba: Ez nem szám!

### 15. feladat – Általános lista feldolgozó (8 pont)

Írj egy programot, amely a következő függvényeket tartalmazza:

1. `szamok_beolvasasa()` – Bekér számokat listába (0 = vége), kezeli ha nem számot írnak be
2. `parosak_szurese(lista)` – Visszaadja a páros számokat (használd list comprehension-t!)
3. `negyzetre_emeles(lista)` – Visszaadja az elemek négyzetét (generátor yield-del!)
4. `statisztika(lista, muvelet)` – Visszaadja az összeget, átlagot vagy szorzatot a művelet paraméter alapján (alapértelmezett: "összeg")

A főprogramban hívd meg az összes függvényt és írd ki az eredményeket!

**Példa kimenet**

```
Adj meg számokat (0 = vege): 4
Adj meg számokat (0 = vege): abc
Hiba! Csak számot adj meg!
Adj meg számokat (0 = vege): 7
Adj meg számokat (0 = vege): 2
Adj meg számokat (0 = vege): 9
Adj meg számokat (0 = vege): 0
```

```
Eredeti lista: [4, 7, 2, 9]
Paros számok: [4, 2]
Negyzetszámok: [16, 49, 4, 81]
Összeg: 22
Átlag: 5.5
Szorzat: 504
```

# Megoldások

## Megoldás

### 1. feladat – Számológép

```
1 a = float(input("Add meg az első számot: "))
2 b = float(input("Add meg a második számot: "))
3
4 print(f"Összeg: {a + b}")
5 print(f"Különbség: {a - b}")
6 print(f"Sorzat: {a * b}")
7 print(f"Egész osztás: {int(a // b)}")
8 print(f"Maradék: {int(a % b)}")
9 print(f"Hatvány: {a ** 2}")
```

## Megoldás

### 2. feladat – Pénzváltó

```
1 osszeg = int(input("Add meg az összeget: "))
2 cimletek = [10000, 5000, 2000, 1000, 500, 200, 100, 50, 20, 10, 5]
3
4 for cimlet in cimletek:
5     darab = osszeg // cimlet
6     osszeg = osszeg % cimlet
7     print(f"{cimlet} Ft: {darab} db")
```

## Megoldás

### 3. feladat – Tudományos számformátum

```
1 mantissza = float(input("Add meg a mantisszat: "))
2 kitevo = int(input("Add meg a kitevót: "))
3
4 ertekek = mantissza * (10 ** kitevo)
5 print(f"Tényleges érték: {ertekek}")
6 print(f"Tudományos formátum: {mantissza}e{kitevo:+03d}")
7
8 if ertekek > 0:
9     print("A szám pozitív.")
10 elif ertekek < 0:
11     print("A szám negatív.")
12 else:
13     print("A szám nulla.")
```

## Megoldás

### 4. feladat – Osztályzat meghatározása

```
1 pont = int(input("Add meg a pontszámot: "))
2
3 if pont < 0 or pont > 100:
4     print("Hibas pontszám!")
5 elif pont >= 90:
6     print("Osztalyzat: Jeles (5)")
7 elif pont >= 75:
8     print("Osztalyzat: Jo (4)")
9 elif pont >= 60:
10    print("Osztalyzat: Kozepes (3)")
11 elif pont >= 40:
12    print("Osztalyzat: Elegseges (2)")
13 else:
14    print("Osztalyzat: Elegtelen (1)")
```

## Megoldás

## 5. feladat – Prímszám ellenőrzés

```
1 while True:
2     szam = int(input("Add meg a számot: "))
3     if szam > 0:
4         break
5     print("Kerem, pozitív számot adj meg!")
6
7 if szam == 1:
8     print("Az 1 nem prímszám.")
9 else:
10    prim = True
11    for i in range(2, int(szam ** 0.5) + 1):
12        if szam % i == 0:
13            prim = False
14            break
15
16    if prim:
17        print(f"A {szam} prímszám.")
18    else:
19        print(f"A {szam} nem prímszám.")
```

## Megoldás

## 6. feladat – Szorzótábla

```
1 n = int(input("Add meg a meretet: "))
2
3 for i in range(1, n + 1):
4     for j in range(1, n + 1):
5         print(f"{i * j:4}", end="")
6     print()
```

## Megoldás

## 7. feladat – Számkitaláló játék

```
1 import random
2
3 szam = random.randint(1, 100)
4 print("Gondoltam egy számra 1 es 100 között!")
5
6 for probalkozas in range(1, 11):
7     tipp = int(input("Tipped: "))
8
9     if tipp == szam:
10        print(f"Gratulalok! Eltalaltad {probalkozas} probalkozasbol!")
11        break
12    elif tipp < szam:
```

```
13         print("Nagyobbra gondoltam!")
14     else:
15         print("Kisebbre gondoltam!")
16 else:
17     print(f"Vesztettel! A szam {szam} volt.")
```

