Programtervezési ismeretek

gyakorlat

1. Halmazok, számrendszerek

1. Tekintsük az $A = \{p, q, r\}$ és $B = \{5, 8\}$ halmazokat!
   - Írjuk fel az $A \times A, A \times B, B \times A, B \times B$ Déscartes szorzatokat!
   - Írjuk fel a 2^A, 2^B halmazokat!
   - A halmazok kiszámítása előtt határozzuk meg azok elemszámát!

   A Déscartes szorzat nem kommutatív, asszociatív.

   A párok tagjai nem cserélhetők fel, viszont a párok a halmazon belül igen.

   A 2^A helyett szokták még a $\mathcal{P}(A)$ jelölést használni (mint *power*).

2. Számítsuk ki az $52 \cdot 149$ szorzatot orosz-paraszt módszerrel!
   - Ellenőrízzük kézzel kiszámítva a szorzatot!
   - Számítsuk ki $149 \cdot 52$ formában is!

szorzandó	szorzó	szorzó páratlan?	szorzat
52	149	igen	0 + 52
104	74	nem	52
208	37	igen	52 + 208 = 260
416	18	nem	260
832	9	igen	260 +832 = 1092
3328	2	nem	1092
6656	1	igen	1092 + 6656 = **7748**
	0		

   Megvitatni, hogy ez a módszer egyszerűbb, vagy bonyolultabb-e, mint a hagyományos szorzási módszer. $\Rightarrow$ *Gépeknek egyszerűbb*

   Számítsuk ki a $63 \cdot 154$ szorzatot! $\Rightarrow$ 9702

3. Számítsuk ki a következő kifejezések értékét!

$$\lfloor x \rfloor = \max_{k \le x} k, \quad x \in \mathbb{R}, k \in \mathbb{Z} \qquad \lceil x \rceil = \min_{k \ge x} k, \quad x \in \mathbb{R}, k \in \mathbb{Z}$$

$$\mathrm{Round}(x) = \left\lfloor x + \frac{1}{2} \right\rfloor, \quad x \in \mathbb{R} \qquad \{x\} = x - \lfloor x \rfloor, \quad x \in \mathbb{R}$$

$$a \text{ div } b = \left\lfloor \frac{a}{b} \right\rfloor \qquad a \bmod b = \begin{cases} a, & \text{ha } b = 0, \\ a - \left\lfloor \frac{a}{b} \right\rfloor \cdot b, & \text{ha } b \ne 0. \end{cases} \qquad a \bmod 1 = \{a\}$$

   Mi a függvények értelmezési tartománya és értékkészlete?

   - Egészrész függvény: $\lceil 8.1 \rceil$, $\lfloor 0.2 \rfloor$, $\left\lfloor \frac{52}{7} \right\rfloor$, $\lfloor -6.5 \rfloor$, $\lceil -7.51 \rceil$

- Kerekítő függvény: Round(8.1), Round(0.2), Round$\left(\left\lfloor \frac{52}{7} \right\rfloor\right)$, Round(−6.5), Round(−7.51)
- Törtrész függvény: {8.1}, {0.2}, $\left\{\frac{52}{7}\right\}$, {−6.5}, {−7.51}
- Egész hányados képzése, div művelet: 27 div 5, 12 div 0, 21 div -9, -11 div 3, -20 div -7
- Egész maradék képzése, mod művelet: 27 mod 5, 12 mod 0, 21 mod -9, -11 mod 3, -20 mod -7

Elegendő visszahelyettesíteni a definíciókba.

Megnézni az előjeles és előjel nélküli változatok eredményeit, például $31 \div 7$ és $31 \mod 7$ esetén!

4. Egész szám adatstruktúra

$$c_n, c_{n-1}, \ldots, c_1, c_0 \quad 0 \leq c_k < b, b \geq 2$$

$$x = c_n \cdot b^n + c_{n-1} \cdot b^{n-1} + \cdots + c_1 \cdot b^1 + c_0 \cdot b^0 = \sum_{i=1}^{n} c_n \cdot b^n$$

- Milyen értéket jelölnek a következő számalakok?

$$7352_{10}, 10111010_2, 25136_7, 8FB2_{16}, 121020_3, 444_5, 5566_8$$

  Felírni az $A - F$ hexadecimális számjegyek megfelelőit!

- Írjuk fel a 2016-ot 2, 3, 7, 8, 10, 11, 16-os számrendszerben!
  Megmutatni a 2, 4, 8 és 16 számrendszerek közötti könnyebb átjárhatóságot!

- Ellenőrízzük helyiértékesen és Horner sémával!

5. Adjuk össze a következő értékeket a megfelelő számrendszerekben!

$$1011101_2 + 11001_2$$
$$2133_4 + 321_4$$
$$43705_8 + 1162_8$$
$$30112_5 + 4040_5$$

6. Mennyi számjegyből fog állni a 62982 szám 2, 3, 6, 10, 15, 16-os számrendszerben?

$$n + 1 = \lfloor \log_b x \rfloor + 1$$

Hangsúlyozni, hogy a számjegyek száma az $n + 1$.

Kitérni arra, hogy hogyan lehet tetszőleges természetes szám alapú logaritmust számolni.

* Megmutatni, hogy az orosz-paraszt módszer táblázatában a sorok számát ezzel lehet számolni, mivel a harmadik oszlop a kettes számrendszerbeli alakot adja, csak más reprezentációval.

2. Számábrázolás, negatív és lebegőpontos számok

Feladatok az előző témakörhöz

- $A = \{2, 3, 4\}, B = \{p, q\}, |A^7 \times B^8| = ?$
- Az alsó- és felsőegészrész függvény ábrázolása.
- A törtrész függvény ábrázolása.
- Megvizsgálni, hogy teljesül-e a következő:

$$\text{Round}(x) = \left\lfloor x + \frac{1}{2} \right\rfloor = \left\lceil x - \frac{1}{2} \right\rceil$$

1. Egészek ábrázolása

   Számábrázolási tartományok

byte-ok száma	előjel nélküli	előjeles
1	$[0, 255]$	$[-128, 127]$
2	$[0, 65535]$	$[-32768, 32767]$
3	$[0, 2^{24} - 1]$	$[-2^{23}, 2^{23} - 1]$
4	$[0, 2^{32} - 1]$	$[-2^{31}, 2^{31} - 1]$
k bit esetén	$[0, 2^k - 1]$	$[-2^{k-1}, 2^{k-1} - 1]$

   A 16777215 az a kijelzőknél emlegetett 17 millió színt jelöli.

   A 2^{32} a harminckét bites címzés miatt lehet ismerős.

2. Negatív egészek ábrázolása, kettes komplemens

   Bemutatni az algoritmust az 1011100_2 értéken.

   (Ezen azért jó, mert kicsi a számérték, vannak a végén nullák, és nem tölt ki egy byte-ot.)

   Megmutatni, hogy az összeg valóban 0-át ad (egy átviteli bittel a legnagyobb helyiérték után).

   Mennyi byte szükséges a -281, -512, -14890 számok ábrázolásához?

