Emelt digitális kultúra érettségi szóbeli B tételekhez kapcsolódó részletes leírások és megoldások:

---

1. tétel: Hibakeresés egy összegzési algoritmusban

Hibás pszeudokód:

```
összeg = 0
LISTA = [1, 2, 3, 4, 5, 6]
Ciklus i = 1-től LISTA hosszáig:
    összeg = összeg + LISTA[i]
KIÍR összeg
```

Hiba:

A ciklus 1-től indul, ezért a `LISTA[1]` az első elemet kihagyja, és a hosszig (`LISTA hossza`) fut, ami túllépi a lista indexelését, hibát okozva.

Helyes pszeudokód:

```
összeg = 0
LISTA = [1, 2, 3, 4, 5, 6]
Ciklus i = 0-tól LISTA hossza - 1-ig:
    összeg = összeg + LISTA[i]
KIÍR összeg
```

---

2. tétel: Pszeudokód kielemzése - Maximumkeresés

Pszeudokód:

```
MAX = LISTA[0]
Ciklus i = 1-től LISTA hossza - 1-ig:
    HA LISTA[i] > MAX:
        MAX = LISTA[i]
KIÍR "A legnagyobb elem: ", MAX
```

Magyarázat:

Az algoritmus a `MAX` változót a lista első elemére állítja, majd végighalad a lista többi elemén. Minden elemnél összehasonlítja az aktuális elemet (`LISTA[i]`) a `MAX` értékkel, és ha az aktuális elem nagyobb, akkor frissíti a `MAX` értékét. Az algoritmus végén a `MAX` változó tartalmazza a legnagyobb elemet.

---

3. tétel: Pszeudokód írás - Két lista metszetének meghatározása

Feladat:

Írj pszeudokódot, amely meghatározza két számokból álló lista metszetét.

Pszeudokód:

```
METSZET = []
Ciklus minden elem A az ELSŐ_LISTA-ban:
    HA A szerepel a MÁSODIK_LISTA-ban:
        METSZET hozzáad A
KIÍR METSZET
```

---

4. tétel: Típusalgoritmus leírása - Másolás

Magyarázat:

A másolás algoritmusa egy lista elemeit egy másik listába másolja, megőrizve az eredeti elemek sorrendjét.

Pszeudokód:

```
MÁSOLAT = []
Ciklus minden elem ELEM az EREDETI_LISTA-ban:
    MÁSOLAT hozzáad ELEM
KIÍR MÁSOLAT
```

---

5. tétel: Hibakeresés egy eldöntési algoritmusban

Hibás pszeudokód:

```
Talált = HAMIS
Ciklus i = 0-tól LISTA hossza - 1-ig:
    HA LISTA[i] = KeresettElem:
        Talált = IGAZ
        Kilépés a ciklusból
HA Talált = HAMIS:
    KIÍR "Az elem nem található."
```

Hiba:

Az egyenlőség-összehasonlítás jelölése hibás; helyesen == jelöléssel történik.

Javított pszeudokód:

```
Talált = HAMIS
Ciklus i = 0-tól LISTA hossza - 1-ig:
    HA LISTA[i] == KeresettElem:
        Talált = IGAZ
        Kilépés a ciklusból
HA Talált == HAMIS:
    KIÍR "Az elem nem található."
```

---

6. tétel: Pszeudokód kielemzése - Páros és páratlan számok szétválasztása

Pszeudokód:

```
PÁROSOK = []
PÁRATLANOK = []
Ciklus AMÍG van még elem a LISTA-ban:
    HA LISTA elem MOD 2 == 0:
        PÁROSOK hozzáad LISTA elem
    KÜLÖNBEN:
        PÁRATLANOK hozzáad LISTA elem
KIÍR "Páros számok: ", PÁROSOK
KIÍR "Páratlan számok: ", PÁRATLANOK
```

Magyarázat:

Az algoritmus végighalad a `LISTA` elemein, és minden elemet a `MOD 2` művelet eredménye alapján sorol be. Ha az elem osztási maradéka 0, akkor páros, és a `PÁROSOK` listába kerül; egyébként a `PÁRATLANOK` listába.

---

7. tétel: Feladatleírás alapján pszeudokód - LKT meghatározása

Feladat:

Írj pszeudokódot, amely kiszámítja két szám legkisebb közös többszörösét (LKT).

Pszeudokód:

```
A = első szám
B = második szám
LKT = A
AMÍG LKT MOD B != 0:
    LKT = LKT + A
KIÍR LKT
```

Magyarázat:

Az algoritmus a `LKT` változót kezdetben az első számra állítja, majd addig növeli az értékét, amíg mindkét szám osztója nem lesz.

---

8. tétel: Típusalgoritmus leírása és pszeudokód - Unió

Feladat:

Magyarázd el az Unió algoritmusát és írj hozzá egy pszeudokódot.

