09-algoritmizálás-02

I. Egyszeres választás

1. Milyen típusú nyelv a Python?

a) Procedurális
b) Objektumorientált
c) Értelmezett
d) *Mindhárom fent említett módon használható* ☑

☑ **Megoldás: d) Mindhárom fent említett módon használható**

Magyarázat: A Python egy általános célú nyelv, amely egyszerre támogatja a procedurális (utasításalapú), objektumorientált (osztályok és objektumok alapú), és értelmezett (nem fordított, hanem interpreter végzi a futtatást) paradigmákat.

❌ a), b), c): Bár mindegyik részben igaz, csak a d) válasz fejezi ki teljeskörűen a Python nyelv jellemzőit.

---

2. Mi történik, ha egy változót használunk anélkül, hogy előtte definiáltuk volna?

a) A Python automatikusan inicializálja 0-val
b) A program fut tovább, de figyelmeztetést kapunk
c) *Hibaüzenetet kapunk és a program leáll* ☑
d) Az értéke None lesz

☑ **Megoldás: c) Hibaüzenetet kapunk és a program leáll**

Magyarázat: Ha a Pythonban egy nem definiált változóra hivatkozunk, NameError típusú hiba lép fel, és a program leáll.

❌ a), b), d): Ezek jellemzőek lehetnek más nyelvekben (pl. JavaScript automatikusan undefined-et adhat), de Pythonban kötelező a deklaráció használat előtt.

---

3. Melyik kulcsszó szükséges Pythonban egy saját függvény definiálásához?

a) function
b) define
c) *def* ☑
d) func

☑ **Megoldás: c) def**

Magyarázat: A def kulcsszó segítségével hozunk létre saját függvényeket Pythonban.

❌ a), b), d): Ezek nem érvényes kulcsszavak Pythonban (pl. function JavaScriptben használt).

---

4. Hogyan kezdődik egy Python komment?

a) //

b) # ☑

c) /

d) –

☑ **Megoldás: b) #**

Magyarázat: Pythonban a # jel indítja az egysoros megjegyzéseket (kommenteket).

❌ a): C/C++ vagy Java jelölés

❌ c), d): Nem használatos a kommentálásra Pythonban

---

5. Melyik adattípus NEM létezik a Pythonban?

a) list

b) set

c) tuple

d) *array* ☑

☑ **Megoldás: d) array**

Magyarázat: A list, set, tuple beépített adattípusok. A array nem önálló típus Pythonban, csak a array modul használatával érhető el, így nem alapértelmezett típus.

❌ a), b), c): Ezek szabványos beépített adattípusok.

---

6. Hogyan lehet egy Python program végrehajtását megszakítani egy adott ponton?

a) stop

b) *break* ☑

c) exit

d) end

☑ **Megoldás: b) break**

Magyarázat: A break a ciklusok futását szakítja meg. Ha a programot teljesen le akarjuk állítani, exit()-et is használhatunk, de itt ciklus megszakításról van szó.

❌ a), d): Nem létező kulcsszavak

❌ c): Ez a teljes program leállítására szolgál, nem csak „egy adott ponton”

---

7. Mi az elif kulcsszó szerepe Pythonban?

a) Egy ciklus befejezését jelzi

b) *Egy többágú feltételes szerkezet közbenső ágát határozza meg* ☑

c) Egy függvény végét jelzi

d) Egy ciklus következő iterációját jelzi

☑ **Megoldás: b) Egy többágú feltételes szerkezet közbenső ágát határozza meg**

Magyarázat: Az elif az if és else közötti feltételes ágakat jelöli.

❌ a), c), d): Ezek más funkcióhoz kapcsolódnak (continue, return, stb.)

---

8. Milyen értéket ad vissza egy függvény alapértelmezetten, ha nincs benne return utasítás?

a) 0

b) False

c) *None* ☑

d) Undefined

☑ **Megoldás: c) None**

Magyarázat: Ha egy függvény nem tartalmaz return utasítást, akkor automatikusan a None objektumot adja vissza.

❌ a), b), d): Nem igaz Pythonra, undefined például JavaScript-re jellemző.

---

9. Hogyan lehet egy Python változó értékét törölni?

a) delete var

b) remove var

c) *del var* ☑

d) unset var

☑ **Megoldás: c) del var**

Magyarázat: A del kulcsszóval törölhetünk egy változót a memóriából.

❌ a), b), d): Nem Python szintaxis

---

10. Melyik parancs állítja meg a végtelen ciklust?

a) return

b) stop

c) *break* ☑

d) continue

☑ **Megoldás: c) break**

Magyarázat: A break megszakítja a ciklust, így a végtelen ciklusból való kilépésre is alkalmas.