3. Törtek átírása

   - Írjuk át a 0.110101_2, 0.1201_3 és $0.AB1E_{16}$ értékeket tizes számrendszerbe!
   - Írjuk át a 0.1, 0.15 és 0.625 értékeket kettes és tizenhatos számrendszerbe!

   Megmutatni a helyiértékes és a Horner-sémás átírást is!

4. Váltsuk át a 14889 és -2488 számokat kettes számrendszerbe!

   - Mennyi bitből fog állni?
   - Adjuk össze őket kettes számrendszerben!
   - Az összeget számoljuk vissza 10-es számrendszerbe, és ellenőrízzük!

5. Írjuk fel kettes számrendszerbe a következő értékeket!

$$0.42578125_{10}, \quad 0.70312_{10}, \quad 0.74_{10}, \quad 0.14062_{10}, \quad 0.15_{10}, \quad 0.ABCD_{16}$$

   Ellenőrízzük helyiértékesen és Horner sémával is!

6. Lebegőpontos számok ábrázolása!

   Ábrázoljuk egyszeres lebegőpontos számábrázolással a következő értékeket!

$$149.765625, \quad -0.7, \quad 7513.625, \quad 0.41, \quad 9.7$$

7. Számítsuk ki, hogy milyen értékeket ábrázolnak a következő byte-ok egyszeres lebegőpontos ábrázolást feltételezve!

$$8AC20000, \quad FE90C000, \quad 14011000, \quad C5D20000, \quad 7D3C8000$$

8. Ha egy szám 61 jegyű kettes számrendszerben, akkor hány jegyű lesz 16 és 10-es számrendszerben?

   Ha egy szám 9 jegyű 16-os számrendszerben, akkor hágy jegyű lesz 10-esben és kettesben?

9. Mennyi byte-on lehet ábrázolni a Neptun kódokat?

   Mi lenne, ha nem lennének benne számjegyek?

10. Egyszeres pontosságú lebegőpontos számábrázolás

    -51000.3046875, 0.9, 3.6, 4.78125, 3.141592 85EA0000, ABCD0000, FCA20000, 90909090

11. Adjuk meg a végtelen értékeket egyszeres pontossággal!

    Milyen érték a 7F801000?

12. Egyszeres pontossággal milyen értékeket ábrázolnak a következő byte-ok?

$$661B0C42, \quad 661B0A1F$$

13. Töltsük ki az alábbi táblázatot!

duplaszó hexában	FA00AB87	10203040	DEADBEEF
előjel nélküli előjeles egyszeres lebegőpontos			

3. Logikai értékek és műveletek

Feladatok az előző témakörhöz

- Ábrázoljuk a -75.1796875 értéket egyszeres lebegőpontos számábrázolással!

  Hexadecimálisan a `C2965C00` byte-okon ábrázolható.

- Milyen számértéket ábrázolnak a `BB6D7000` byte-ok egyszer lebegőpontos ábrázolást feltételezve?

  A -0.0036230087 értéket ábrázolja. (A kitevő -9, az eredmény $\frac{\mathrm{ED7}_{16}}{2^{20}}$ formában számolható.)

- Írjuk fel egyszeres lebegőpontos számábrázolással a 0-át és a végtelent, illetve adjunk példát *denormalizált* és *nem szám*ra.

+0	`00 00 00 00`
-0	`80 00 00 00`
+Inf	`7F 80 00 00`
-Inf	`FF 80 00 00`
Den	`00 01 00 00`
NaN	`7F 80 10 00`

  $0 \rightarrow$ denormalizált, ha nem nulla a szignifikáns.

  $\infty \rightarrow$ NaN, ha nem nulla a szignifikáns.

1. Logikai műveletek

- Milyen logikai műveleteket ismerünk?
- Melyek unárisak és melyek binárisak ezek közül?
- Mennyi lehetséges unáris és bináris művelet van?
- Írjuk fel a logikai műveletek művelettábláit!
  (Az $\wedge, \vee, \oplus, \leftrightarrow, \rightarrow, \downarrow, |$ műveleteket külön megnézni!)

A művelettáblákba itt még elég, ha i és h betűk kerülnek.

2. Írjuk fel a 3 bemenetes többségi szavazás diszjunktív normál formáját, és rajzoljuk fel a kapuáramkört!

Jelölje a, b és c az egyes szavazatokat. Ekkor a szavazógép igazságtáblája a következőképpen néz ki.

a	b	c	$f(a,b,c)$
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

A diszjunktív normálforma felírásához tekintsük azokat a sorokat, amelyekben $f(a,b,c) = 1$. (Ezek a 4., 6., 7. és 8. sorok.) Írjuk fel elemi konjunkciókat a változókból úgy, hogy ahol a változó értéke 0 az adott sorban, ott a változó negáltját szerepeltessük. Az elemi konjunkciók a következők lesznek.

$$\overline{a} \wedge b \wedge c, \quad a \wedge \overline{b} \wedge c, \quad a \wedge b \wedge \overline{c}, \quad a \wedge b \wedge c$$

A normál diszjunktív normálformát ezeknek az elemi konjunkcióknak a diszjunkciójából kapjuk.

$$f(a,b,c) = (\overline{a} \wedge b \wedge c) \vee (a \wedge \overline{b} \wedge c) \vee (a \wedge b \wedge \overline{c}) \vee (a \wedge b \wedge c)$$

3. Fejezzük ki az antivalencia, implikáció és az ekvivalencia műveleteket az $\wedge$, $\vee$ és negáció műveletekkel!

$$\begin{aligned} x \oplus y &\equiv (\overline{x} \wedge y) \vee (x \wedge \overline{y}) \\ x \rightarrow y &\equiv \overline{x} \vee y \\ x \leftrightarrow y &\equiv (x \wedge y) \vee (\overline{x} \wedge \overline{y}) \end{aligned}$$

Érdemes inkább levezetni őket igazságtáblával!

4. Tervezzünk 5 bemenetes automatát, amely a maximumot adja vissza!

- Fel kell rajzolni az 5 bemenetes kapuáramkör be- és kimenetét.
- Meg kell nézni, hogy 2 bemenet elején hogy nézne ki.
- Kettessével össze kell vagyolni a bemeneteket.
- Ki kell írni a kifejezést, és a kapuáramkörnek megfelelően be kell zárójelezni.
- Meg lehet mutatni, hogy a minimum is hasonlóképpen alakulna.