**Megoldás****8. feladat – Lista statisztikák**

```
1  szamok = []
2
3  while True:
4      szam = int(input("Adj meg szamokat (0 = vege): "))
5      if szam == 0:
6          break
7      szamok.append(szam)
8
9  print(f"Lista: {szamok}")
10 print(f"Elemek szama: {len(szamok)}")
11 print(f"Osszeg: {sum(szamok)}")
12 print(f"Atlag: {sum(szamok) / len(szamok)}")
13 print(f"Legnagyobb: {max(szamok)}")
14 print(f"Legkisebb: {min(szamok)}")
15 print(f"Novekvo sorrendben: {sorted(szamok)}")
```

**Megoldás****9. feladat – Mátrix műveletek**

```
1  matrix = []
2  print("Add meg a matrix elemeit soronkent:")
3  for i in range(3):
4      sor = input(f"{i+1}. sor: ").split()
5      matrix.append([int(x) for x in sor])
6
7  print("\nMatrix:")
8  for sor in matrix:
9      print(" ".join(str(x) for x in sor))
10
11 # Foatlo osszege
12 foatlo = sum(matrix[i][i] for i in range(3))
13 print(f"\nFoatlo osszege: {foatlo}")
14
15 # Sorok osszege
16 sorok = [sum(sor) for sor in matrix]
17 print(f"Sorok osszege: {sorok}")
18
19 # Legnagyobb elem
20 max_elem = matrix[0][0]
21 max_poz = (0, 0)
22 for i in range(3):
23     for j in range(3):
24         if matrix[i][j] > max_elem:
25             max_elem = matrix[i][j]
26             max_poz = (i, j)
27 print(f"Legnagyobb elem: {max_elem} (pozicio: {max_poz[0]}. sor, {
    max_poz[1]}. oszlop)")
```

## Megoldás

## 10. feladat – Tuple vs Lista

```
1 diakok = ('Anna', 'Bela', 'Csilla', 'David', 'Eva')
2 jegyek = [4, 3, 5, 2, 4]
3
4 print(f"Diakok: {diakok}")
5 print(f"Jegyek: {jegyek}\n")
6
7 for nev, jegy in zip(diakok, jegyek):
8     print(f"{nev}: {jegy}")
9
10 print()
11
12 try:
13     diakok[0] = "Zoltan"
14 except TypeError:
15     print("A tuple nem modosithato! (TypeError)")
16
17 # Legrosszabb jegy kicserelese
18 min_index = jegyek.index(min(jegyek))
19 jegyek[min_index] = 5
20 print(f"Uj jegyek: {jegyek}")
```

## Megoldás

## 11. feladat – Telefonkönyv

```
1 telefonkonyv = {}
2
3 while True:
4     print("\n--- Telefonkonyv ---")
5     print("1. Uj kontakt")
6     print("2. Kereses")
7     print("3. Torles")
8     print("4. Listazas")
9     print("5. Kilepes")
10
11     valasztas = input("Valasztas: ")
12
13     if valasztas == "1":
14         nev = input("Nev: ")
15         tel = input("Telefonszam: ")
16         telefonkonyv[nev] = tel
17         print("Kontakt hozzaadva!")
18     elif valasztas == "2":
19         nev = input("Keresett nev: ")
20         if nev in telefonkonyv:
21             print(f"Telefonszam: {telefonkonyv[nev]}")
22         else:
23             print("Nem talalhato!")
```

```
24     elif valasztas == "3":
25         nev = input("Torlendo nev: ")
26         if nev in telefonkonyv:
27             del telefonkonyv[nev]
28             print("Torolve!")
29         else:
30             print("Nem talalhato!")
31     elif valasztas == "4":
32         for nev, tel in telefonkonyv.items():
33             print(f"{nev}: {tel}")
34     elif valasztas == "5":
35         break
```

## Megoldás

## 12. feladat – Szógyakoriság számláló

```
1 print("Írj be szöveget (üres sor = vége):")
2 szoveg = ""
3 while True:
4     sor = input()
5     if sor == "":
6         break
7     szoveg += sor + " "
8
9 szavak = szoveg.lower().split()
10 print(f"\nÖsszes szó: {len(szavak)}")
11
12 gyakorisag = {}
13 for szo in szavak:
14     if szo in gyakorisag:
15         gyakorisag[szo] += 1
16     else:
17         gyakorisag[szo] = 1
18
19 print(f"Szógyakoriság: {gyakorisag}")
20
21 # Top 3
22 rendezett = sorted(gyakorisag.items(), key=lambda x: x[1], reverse=True)
23
24 print("Top 3 leggyakoribb:", end=" ")
25 for szo, db in rendezett[:3]:
26     print(f"{szo} ({db})", end=", ")
27 print()
28
29 # Leghosszabb szó
30 leghosszabb = max(szavak, key=len)
31 print(f"Leghosszabb szó: {leghosszabb} ({len(leghosszabb)} betű)")
```