Magyarázat:

Az Unió algoritmus két lista elemeit egyesíti, úgy, hogy csak az egyszer előforduló elemeket tartja meg. A cél, hogy egy lista minden elemét csak egyszer tartalmazza az eredmény.

Pszeudokód:

```
UNIÓ = ELSŐ_LISTA
Ciklus minden elem ELEM a MÁSODIK_LISTA-ban:
    HA ELEM nincs UNIÓ-ban:
        UNIÓ hozzáad ELEM
KIÍR UNIÓ
```

Ez az algoritmus minden elemet hozzáad az `UNIÓ` listához, ami nem szerepelt benne korábban, és az eredmény az egyesített elemek listája lesz.

9. tétel: Hibakeresés egy rendezési algoritmusban (Buborékrendezés)

Hibás Pszeudokód:

```
Ciklus i = 0-tól LISTA hossza - 1-ig:
    Ciklus j = 0-tól i-ig:
        HA LISTA[j] > LISTA[j+1]:
            CSERE LISTA[j] és LISTA[j+1]
```

Hiba:

A belső ciklusban a `j` változónak nem `0`-tól `i`-ig kellene futnia, hanem `LISTA hossza - i - 1`-ig, hogy a még nem rendezett részt hasonlítsa össze.

Helyes Pszeudokód:

```
Ciklus i = 0-tól LISTA hossza - 1-ig:
    Ciklus j = 0-tól LISTA hossza - i - 2-ig:
        HA LISTA[j] > LISTA[j+1]:
            CSERE LISTA[j] és LISTA[j+1]
```

Ez a javított algoritmus biztosítja, hogy a legnagyobb elemek sorban a tömb végére kerüljenek.

---

10. tétel: Pszeudokód kielemzése - Fibonacci-sorozat

Pszeudokód:

```
FIBONACCI = [0, 1]
Ciklus i = 2-től N-ig:
    ÚjElem = FIBONACCI[i-1] + FIBONACCI[i-2]
    FIBONACCI hozzáad ÚjElem
KIÍR FIBONACCI
```

Elemzés:

Ez az algoritmus a Fibonacci-sorozat első `N` elemét generálja. A sorozat kezdeti értéke `0` és `1`. Minden további elem az előző két elem összege, amelyet a listához adunk. A `Ciklus` biztosítja, hogy minden elem számítása után hozzáadásra kerüljön a sorozathoz, végül kiírva azt.

---

11. tétel: Tömb elemeinek átlaga

Feladat: Írj pszeudokódot, amely kiszámítja egy számokból álló tömb elemeinek átlagát.

Pszeudokód:

```
Összeg = 0
Ciklus i = 0-tól LISTA hossza - 1-ig:
    Összeg = Összeg + LISTA[i]
Átlag = Összeg / LISTA hossza
KIÍR Átlag
```

Magyarázat:

A `Ciklus` végigmegy a tömb minden elemén, és egy `Összeg` változóban tárolja ezek összegét. A ciklus után az átlagot az `Összeg` és az elemek számának hányadosaként számítjuk ki, amit azután kiírunk.

---

12. tétel: Összegzés algoritmus

Feladat: Magyarázd el az összegzés algoritmusának működését és készíts hozzá pszeudokódot.

Pszeudokód:

```
Összeg = 0
Ciklus i = 0-tól LISTA hossza - 1-ig:
    Összeg = Összeg + LISTA[i]
KIÍR Összeg
```

Magyarázat:

Ez az algoritmus minden egyes `LISTA` elemet hozzáad egy `Összeg` változóhoz, amely a ciklus végére tartalmazza a lista összes elemének összegét.

---

13. tétel: Hibakeresés egy kiválasztási algoritmusban

Hibás pszeudokód:

```
LEGNAGYOBB = LISTA[0]
INDEX = 0
Ciklus i = 1-től LISTA hosszáig:
    HA LISTA[i] > LEGNAGYOBB:
        LEGNAGYOBB = i
KIÍR "A legnagyobb elem indexe: ", INDEX
```

Hiba:

A `LEGNAGYOBB = i` sorban a legnagyobb elem helyett az indexet menti el. A `INDEX` változót kell frissíteni az `i` értékével, ha egy nagyobb elem található.

Javított pszeudokód:

```
LEGNAGYOBB = LISTA[0]
INDEX = 0
Ciklus i = 1-től LISTA hossza - 1-ig:
    HA LISTA[i] > LEGNAGYOBB:
        LEGNAGYOBB = LISTA[i]
        INDEX = i
KIÍR "A legnagyobb elem indexe: ", INDEX
```

---

14. tétel: Pszeudokód kielemzése - Prímszámok keresése

Pszeudokód:

```
PRÍMEK = []
Ciklus i = 2-től N-ig:
    Prím = IGAZ
    Ciklus j = 2-től i/2-ig:
        HA i MOD j == 0:
            Prím = HAMIS
            Szakítsd meg a ciklust
    HA Prím:
        PRÍMEK hozzáad i
KIÍR PRÍMEK
```

Magyarázat:

Az algoritmus végigmegy a 2 és N közötti számokon, és mindegyik számot teszteli, hogy osztható-e 2 és `i/2` közötti számokkal. Ha talál olyan `j` osztót, amelyre `i MOD j == 0`, akkor a szám nem prím. Az algoritmus az összes prím számot hozzáadja a `PRÍMEK` listához.