5. Tervezzünk kapuáramkört az $\wedge$, $\vee$ és negáció műveletekkel az $u = f(x, y, z) = (x \oplus y) \to z$ logikai függvényhez és írjuk fel a diszjunktív normál formáját!

x	y	z	$x \oplus y$	$(x \oplus y) \to z$
0	0	0	0	1
0	0	1	0	1
0	1	0	1	0
0	1	1	1	1
1	0	0	0	1
1	0	1	0	1
1	1	0	1	0
1	1	1	1	1

A diszjunktív normálforma felírásához a 3. és 7. sort leszámítva minden sort figyelembe kell venni. A DNF a következő lesz.

$$f(x, y, z) = (\overline{x} \wedge \overline{y} \wedge \overline{z}) \vee (\overline{x} \wedge \overline{y} \wedge z) \vee (\overline{x} \wedge y \wedge z) \vee (x \wedge \overline{y} \wedge \overline{z}) \vee (x \wedge \overline{y} \wedge z) \vee (x \wedge y \wedge z)$$

4. gyakorlat

1. Készítsünk egész összeadót félösszeadókból!

   - Érdemes felírni a félösszeadó és a teljes összeadó igazságtábláját, és felrajzolni hozzá a kapuáramköröket!

2. Írjuk fel az $\wedge$, $\vee$, $\neg$, $\to$, $\oplus$, $\leftrightarrow$, $\downarrow$ műveleteket *Scheffer vonás*sal, és rajzoljuk fel a kapuáramköröket *NAND* kapukkal!

   Az $\oplus$ ábráját egyben felrajzolni úgy, hogy utána az élekre lehessen írni, hogy mi az aktuális részeredmény, és így belátni, hogy valóban jó eredményt ad!

3. Lássuk be a következőket!

   Felírni hozzá a műveletekre vonatkozó azonosságokat!

   Megemlíteni, hogy igazszágtáblával is lehetne bizonyítani, viszont az több változó esetén már nem lenne praktikus.

   $x \wedge (y \oplus z) \equiv (x \wedge y) \oplus (x \wedge z)$

   $(p \wedge q \wedge r) \to s \equiv p \to (q \to (r \to s))$

   $(p \wedge (p \to q)) \to q \equiv 1$

   $(a|b) \oplus (a \downarrow b) \equiv a \oplus b$

4. Igazoljuk a következő egyenlőségeket!

   $\overline{A \setminus B} \cap (\overline{A} \cup \overline{B}) = \overline{A}$

   $A \cap (\overline{A \setminus B}) = A \cap B$

   $\overline{(P \setminus Q) \mathbin{\triangle} R} = (\overline{R} \cup P) \cap (\overline{R} \cup \overline{Q}) \cap (\overline{P} \cup Q \cup R)$

   $A \setminus (B \setminus (C \setminus D)) = A \cap (\overline{B} \cup \overline{C} \cup D)$

5. gyakorlat

1. Tervezzünk 4 bemenetes paritás ellenörző automatát!
2. Vizsgáljuk meg az alábbi azonosságokat!
   $a \rightarrow ((b|a) \wedge \overline{b}) \equiv a$
   $\overline{a \wedge \overline{\overline{b \wedge \overline{c \wedge d}}}} \equiv ?$
   $\overline{(x \oplus y) \rightarrow z} \equiv (x \wedge \overline{y} \wedge \overline{z}) \vee (\overline{x} \wedge y \wedge \overline{z})$
   $(a|b) \downarrow (c|d) \equiv (d|a) \downarrow (c|b)$
3. Írjuk fel az $f(x_1, x_2, x_3) = (x_1 \oplus x_3) \wedge (x_2 \rightarrow x_1)$ DNF-jét! Rajzoljunk kapuáramkört!
   ...hasonlóképp a $g(x_1, x_2, x_3) = (x_2 \vee x_3) \leftrightarrow (x_1 \wedge x_2)$-vel.
4. Mennyi olyan halmazpár van, amelyeknek a metszete $\{a, b, c\}$ uniója pedig $\{a, b, c, d, e, f\}$?
5. Igazoljuk a következő egyenlőségeket!
   $C \cap (A \triangle B) = A \cap \overline{B} \cap C$, ha B és C diszjunkt.
   $(K \cap L) \setminus (K \setminus M) = K \cap L \cap M$
   $(K \setminus L) \setminus M = (K \setminus M) \setminus (L \setminus M)$
   $\overline{P \triangle Q} \setminus R = (\overline{P} \cup Q) \cap (P \cup \overline{Q}) \cap \overline{R}$
6. Tekintsük az $A = (-5; 4]$ és a $B = [3; 9)$ intervallumokat. Határozzuk meg a következő kifejezések értékét: $A \cup B, A \cap B, A \setminus B, B \setminus A, A \triangle B, A \times B, 2^A, 2^B$.
7. Írjuk fel a $2^{2^{\{a,b\}} \setminus \{b\}}$ halmaz elemeit!

   Van-e jelentősége annak, hogy $\{b\}$ szerepel, nem pedig csak b?
8. Ha $|H| = 60, A \subset H, B \subset H, |A| = 48, |B| = 32$ akkor mennyi lesz $|A \cup B|$ és $|A \cap B|$?

6. gyakorlat

1. Karakterkódolás
   - Fix és változó hosszúságú kódolás előnyei, hátrányai.
   - UTF-8 kódolás, tárolási mód
   - Nem egyértelmű megadás (szabvány szerint tiltott)
   - Szövegek átalakítása számokká.
2. Alakítsuk át a következő unicode szimbólumokat UTF-32 és UTF-8 kódolásúra!

   U+0048, U+082C, U+10EE65, U+9AF1

   Átírni bináris alakba. Megszámolni, hogy mennyi bit szükséges az ábrázoláshoz. Meghatározni az alapján a byte-ok számát. Felrajzolni a keretet formátumnak megfelelően. Kitölteni a bit-eket. Visszaírni hexadecimális alakba a byte-okat.
3. Adjuk meg az első és az utolsó érvényes unicode kódot a byte-ok számának megfelelően!

byte-ok száma	felhasználható bitek száma	első kód	utolsó kód
1	7	U+0000	U+007F
2	11	U+0080	U+07FF
3	16	U+0800	U+FFFF
4	21	U+10000	U+1FFFF
5	26	U+20000	U+3FFFFFF
6	31	U+4000000	U+7FFFFFFF

4. Szövegláncok/sztringek

   Hogyan tudunk sztringeket tárolni? *(C, Pascal, előnyök, hátrányok)*

   Felírni egymás mellé a két fajta konvenciót, és a következő szempontokat érdemes megtárgyalni:

   - szövegek maximális hossza,
   - hossz meghatározása,
   - ábrázolható karakterek,
   - konkatenálás költsége,
   - hibatűrés (pl.: termináló 0 lemarad, vagy több van).

   Szövegszerkesztő megvalósítása. → A karakter beszúrásánál vagy törlésénél az egész buffer átméretezése túl költséges lenne.

   Milyen attribútumai és műveletei vannak a sztringeknek?

   Az attribútumos jelölésmódra külön kitérni!

   A konkatenáció művelete kommutatív, asszociatív-e?

   Nem. Igen.

5. Adott egy tömb a memóriában, amelynek elemei 48 bájtosak. Az 50-edik elem a tömbelem 9. bájtjának 5. bitje hanyadik bitje lesz a tömbnek?

   - Vizsgáljuk külön a 0 és 1 kezdőindexek eseteit!
   - Hogyan tudnánk felírni általánosan, ha m bájtos a tömb, i-edik tömbelem j-edik bájtjának k-adik bitjéről van szó?