## Megoldás

## 13. feladat – Jelszó ellenőrző

```
1 jelszo = input("Add meg a jelszót: ")
2 specialis = "!@#$%^&*"
3
4 print("\nJelszó ellenőrzés:")
5
6 megfelel = True
7
8 if len(jelszo) >= 8:
9     print("[ ] Legalább 8 karakter")
10 else:
11     print(f"[X] Legalább 8 karakter - NEM TELJESUL ({len(jelszo)} karakter)")
```

```
12     megfelel = False
13
14 if any(c.isupper() for c in jelszo):
15     print("[ ] Tartalmaz nagybetut")
16 else:
17     print("[X] Tartalmaz nagybetut - NEM TELJESUL")
18     megfelel = False
19
20 if any(c.islower() for c in jelszo):
21     print("[ ] Tartalmaz kisbetut")
22 else:
23     print("[X] Tartalmaz kisbetut - NEM TELJESUL")
24     megfelel = False
25
26 if any(c.isdigit() for c in jelszo):
27     print("[ ] Tartalmaz szamot")
28 else:
29     print("[X] Tartalmaz szamot - NEM TELJESUL")
30     megfelel = False
31
32 if any(c in specialis for c in jelszo):
33     print("[ ] Tartalmaz specialis karaktert")
34 else:
35     print("[X] Tartalmaz specialis karaktert - NEM TELJESUL")
36     megfelel = False
37
38 if megfelel:
39     print("\nA jelszo MEGFELEL a kovetelmenyeknek!")
40 else:
41     print("\nA jelszo NEM felel meg a kovetelmenyeknek!")
```

## Megoldás

## 14. feladat – Rekurzív Fibonacci

```
1 def fibonacci(n):
2     if n <= 1:
3         return n
4     return fibonacci(n - 1) + fibonacci(n - 2)
5
6 def fibonacci_sorozat(n):
7     return [fibonacci(i) for i in range(n)]
8
9 try:
10    n = int(input("Hanyadik Fibonacci szamot szamoljam? "))
11    if n < 0:
12        print("Hiba: Pozitiv szamot adj meg!")
13    else:
14        print(f"A {n}. Fibonacci szam: {fibonacci(n)}")
15        print(f"Az elso {n} elem: {'', '.join(map(str, fibonacci_sorozat(n)))}")
16 except ValueError:
17    print("Hiba: Ez nem szam!")
```

## Megoldás

## 15. feladat – Általános lista feldolgozó

```
1 def szamok_beolvasasa():
2     lista = []
3     while True:
4         try:
5             szam = int(input("Adj meg szamokat (0 = vege): "))
6             if szam == 0:
7                 break
8             lista.append(szam)
9         except ValueError:
10            print("Hiba! Csak szamot adj meg!")
11    return lista
12
13 def parosak_szurese(lista):
14    return [x for x in lista if x % 2 == 0]
15
16 def negyzetre_emeles(lista):
17    for x in lista:
18        yield x ** 2
19
20 def statisztika(lista, muvelet="osszeg"):
21    if muvelet == "osszeg":
22        return sum(lista)
23    elif muvelet == "atlag":
24        return sum(lista) / len(lista)
25    elif muvelet == "szorzat":
```

```
26         eredmeny = 1
27         for x in lista:
28             eredmeny *= x
29         return eredmeny
30
31 # Foprogram
32 szamok = szamok_beolvasasa()
33 print(f"\nEredeti lista: {szamok}")
34 print(f"Paros szamok: {parosak_szurese(szamok)}")
35 print(f"Negyzetszamok: {list(negyzetre_emeles(szamok))}")
36 print(f"Osszeg: {statisztika(szamok, 'osszeg')}")
37 print(f"Atlag: {statisztika(szamok, 'atlag')}")
38 print(f"Szorzat: {statisztika(szamok, 'szorzat')}")
```

## Pontozási útmutató

Feladat	Témakör	Pontszám	Részpontok
1	Számológép	6	1 pont / művelet
2	Pénzváltó	7	2p ciklus + 2p logika + 3p helyes kimenet
3	Tudományos formátum	7	3p számítás + 2p formázás + 2p feltétel
4	Osztályzat	5	1p / helyes ág
5	Prímszám	6	2p input + 2p algoritmus + 2p kimenet
6	Szorzó tábla	6	3p ciklusok + 3p formázás
7	Számkitaláló	8	2p random + 3p ciklus + 2p else + 1p számláló
8	Lista statisztikák	6	1 pont / statisztika
9	Mátrix	8	2p beolvasás + 2p kiírás + 2p főátló + 2p max
10	Tuple vs Lista	6	2p létrehozás + 2p try-except + 2p módosítás
11	Telefonkönyv	7	1p / menüpont + 2p ciklus
12	Szógyakoriság	7	2p dict + 2p top3 + 2p leghosszabb + 1p összeg
13	Jelszó ellenőrző	6	1p / feltétel + 1p összesítés
14	Fibonacci	7	3p rekurzió + 2p sorozat + 2p try-except
15	Lista feldolgozó	8	2p / függvény
Összesen:		100	

## Értékelési szempontok

- **Szintaxis:** Helyes Python szintaxis, megfelelő behúzások
- **Működés:** A program a példa bemenetekre a várt kimenetet adja
- **Logika:** Az algoritmus helyes és hatékony
- **Olvashatóság:** Értelmes változónevek, tiszta kód
- **Hibakezelés:** Ahol kérve van, megfelelő try-except blokkok