---

15. tétel: Feladatleírás alapján pszeudokód - Tömb fordított sorrendbe rendezése

Pszeudokód:

```
TOMB = [elemek]
FORDÍTOTT = []
Ciklus i = TOMB hossza - 1-től 0-ig lépésenként -1:
    FORDÍTOTT hozzáad TOMB[i]
KIÍR FORDÍTOTT
```

Magyarázat:

Az algoritmus a `TOMB` elemeit a végéről az elejére haladva hozzáadja egy `FORDÍTOTT` listához, így fordított sorrendben rendezi őket.

---

16. tétel: Típusalgoritmus leírása és pszeudokód - Kiválogatás

Feladat: Magyarázd el a kiválogatás algoritmusát, és írj hozzá egy pszeudokódot, amely kiválogatja a páratlan számokat egy tömbből.

Magyarázat:

A kiválogatás algoritmusa adott feltétel szerint választja ki egy lista elemeit, és új listába helyezi azokat.

Pszeudokód:

```
PÁRATLANOK = []
Ciklus minden elem ELEM a LISTA-ban:
    HA ELEM MOD 2 != 0:
        PÁRATLANOK hozzáad ELEM
KIÍR PÁRATLANOK
```

---

17. tétel: Hibakeresés egy megszámlálás algoritmusban

Hibás pszeudokód:

```
NULLÁK = 0
LISTA = [0, 1, 0, 1, 1, 0, 1]
Ciklus i = 0-tól LISTA hosszáig:
    HA LISTA[i] == 0:
        NULLÁK = NULLÁK + 1
KIÍR "Nullák száma: ", NULLÁK
```

Hiba:

A ciklus hossza egy elemmel túl nagy, mert `LISTA hossza` helyett `LISTA hossza - 1` kell.

Javított pszeudokód:

```
NULLÁK = 0
LISTA = [0, 1, 0, 1, 1, 0, 1]
Ciklus i = 0-tól LISTA hossza - 1-ig:
    HA LISTA[i] == 0:
        NULLÁK = NULLÁK + 1
KIÍR "Nullák száma: ", NULLÁK
```

---

18. tétel: Pszeudokód kielemzése - Többször előforduló elemek szűrése

Pszeudokód:

```
ELEMEK = [1, 2, 2, 3, 4, 4, 4, 5]
EGYEDI = []
Ciklus minden elem ELEM az ELEMEK-ben:
    HA ELEM nem szerepel EGYEDI-ben:
        EGYEDI hozzáad ELEM
KIÍR "Egyedi elemek: ", EGYEDI
```

Magyarázat:

Az algoritmus minden elemet hozzáad az `EGYEDI` listához, ha az még nem szerepelt benne. Így a duplikált elemek elkerülésével az összes egyedi elem szerepelni fog az `EGYEDI` listában.

---

19. tétel: Feladatleírás alapján pszeudokód - Legnagyobb és legkisebb elem keresése

Feladat: Írj pszeudokódot, amely megtalálja és kiírja egy tömb legnagyobb és legkisebb elemét.

Pszeudokód:

```
LEGNAGYOBB = LISTA[0]
LEGKISEBB = LISTA[0]
Ciklus i = 1-től LISTA hossza - 1-ig:
    HA LISTA[i] > LEGNAGYOBB:
        LEGNAGYOBB = LISTA[i]
    HA LISTA[i] < LEGKISEBB:
        LEGKISEBB = LISTA[i]
KIÍR "Legnagyobb elem: ", LEGNAGYOBB
KIÍR "Legkisebb elem: ", LEGKISEBB
```

---

20. tétel: Típusalgoritmus leírása és pszeudokód - Eldöntés

Feladat: Magyarázd el az eldöntés algoritmusát, és írj hozzá egy pszeudokódot, amely eldönti, hogy egy adott szám szerepel-e egy listában.

Magyarázat:

Az eldöntés algoritmusának célja, hogy megvizsgálja, létezik-e a keresett elem a listában.

Pszeudokód:

```
TALÁLT = HAMIS
Ciklus minden elem ELEM a LISTA-ban:
    HA ELEM == KERESETT:
        TALÁLT = IGAZ
        Kilépés a ciklusból
HA TALÁLT:
    KIÍR "Az elem megtalálható."
KÜLÖNBEN:
    KIÍR "Az elem nem található."
```