6. Sorozatok. Sor- és oszlopfolytonos tárolási mód.

   Felírni, hogy általánosan hogyan néznek ki a tárolási módok!

   @: Címe operátor/címképző operátor (at, hasonló a C nyelv `&` operátorához)

   - vektor: $@(v_k) = a + (k-1) \cdot h$
   - sorfolytonos tárolás: $@(a_{ij}) = a + [(i-1) \cdot n + j - 1] \cdot h$
   - oszlopfolytonos tárolás: $@(a_{ij}) = a + [(j-1) \cdot m + i - 1] \cdot h$

   - Hogyan tudjuk az indexből visszaszámolni a sort és az oszlopot?
   - Hogyan számíthatjuk át az indexet sor- és oszlopfolytonos tárolási mód között?

7. Írjuk fel általánosan az index kiszámítási módját a következő sematizált esetekre!

1	8	9	16
2	7	10	15
3	6	11	14
4	5	12	13

1			
2	3		
4	5	6	
7	8	9	10

1	2	6	7
3	5	8	13
4	9	12	14
10	11	15	16

   Megemlíteni, hogy az utolsót hol használják.

8. Hogyan oldható meg az egységmátrix, diagonális mátrix és a szimmetrikus mátrix tárolási módja?

9. Hogyan tudjuk az indexeket átszámolni egy mátrix részmátrixaira?

10. Hogyan tudunk többdimenziós tömböket tárolni?

11. Személyek adatait szeretnénk névvel, Neptun kóddal és születési dátummal tárolni. Hogyan tudjuk ezt megoldani rekordok segítségével?

    Felírni a mezőket hozzá, és csinálni egy 2-3 elemű példát.

    Egy tetszőleges byte pozícióról hogyan tudjuk megállapítani, hogy az melyik rekord melyik mezője?

    Egész hányadossal számítható a rekord indexe. A maradék megadja a mezőt.

12. Tekintsünk egy fájlt, amely BMP formátumban tárol egy képet. A BMP formátumnak a fejléce 54 bájtos. Minden pixel 3 byte-ot foglal. Egy pixelen belül RGB komponensek vannak 1-1 byte-on.

    - Mennyi helyet fog foglalni egy 1024×768 felbontású kép?
    - Mi lesz az offsetje a $(10, 20)$ pozícióban lévő pixelnek? (A bal felső pixel a $(0, 0)$, sorfolytonos tárolási mód.)
    - A 100. byte az melyik pixel, melyik komponensének az értékét tartalmazza?

7. gyakorlat

1. Írjunk pszeudó kódot, és rajzoljunk folyamatábrát!

    - Számoljuk meg egy vektorban a páros számok darabszámát!
    - Számoljuk ki a páratlan számok összegét!
    - Határozzuk meg a negatív számok maximumát!
    - Vizsgáljuk meg, hogy egy vektor tartalmaz-e egy bizonyos elemet!
    - Vizsgáljuk meg, hogy egy vektor rendezett-e!
    - Ellenőrizzük, hogy egy vektor minden eleme egyedi!
    - Számoljuk meg, hogy egy bináris vektorban mennyi 0 és 1 érték van!

2. Írjunk pszeudó kódot, és rajzoljunk folyamatábrát!

    - Ellenőrízzük, hogy egy mátrix egységmátrix-e!
    - Ellenőrízzük, hogy egy mátrix diagonális mátrix-e!
    - Ellenőrízzük, hogy egy mátrix szimmetrikus-e!
    - Ellenőrízzük, hogy egy mátrix alsóháromszög mátrix-e!
    - Határozzuk meg egy mátrix legkisebb és legnagyobb elemét!
    - Írjuk ki annak a sornak az indexét, amelyben az elemek összege a legnagyobb!
    - Számoljuk meg, hogy egy mátrixban mennyi 0 érték van.
    - Tároljunk ritka mátrixot vektorok segítségével!
    - Írjuk ki azon elemek értékét, amelyek megegyeznek a sor/oszlop tárolás során a relatív indexszel (*vagyis, ha az offset 1*).

3. Rekurzív függvények

$$f(n) = \begin{cases} 2 \cdot f(n-3), & \text{ha } n > 20, \\ 5, & \text{egyébként.} \end{cases}$$

$$g(n) = \begin{cases} g(n-1) + 4 \cdot g(n-2), & n > 0, \\ 8, & \text{egyébként.} \end{cases}$$

    Számítsuk ki az $f(30)$ és $g(5)$ értékeket!

4. Számítsuk ki egy vektor első 6 elemének szorzatát!

   - Írjuk fel a procedúra be- és kimeneteit!
   - Írjuk fel a procedúra és a főprogram működését lépésenként!
   - Vizsgáljuk a futás közben a verem állapotát!

   Tekintsünk egy valós vektort. Ekkor a bemenet $x \in \mathbb{R}^6$ paraméter lesz. Mivel valós értékeket szorzunk össze, ezért a szorzat szintén valós érték lesz. Jelöljük például $p \in \mathbb{R}$-rel.

   Egy iteratív megoldás a következő lépésekben számíthatja ki a szorzatot.

   1 A p értéket állítsuk 1-re.
   2 Az i indexváltozó értékét állítsuk 1-re.
   3 Vizsgáljuk meg, hogy az i értéke kisebb vagy egyenlő-e, mint 6. Amennyiben igen, akkor térjünk vissza a p értékkel.
   4 A p értéket szorozzuk meg, az x_i értékkel.
   5 Növeljük az i ciklusváltozó értékét 1-gyel.
   6 Folytassuk a program futtatását a 3. ponttól.

   Pszeudokód formájában ez a következőképpen nézhet ki:

```
PROD(@X, @p)
// Input: X ∈ ℝ^6
// Output: p ∈ ℝ, a szorzat
p ← 1
i ← 1
WHILE i ≤ 6 DO
    p ← p · x_i
    INC(i)
RETURN(p)
```

   A procedúrát meghívva a veremben helyet kell foglalni a lokális változóknak, vagyis a p-nek és az i-nek. Mivel iteratív algoritmusról van szó, ezért a lépések végrehajtása során a verem mélysége nem változik. A fután során azt figyelhetjük meg, hogy hogyan változik a p és i értéke.

5. Készítsünk rekurzív procedúrát az előző feladatra!

   - Hogyan változik a verem állapota futás közben?
   - Rajzoljuk fel a rekurzív hívási fát!
   - Mennyi rekurzív hívás történt, és mennyi volt a maximális veremmélység?
   - Milyen előnyei/hátrányai vannak a rekurzív megvalósításnak?

   A rekurzív megoldáshoz érdemes a procedúrát általánosan az $[i, j]$ indexek közé eső értékek szorzatának kiszámítására visszavezetni.

   Legalább 2 féle rekurzív megoldás is elképzelhető:

   a. a kezdő indexen lévő értéket összeszorozzuk az utána lévő elemek szorzatával, vagy
   b. minden lépésben két részre bontjuk a tömböt, amíg tudjuk.