❌ a): Függvényből való kilépés

❌ b): Nem létező parancs

❌ d): A ciklus következő iterációjára ugrik, nem állítja le

---

11. Melyek lehetnek a Python adatszerkezetek?

a) *list* ☑

b) *tuple* ☑

c) *dictionary* ☑

d) stack

☑ **Megoldás: a), b), c)**

Magyarázat: A Pythonban a list, tuple és dictionary beépített adatszerkezetek. A stack nem külön adattípus, hanem listából, vagy collections.deque segítségével építhető fel.

❌ d): A „stack” nem natív adattípus, nincs önálló kulcsszava vagy típusa a nyelvben.

---

12. Melyek a Python vezérlési szerkezetek közé tartoznak?

a) *if-else* ☑

b) *for* ☑

c) switch-case

d) *while* ☑

☑ **Megoldás: a), b), d)**

Magyarázat: Pythonban az if-else, for és while valódi vezérlési szerkezetek. A switch-case nem része a nyelvnek, helyette if-elif-else szerkezeteket használunk.

❌ c): Nem támogatott Pythonban.

---

13. Melyik Python operátorok használhatók logikai műveletekre?

a) *and* ☑

b) *or* ☑

c) xor

d) *not* ☑

☑ **Megoldás: a), b), d)**

Magyarázat: A and, or, not logikai operátorok Pythonban. Az xor nem kulcsszóként létezik, de a ^ operátor vagy a operator.xor() használható helyette.

❌ c): Nem érvényes operátor szöveges formában Pythonban.

---

14. Milyen adattípusok lehetnek egy Python listában?

a) Csak számok

b) Csak szövegek

c) *Akármilyen Python adattípus* ☑

d) Csak azonos típusú elemek lehetnek benne

☑ **Megoldás: c) Akármilyen Python adattípus**

Magyarázat: A listák heterogének lehetnek Pythonban, azaz különböző típusú elemeket is tartalmazhatnak egyszerre (pl. szám, szöveg, másik lista, stb.).

❌ a), b), d): Ezek nem igazak Python listákra – nincs típuskorlátozás.

---

15. Hogyan lehet egy Python listát módosítani?

a) *append()* ☑

b) *insert()* ☑

c) *remove()* ☑

d) replace()

☑ **Megoldás: a), b), c)**

Magyarázat:

- append(): új elemet tesz a lista végére
- insert(): adott pozícióba szúr be elemet
- remove(): első előfordulást törli

❌ d): A replace() nem létezik listák esetén, ez str típusra vonatkozó függvény.

---

16. Melyek a Python math moduljának függvényei?

a) *sqrt()* ☑

b) *log()* ☑

c) power()

d) *ceil()* ☑

☑ **Megoldás: a), b), d)**

Magyarázat: A math modul sqrt() (négyzetgyök), log() (logaritmus), ceil() (kerekítés felfelé) függvényeket tartalmazza.

❌ c): power() nem része a math modulnak – a math.pow() vagy a ** operátor használható hatványozásra.

---

17. Mely állítások igazak a Python for ciklusára?

a) Csak előre meghatározott számú iterációt végezhet

b) *Lehetőség van lista vagy más iterálható objektum elemein való végighaladásra* ☑

c) *Egy else ág is tartozhat hozzá* ☑

d) Végtelen ciklusra is képes

☑ **Megoldás: b), c)**

Magyarázat: A for ciklus iterálható objektumon megy végig (lista, string stb.), és lehet else ága is. Végtelen ciklusra nem alkalmas, ahhoz while True szükséges.

❌ a), d): Nem igaz – nem kizárólag előre ismert számú lépésből állhat, és nem képes végtelen ciklusra önmagában.

---

18. Milyen módon lehet egy Python sztringet darabolni?

a) *split()* ☑

b) slice()

c) divide()
d) *partition()* ☑

☑ **Megoldás: a), d)**
Magyarázat:

- split(): szeparátor mentén feldarabolja a sztringet
- partition(): egy megadott szeparátornál bontja három részre

❌ b): A „slice” nem függvény, hanem szintaxis ([start:stop])
❌ c): Nem létezik ilyen sztring-függvény Pythonban

---

19. Milyen kivételkezelő szerkezetek léteznek Pythonban?
a) *try-except* ☑
b) try-catch
c) *try-except-finally* ☑
d) try-throw

☑ **Megoldás: a), c)**
Magyarázat: Pythonban try-except a kivételkezelés alapja, finally pedig mindig lefutó blokk. throw helyett raise kulcsszó van.
❌ b), d): Ezek inkább más nyelvekben (pl. Java, C++) használatosak.