   A rekurzív függvényeknél nagyon fontos a leállási feltétel megadása. Jelen esetben ez lehet az 1 elemű részvektor szorzatának a számítása.

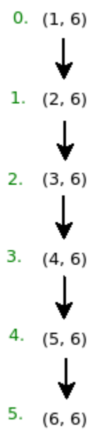
   A rekurzív megoldásra például az alábbi procedúrák használhatóak. Az első változat:

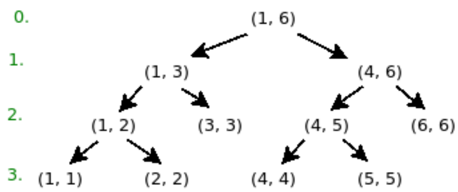
```
PROD_A(@X, i, j, @p)
// Input: X ∈ ℝ^6
//        i, j a kezdo es vegindex
// Output: p ∈ ℝ, a szorzat
IF i = j
   THEN p ← x_i
   ELSE CALL PROD_A(X, i+1, j, q)
        p ← p · q
RETURN(p)
```

A második változat:

```
PROD_B(@X, i, j, @p)
// Input: X ∈ ℝ^6
//        i, j a kezdo es vegindex
// Output: p ∈ ℝ, a szorzat
IF i = j
   THEN p ← x_i
   ELSE m ← ⌊(i + j) / 2⌋
        CALL PROD_A(X, i, m, q_1)
        CALL PROD_A(X, m+1, j, q_2)
        p ← q_1 · q_2
RETURN(p)
```

A hat elem szorzatának kiszámításához az $i = 1$ és $j = 6$ paraméterekkel kell meghívni a procedúrákat. A rekurzív hívási fák:





A csomópontokban az egyszerűség kedvéért csak az (i, j) párok vannak feltüntetve. A hívási fák bal oldalán szerepel a verem aktuális szintje.

Az ábrákból leolvasható, hogy az első változatban 5 rekurzív hívás volt, és a maximális veremmélység is 5 volt. A másik esetében 10 rekurzív hívás történt, viszont a maximális veremmélység csak 3.

A rekurzív változat a második esetében lehet előnyös, ha van módunk egymással párhuzamosan számmíttatni a független ágakat. Így a futási idő az elem számának logaritmusával arányosan fog csak növekedni. (A másik esetében ez lineáris növekedést eredményez.)

6. Számítsuk ki az $\binom{5}{3}$ értékét!

$$\binom{n}{k} = \binom{n-1}{k-1} + \binom{n-1}{k}$$

7. Számítsuk ki a $\sqrt{17}$ értékét az iteratív képletnek megfelelően $\varepsilon = 0.001$ pontosságig!

$$x_{k+1} = \frac{1}{2} \cdot \left(x_k + \frac{s}{x_k} \right)$$

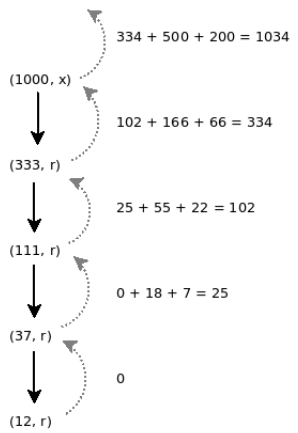
k	x_k	x_k kiszámítása	$\lvert x_k - x_{k-1} \rvert$
0	1	-	-
1	9	$= \frac{1}{2} \cdot \left(1 + \frac{17}{1}\right)$	8
2	5.444	$= \frac{1}{2} \cdot \left(9 + \frac{17}{9}\right)$	3.5556
3	4.2834	$= \frac{1}{2} \cdot \left(5.444 + \frac{17}{5.444}\right)$	1.8884
4	4.1261	$= \frac{1}{2} \cdot \left(4.2834 + \frac{17}{4.2834}\right)$	0.15729
5	4.1231	$= \frac{1}{2} \cdot \left(4.1261 + \frac{17}{4.1261}\right)$	0.003
6	4.1231	$= \frac{1}{2} \cdot \left(4.1231 + \frac{17}{4.1231}\right)$	0.000

A 6. lépésben csak az adott pontosságnak megfelelően lesz a két közelítő érték különbsége 0.

8. Számítsuk ki a Div_Sum(1000, x) hívás esetén az x értékét!

Div_Sum(a, @r)
// Input: $a \in \mathbb{Z}$
// Output: $r \in \mathbb{Z}$
IF a $<$ 20
  THEN $r \leftarrow 0$
  ELSE CALL Div_Sum(a DIV 3, r)
    $r \leftarrow r + (a \text{ DIV } 2) + (a \text{ DIV } 5)$
RETURN(r)

A rekurzív hívási fa a következőképpen néz ki.



Az eredmény kiszámításához 4 rekurzív hívásra volt szükség, és a maximális veremmélység is 4 volt.

9. Írjunk egy procedúrát, amely egy tömbhöz egy olyan tömböt számol ki, melynek i-edik elemén az eredeti tömb i-edik utáni elemeinek összege szerepel!

```
CUMULATIVE(@X, @Y)
// Input: X ∈ ℝ^n
// Output: Y ∈ ℝ^n
Hossz[Y] ← Hossz[X]
y_Hossz[Y] ← 0
FOR i ← Hossz[X] − 1 TO 1 DO
        y_i ← y_{i+1} + x_{i+1}
RETURN(Y)
```

8. gyakorlat

1. Írjunk fel egy rekurzív függvényt a faktoriális kiszámítására!
   - Számítsuk ki az 5! értékét!
   - Mennyi rekurzív hívás volt a számítás során?
   - Mekkora volt a maximális mélysége a hívási veremnek?
   - Rajzoljuk fel a rekurzív hívási fát!

   A rekurzív formula:

$$n! = \begin{cases} n \cdot (n-1)!, & n > 1, \\ 1, & n = 1. \end{cases}$$

   Az 5! számítása:

$$5! = 5 \cdot (5-1)! = 5 \cdot 4! = 5 \cdot (4 \cdot (4-1)!) = 5 \cdot 4 \cdot 3! = 5 \cdot 4 \cdot 3 \cdot 2! = 5 \cdot 4 \cdot 3 \cdot 2 \cdot 1 = 120$$

   A rekurzív függvény pszeudokódja:

```
FACTORIAL(n)
// Input: n ∈ ℕ
// Output: FACTORIAL
IF n > 1
    THEN RETURN(n · FACTORIAL(n − 1)
    ELSE RETURN(1)
```

   A rekurzív hívási fa az 1. ábrán látható. Ebből leolvasható, hogy 4 rekurzív hívás történt, és a maximális veremmélység is 4 volt.

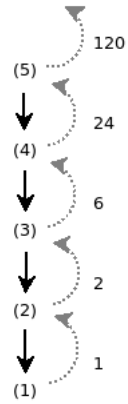
2. Számítsuk ki a $\binom{6}{4}$ értékét a rekurzív formula segítségével! Rajzoljuk fel a rekurzív hívási fát!