---

20. Hogyan lehet egy fájlt megnyitni Pythonban?
a) *open()* ☑
b) file.open()
c) *with open()* ☑
d) readfile()

☑ **Megoldás: a), c)**
Magyarázat:

- open() alap parancs fájl megnyitásához
- with open() ajánlott forma, automatikusan kezeli a fájl lezárását is

❌ b): Ilyen szintaxis nem létezik
❌ d): Nincs ilyen függvény

---

21. Egy Python függvénynek mindig kell paramétert fogadnia.

❌ **Hamis** ☑

☑ **Magyarázat:** A Pythonban definiálható paraméter nélküli függvény is. Például:
python
def udvozles():
    print("Szia!")
Ez teljesen szabályos. A paraméterek nem kötelezőek, csak akkor szükségesek, ha adatot akarunk átadni a függvénynek.

---

22. A Pythonban az is operátor az objektumok azonosságát vizsgálja.

✔ **Igaz** ☑

☑ **Magyarázat:** Az is operátor azt ellenőrzi, hogy két változó ugyanarra az objektumra mutat-e a memóriában (nem csak az értékük egyezik-e). Például:
python
a = [1, 2]
b = a
c = [1, 2]

print(a is b) # True – ugyanaz az objektum
print(a is c) # False – azonos érték, de más objektum

---

23. A Pythonban a while ciklus a feltétel igazságától függően fut.

✔ **Igaz** ☑

☑ **Magyarázat:** A while ciklus addig ismétlődik, amíg a feltétel igaz (True). Ha a feltétel hamissá válik, a ciklus befejeződik. Ez alapvető működési elv.

---

24. Az and és az or operátorok mindkét oldalán logikai érték kell, hogy szerepeljen.

❌ **Hamis** ☑

☑ **Magyarázat:** Pythonban ezek az operátorok „igazságérték szerint” működnek, de nem kizárólag logikai típusokra. Bármilyen típus használható, és az értékelés az igaz-hamis logika szerint történik.
python
```
print(0 and 5)  # 0 (hamis érték)
print(5 or 0)   # 5 (első igaz érték)
```

---

25. A Python listák indexelése 1-től kezdődik.
❌ **Hamis** ☑

☑ **Magyarázat:** A Python listák 0-tól indexelődnek. Tehát az első elem indexe **0**, nem 1.
python
```
lista = ['alma', 'banán']
print(lista[0])  # 'alma'
```

---

26. A pass kulcsszó egy olyan helyőrző, amely lehetővé teszi, hogy egy blokk üres maradjon.
✔ **Igaz** ☑

☑ **Magyarázat:** A pass kulcsszó szintaktikailag szükséges, ha egy vezérlési szerkezet vagy függvény definíciója még üres, de hiba nélkül futni kell a programnak:
python
```
def fuggveny():
    pass  # később fogjuk megírni a tartalmát
```

---

27. A Pythonban a continue kulcsszó megszakítja a ciklus végrehajtását.
❌ **Hamis** ☑

☑ **Magyarázat:** A continue nem szakítja meg a ciklust, csak kihagyja a ciklus aktuális iterációjának hátralevő részét, és a következő iterációra ugrik.
python
```
for i in range(5):
    if i == 2:
        continue
```

```
    print(i)
```
Ez kihagyja a 2 kiírását, de a ciklus nem áll le.

---

28. Egy Python függvényben definiált változók globálisak.

❌ **Hamis** ☑

☑ **Magyarázat:** A Python függvényen belüli változók **lokálisak**, azaz csak a függvényen belül érhetők el. Ha globálisan is el akarunk érni egy változót, azt global kulcsszóval kell megjelölni.

---

29. A print() függvény több paramétert is elfogadhat kiíratásra.

✔ **Igaz** ☑

☑ **Magyarázat:** A print() függvény tetszőleges számú argumentumot képes fogadni:
python
```
print("Név:", "Anna", "Kor:", 12)
```
Ez szóközökkel elválasztva írja ki az értékeket.

---

30. A Python input() függvény mindig egy sztringet ad vissza.

✔ **Igaz** ☑

☑ **Magyarázat:** Az input() függvény **mindig** str típusú adatot ad vissza, még akkor is, ha számot írunk be. Ha más típusra van szükség, például int, akkor konvertálni kell:
python
```
szam = int(input("Adj meg egy számot: "))
```