3. A bináris keresés algoritmusával keressük meg a 10 értéket a következő tömbben!

$$A = [1, 4, 5, 10, 11, 13, 14, 16, 17]$$

   Az A tömbnek 9 darab eleme van. A bináris keresés algoritmusát célszerű nyílt intervallummal megfogalmazni. Minden lépésben meg kell határozni a középső (m) elemet, majd megnézni, hogy az a keresett elemnél kisebb, vagy nagyobb-e, majd annak megfelelően a bal, vagy a jobb oldali részintervallumra folytatni a keresést.

   Az A tömb esetében a következő lépésekben történik a keresés. Az i az intervallum elejét, a j az intervallum végét, a k pedig a középső elem indexét jelöli.



1. ábra. Az 5! számítása

i	j	k	m	folytatás
0	10	5	11	bal oldalon
0	5	2	4	jobb oldalon
2	5	3	5	jobb oldalon
3	5	4	10	-

9. gyakorlat

1. Készítsünk egy procedúrát, amely megállapítja, hogy az egy vektor elemei egy adott intervallumba esnek-e!

```
BELUL_E(@S, a, b, @belul)
// Input: S ∈ ℝ^n, valos ertekek vektora
//        a,b ∈ ℝ az [a,b] intervallum megadasahoz
// Output: belul ∈ {igaz, hamis}
i ← 1
WHILE i ≤ Hossz[S] and s_i ≥ a and s_i ≤ b DO
    INC(i)
IF i > Hossz[S]
    THEN belul ← igaz
    ELSE belul ← hamis
RETURN(belul)
```

Oldjuk meg minimum és maximum számításával is!

Tegyük fel, hogy van minimum és maximum érték számításához függvényünk. Ekkor a feladat a következő procedúrával is megoldható.

```
BELUL_E(@S, a, b, @belul)
// Input: S ∈ ℝ^n, valos ertekek vektora
//        a,b ∈ ℝ az [a,b] intervallum megadasahoz
// Output: belul ∈ {igaz, hamis}
IF MIN(S) ≥ a and MAX(S) ≤ b
    THEN belul ← igaz
    ELSE belul ← hamis
RETURN(belul)
```

Futási időt tekintve az első változat jobb, mivel csak a szükséges mennyiségű elemet vizsgálja meg. A második megoldás előnye, hogy rövidebb, és így áttekinthetőbb a kód.

2. Ellenőrízzük, hogy egy tömb elemei növekvő sorrendben vannak-e!

   Egy A tömb elemei akkor vannak növekvő sorrendben, ha minden i-re teljesül, hogy $a_i \leq a_{i+1}$. Ezt a feltételt ellenőrízni a következőképpen tudjuk.

```
NOVEKVO_E(@A, @novekvo)
// Input: A ∈ ℝ^n, valos ertekek vektora
// Output: novekvo ∈ {igaz, hamis}
novekvo ← igaz
FOR i ← 1 TO Hossz[A] − 1 DO
          IF a_i > a_{i+1}
               THEN novekvo ← hamis
RETURN(novekvo)
```

   Ebben az esetben is elképzelhető egy egyszerűbb megoldás, aminél nem szükséges feltétlenül az összes elemet végignézni.

   Vizsgáljuk meg a monoton, és szigorúan monoton eseteket is!

   A szigorúan monoton esethez csak annyi változtatásra van szükség, hogy a feltételben a $>$ reláció helyett $\geq$ szerepeljen.

3. Készítsünk *egy* procedúrát, amely kiszámítja két vektor összegét és skaláris szorzatát!

   A procedúra például a következő képpen nézhet ki:

```
SUM_SKALAR(@X, @Y, @S, @prod)
// Input: X, Y ∈ ℝ^n, valos ertekek vektora
// Output: S ∈ ℝ^n, prod ∈ ℝ
Hossz[S] ← Hossz[X]
prod ← 0
FOR i ← 1 TO n DO
          s_i ← x_i + y_i
          prod ← prod + x_i · y_i
RETURN(S, prod)
```

4. Írassuk ki egy vektor értékeinek átlagtól való eltérését!

   Egy lehetséges megoldás:

```
ATLAG_ELTERES(@X, @D)
// Input: X ∈ ℝ^n, valos ertekek vektora
// Output: D ∈ ℝ^n atlagtol elteres
s ← 0
FOR i ← 1 TO n DO
     s ← s + x_i
atlag ← s/n
Hossz[D] ← Hossz[X]
FOR i ← 1 TO n DO
     d_i ← x_i − atlag
RETURN(D)
```

5. Ellenőrízzük, hogy egy vektor minden eleme különböző!

   Függvény az ellenőrzéshez:

```
MIND_KULONBOZO(@X)
// Input: X ∈ ℝ^n, valos ertekek vektora
// Output: MIND_KULONBOZO ∈ {igaz, hamis}
FOR i ← 1 TO n DO
     FOR j ← 1 TO n DO
          IF i ≠ j
               THEN IF x_i = x_j
                         THEN MIND_KULONBOZO ← hamis
                              RETURN
MIND_KULONBOZO ← igaz
RETURN
```

6. Hogyan tudnánk bekérni, tárolni és kiíratni egy alsóháromszögmátrix értékeit?

   A bekérést, tárolást és kiíratást oldjuk meg egy procedúrával, amelynek a bemenete a mátrix sorainak száma lesz. Az elemeket egy A tömbben tároljuk.

```
HAROMSZOG_MATRIX(n)
// Input: n, a matrix sorainak a szama
// Adatbekeres es tarolas
Hossz[A] ← (n · (n + 1))/2
k ← 0
FOR i ← 1 TO n DO
     FOR j ← 1 TO i DO
          INC(k)
          INPUT(a_k)
// Matrix elemeinek kiiratasa
k ← 0
FOR i ← 1 TO n DO
     FOR j ← 1 TO n DO
          IF i ≤ j
               THEN INC(k)
                    OUTPUT(a_k)
               ELSE OUTPUT(0)
     OUTPUT('\n')
```

7. Vizsgáljuk meg, hogy egy mátrix szimmetrikus-e!

```
SZIMMETRIKUS_E(@A)
// Input: A ∈ ℝ^(n×n)
// Output: SZIMMETRIKU_E ∈ {igaz, hamis}
FOR i ← 2 TO n DO
     FOR j ← 1 TO i − 1 DO
          IF a_ij ≠ a_ji
               THEN SZIMMETRIKUS_E ← hamis
                    RETURN
SZIMMETRIKUS_E ← igaz
RETURN
```

8. Fordítsuk meg egy vektor elemeinek a sorrendjét!

```
FORDIT(@X)
// Input: X ∈ ℝ^n
// Output: Az X forditott sorrendben
FOR i ← 1 TO ⌊n/2⌋ DO
    k ← n − i + 1
    t ← x_i
    x_i ← x_k
    x_k ← t
RETURN(X)
```

9. Transzponáljuk egy mátrix elemeit!

   Vizsgáljuk meg a segédmátrixos és helyben transzponálós változatot is!

   A segédmátrixos változat:

```
TRANSPOSE(@A, @B)
// Input: A ∈ ℝ^{n×n}
// Output: B ∈ ℝ^{n×n}
FOR i ← 1 TO n DO
    FOR j ← 1 TO n DO
        b_ij ← a_ji
RETURN(B)
```

   A helyben transzponálós változat:

```
TRANSPOSE(@A)
// Input: A ∈ ℝ^{n×n}
// Output: A ∈ ℝ^{n×n}
FOR i ← 2 TO n DO
    FOR j ← 1 TO i − 1 DO
        t ← a_ij
        a_ij ← a_ji
        a_ji ← t
RETURN(A)
```

   Egy CSERE függvény használatával az utóbbi változat áttekinthetőbbé tehető.

10. Egy egész értékeket tároló vektorban keressünk Pithagoraszi számhármasokat, vagyis amelyekre teljesül, hogy $x^2 + y^2 = z^2$.

```
PITHAGORASZ(@S)
// Input: S ∈ ℝ^n
FOR i ← 1 TO n DO
    FOR j ← 1 TO n DO
        FOR k ← 1 TO n DO
            IF s_i^2 + s_j^2 = s_k^2
                THEN OUTPUT(s_i, s_j, s_k)
RETURN
```

10. gyakorlat

Síkbeli pontok adatait tároljuk egy tömbben (x, y) rendezett párok formájában folytonosan.

$$[\ x_0 \quad y_0 \quad x_1 \quad y_1 \quad \dots \quad x_{n-1} \quad y_{n-1} \quad x_n \quad y_n\ ]$$

Mennyi eleme lesz a vektornak n pont esetén?

1. Írassuk ki a koordináták mellé, hogy melyik pont melyik síknegyedbe esik!
2. Toljuk el az összes pontot egy adott vektorral!
3. Írassuk ki azon pontok koordinátáit, amelyek az origó r sugarú környezetébe esnek!
4. Keressük meg a két legközelebbi és legtávolabbi pontot!
5. Határozzuk meg annak a legkisebb téglalapnak a bal felső és jobb alsó pontjának koordinátáit, amelynek oldalai a tengelyekkel párhuzamosak!
6. Keressük meg azon pontokat, amelyek egy egyenesbe esnek!
7. Általánosan hogy kereshetnénk legkisebb területű téglalapot, négyzetet, kört?
8. Készítsünk egy procedúrát, amely egy szöveges értékről megállapítja, hogy az lehet-e email cím!
9. Vágjunk ki egy sztringből egy adott részt!
10. Konkatenáljunk össze két sztringet!
11. Távolítsuk el egy szövegből a szóközöket!

    Módosítsuk a programot, hogy csak a felesleges/duplikált szóközök kerüljenek eltávolításra!

11. gyakorlat

1. Soroljuk fel a 12 elemi algoritmust!
2. Mik a jó program ismérvei?
3. Mik a struktúrált programok alapelemei?

    Rajzoljuk fel a szekvencia, a ciklus és az elágazás folyamatábráját/programgráfját, illetve írjuk fel a formulájukat és pszeudókódjaikat!

4. Adjuk meg a teljes, és a tömör élhalmazos felírását a következő gráfoknak!
    // TODO: A gráfot még meg kell rajzolni!
    teljes

$$P = \{(START, A), (A, B), (B, C), (C, D/i), (D, E/i), (D, F/h), (E, G),\\ (F, G), (G, H), (H, B), (C, I/h), (I, STOP)\}$$

    tömör

$$P = \{START \to A, A \to C, C \to (D, I), D \to (E, F),\\ E \to H, F \to H, H \to C, I \to STOP\}$$

5. Rajzoljuk fel a programgráfot az élhalmaz alapján!

$$P = \{(START, A), (A, B), (B, C/i), (B, G/h), (G, H), (H, I),\\ (C, D), (D, E), (E, F/i), (E, I/h), (I, J), (J, STOP)\}$$

6. Rajzoljuk fel a programgráfot a pszeudókód alapján!

```
N: A
    IF B
        THEN C
        ELSE GOTO N
```

7. Rajzoljuk fel a programgráfot, és adjuk meg az ekvivalens struktúrált programgráfot!

$$P = \{START \to A, A \to (B, C), B \to (S1, S2), C \to (S2, S3), \\ S1 \to D, S2 \to D, S3 \to D, D \to STOP\}$$

8. Mit nevezünk ciklikus bonyolultságnak, és lényeges bonyolultságnak?

$$m(P) = e - c, \quad m(P) = p + 1, \quad M(P) = m(P) - k$$

12. gyakorlat

1. Adjunk példákat nem valódi programok programgráfjára, amelyek csak adott kritériumoknak nem felelnek meg!

   Kritériumok:

   - Véges számú, nem zérus be- és kimeneti éllel rendelkezik.
   - A programgráf csomópontjai predikátumok, függvények és gyűjtők.
   - A programgráf minden csomópontján keresztül vezet legalább egy útvonal, amely egy bemenő éllel és egy kimenő éllel rendelkezik.

2. A következő, tömör élhalmazos formában megadott programokhoz készítsünk

   - programgráfot
   - írjuk fel a formuláját
   - Írjuk fel az eredeti és a struktúrált program pszeudó kódját
   - Számítsuk ki a ciklikus bonyolultságot
   - Rajzoljuk fel a struktogramot

$$P = \{START \to A, A \to (B, C), B \to (D, E), C \to E, D \to A, E \to STOP\}$$

$$P = \{START \to A, A \to B, B \to (C, D), C \to B, D \to E, E \to (A, STOP)\}$$

$$P = \{START \to A, A \to B, B \to (C, D), C \to B, D \to C, E \to (A, STOP)\}$$

$$P = \{START \to A, A \to (B, C), B \to D, C \to E, D \to (A, F), E \to (A, F), F \to STOP\}$$

$$P = \{START \to A, A \to (B, C), B \to (D, E), C \to A, D \to A, E \to (D, STOP)\}$$

$$P = \{START \to A, A \to (B, C), B \to (D, E), C \to E, D \to A, E \to STOP\}$$

$$P = \{START \to A, A \to B, B \to (C, D), C \to (B, STOP), D \to (C, A)\}$$

$$P = \{START \to A, A \to (B, E), B \to C, C \to (D, G), D \to A, E \to (F, STOP), F \to G, G \to STOP\}$$

$$P = \{START \to A, A \to (D, B), B \to C, C \to (E, STOP), D \to F, E \to A, F \to (B, C)\}$$

3. A pszeudókódok alapján rajzoljuk fel a programgráfot, majd készítsük el az ekvivalens struktúrált programokat.

```
1 IF A
2   THEN B
3       IF C
4           THEN F
5           ELSE GOTO 8
6   ELSE
7       WHILE D DO
8           E
9 G
```

```
1 IF A
2   THEN
3       WHILE B DO
4           C
5       GOTO 10
6   ELSE
7       IF D
8           THEN
9               DO E
10                  WHILE F
11          ELSE G
```

```
1 IF A
2   THEN
3       B
4       IF D
5           THEN GOTO 7
6   ELSE
7       C
8       IF E
9           THEN GOTO 3
```

4. Keressük meg egy valós értékeket tartalmazó tömbben egy procedúrával a paraméterként kapott z értékhez a legközelebbi elemet!

5. Válogassuk szét egy valós tömb elemeit negatív és nemnegatív értékekre!

6. Töröltjük egy tömbből a 0 értékű elemeket! *(a. segédtömbbel, b. segédtömb nélkül)*

7. Írjunk egy procedúrát, amely 2, azonos méretű valós értékeket tároló tömbhöz visszaadja az adott indexeken lévő minimális értékeket.

   pl.: $x = [5, 8, -1], y = [2, 1, 3] \Rightarrow z = [2, 1, -2]$

8. Számítsuk ki két vektor különbségének négyzetösszegét!

13. gyakorlat

COCOMO modell

- PM: Person Month
- T_d: Development Time
- KDSI: Kilo Delivered Source Index

$$PM = 2.4 \cdot KDSI^{1.05}, \quad T_d = 2.5 \cdot \sqrt[3]{PM}, \quad \text{létszám} = \frac{PM}{T_d}, \quad \text{Brook limit} : T_d \cdot 0.75$$

1. Egy 100000 soros alkalmazást szeretnénk kifejleszteni. Mennyi emberre és időre van szükség?

2. 20 programozóval fél év alatt mekkora alkalmazást lehet kifejleszteni?

3. Egy 300000 soros program esetében lehetőségünk van egy függvénykönyvtár megvásárlására, amely 120000 sor kiváltására képes.

   Maximum mennyit érdemes fizetnünk ezért, ha a programozók havi fizetése 240000 Ft?

4. Egy 50000 soros alkalmazást szeretnénk minél gyorsabban elkészíteni. Mennyi plusz embert lehet még bevonni, hogy a határidő tartása ne legyen rizikós?

5. Egy 1000000 soros alkalmazást 25 emberrel szeretnénk kifejleszteni. Az alkalmazást várhatóan 5 millió forintént fog elkelni.

   Legfeljebb mennyi lehet így a programozók havi fizetése?

6. Írjuk át iteratív alakra a következő rekurzív algoritmust!

```
Parity(x, @y)
// Input: x ∈ ℕ
// Output: y ∈ ℕ
IF x = 0
  THEN y ← 1
  ELSE IF x mod 2 = 0
    THEN CALL Parity(x-1, y)
          y ← y + x
    ELSE CALL Parity(x-1, y)
          y ← y · x
RETURN(y)
```

7. Írjuk át iteratív alakra a következő rekurzív algoritmust!

```
ADDER(@X, i, @a, @b)
// Input: X ∈ ℝ, i ∈ ℕ
// Output: a, b ∈ ℝ
IF i > 0
  THEN
    ADDER(X, i - 1, a, b)
    IF x_i > 0
      THEN a ← a + x_i
      ELSE b ← b + x_i
  ELSE
    a ← 0
    b ← 0
RETURN(a, b)
```

8. Írjuk át iteratív alakra a következő rekurzív algoritmust!

   TRNG(x, @y)
   // Input: $x \in \mathbb{R}$
   // Output: $y \in \mathbb{R}$
   IF x > 0
   THEN FOR i ← 1 TO X DO
   CALL TRNG(x-1, y)
   INC(y)
   ELSE y ← 0
   RETURN(y)

9. Készítsünk egy algoritmust, amely úgy osztályozza a tömbben tárolt egészeket, hogy a párosakat a tömb elejére, a páratlanokat a tömb végére helyezi.

10. Készítsünk algoritmust, amely k szerinti maradékosztályok szerint rendezi egy tömb elemeit, vagyis az egyes szakaszok értékeit k-val osztva $0, 1, \ldots k-1$ maradékot adnak.

    Például 4 esetén:

    $$[1, 5, 9, 7, 4, 3, 1, 2, 8, 6, 5] \quad \rightarrow \quad [4, 8, 1, 5, 9, 1, 5, 2, 6, 7, 3]$$

11. Írassuk ki egy vektor elemeinek összes lehetséges permutációját!
    a. Rekurzív algoritmussal
    b. Iteratív algoritmussal

12. Számítsuk ki, hogy egy vektorban melyik értékből mennyi van.
    a. Természetes számokat tároló vektor előre rögzített maximális értékkel
    b. Tetszőleges elemtípusra vektorok vektorával

14. gyakorlat

Az eddigi feladatok gyakorlása.

Kimaradt feladatok

1. Mik lesznek a következő halmazok elemei?
   - $\{x \in \mathbb{R} : x^2 - 2 = 0\}$
   - $\{x \in \mathbb{R} : 2 \log x = \log x^2\}$
   - $\{x \in \mathbb{R} : 1 < |x| \leq 5\}$
   - $\{(x, y) \in \mathbb{R}^2 : x^2 + y^2 = 4 \text{ és } y \geq 0\}$
   - $\{(x, y) \in \mathbb{R}^2 : |x| < y \leq 1\}$

2. Milyen halmazok esetén állnak fenn a következő egyenlőségek?
   - $(A \setminus X) \cap (B \setminus X) = A$
   - $(A \cap B) \cup X = X$
   - $(A \cap X) \cup B = X$
   - $A \setminus X = X \setminus A$

3. Milyen elemek tartoznak a következő halmazokhoz?
   - $\{(a, b) \in \mathbb{N}^2 | a \mod b = 6, a + b = 20\}$

- $\{(a, b) \in \mathbb{N}^2 | a \text{ div } b = 3, a \cdot b < 50\}$
- $\{(x, y) \in \mathbb{R}^2 : \text{Round}(x) + \text{Round}(y) = 3\}$
- $\{(x, y) \in \mathbb{R}^2 : \{x\} > \{y\}\}$

4. Igazoljuk, hogy $(p \wedge q) \to q \equiv p \to (p \vee \overline{q})$ kifejezéssel!

5. Mivel egyenlő az $A \triangle B \triangle C$ kifejezés?

   Előtte be kell látni, hogy kommutatív a kifejezés!

6. Egy sportegyesületnek 550 tagja van. A tagok 20% -a kajakozik, vagy kenuzik. A tagok közül 60-an kajakoznak és 25-en mind a két sportot űzik. Hányan kenuznak?

7. Egy matematika versenyen 2 feladatot tűztek ki. Az elsőt az indulók 80%-a, a másodikat a 40%-a oldotta meg. Minden résztvevő megoldott legalább 1 feladatot, mindkét feladatot ketten oldották meg. Hányan indulhattak a versenyen?

8. Egy faluban 1000 ház van. Ezek közül 250-ben van autó, 900-ban hűtőszekrény, 950-ben televízió és 990-ben rádió. Legalább mennyi házban van mind a négy?

9. Készítsünk egy olyan függvényt, amely a 3 paraméter értékéből a középső elem értékével tér vissza! (med(a, b, c) ...)