Feladatok

1 Alapműveletek alaptípusokkal

1. Olvass be két egész számot és írd ki a
    a. összegüket
    b. különbségüket
    c. szorzatukat
    d. maradékos osztási hányadosukat
    e. osztási maradékukat

2. Olvass be két tört számot és írd ki hogy egyenlőek-e vagy sem.

3. Olvass be két 8 bites egész számot és írd ki, egész mondatban, hogy melyik a nagyobb. (pl.: 5 és 6 estén: 6 nagyobb mint 5. 5 és 5 esetén: 5 egyenlő 5.)

2 Bit operátorok

1. Írj egy programot, ami beolvas egy egész számot, **binárisan összeéseli** 512-vel, majd az eredményt kíjra.
   Ennek a programnak a segítségével adj meg 3-3 olyan különböző számot ami:
    a. Kisebb mint 512 és a program kimenete 0.
    b. Nagyobb mint 512 és a program kimenete 512.
    c. Nagyobb mint 512 és a program kimenete 0.
   (próbáljuk okosan kitalálni, hogy mivel érdemes próbálkozni)

2. Írj egy programot, ami beolvas egy egész számot, **binárisan összevagyolja** 512-vel, majd az eredményt kíjra.
   Ennek a programnak a segítségével adj meg 1-1 olyan különböző számot ami:
    a. Kisebb mint 512 és a program kimenete 548.
    b. Nagyobb mint 512 és a program kimenete 1536.
   (próbáljuk okosan kitalálni, hogy mivel érdemes próbálkozni)

3. Írj egy programot, ami beolvas egy egész számot, **binárisan összekizáróvagyolja** (XOR-olja) 512-vel, majd az eredményt kíjra.
   Ennek a programnak a segítségével adj meg olyan számokat ami:
    a. Kisebb mint 512 és a program kimenete 578.
    b. Nagyobb mint 512 és a program kimenete 1030.
    c. Nagyobb mint 512 és a program kimenete 1534.
   (próbáljuk okosan kitalálni, hogy mivel érdemes próbálkozni)

4. Írj egy programot ami beolvas egy tetszőleges egész számot, majd egy 0 és 4 közötti egész számot. Ha a második szám nem 0 és négy közötti, dorongold le a felhasználót.
   Ezek után írd ki az első szám másodikkal binárisan eltolt értékeit mind a három tanult bináris eltolás operátorral (<<, >>, >>>)

3 Elágazások (if, if ... else, switch)

1. Írj programot ami beolvas egy egész, majd eldönti, hogy az alábbi esetek közül melyik áll fenn:
   a. a szám páratlan
   b. a szám páros és osztható néggyel, de nem osztható nyolccal
   c. a szám páros és osztható nyolccal és néggyel is
   d. a szám páros de nem osztható sem nyolccal sem néggyel
   e. valami más féle szám

2. Kérj be két számot, majd kérd be, hogy számtani vagy mértani középarányost szeretnénk számolni. Számold és írd ki a kért művelet eredményét.

3. Írj programot amiben a felhasnzálónak lehetősége van megadni, hogy egy kör sugarát vagy átmérőjét szeretné megadni. Ezek után kérd be a választott adatot, majd számold ki a kör területét és írd ki.

4. Kérjen be egy számot 1 és 7 között és írja ki, hogy az a szám a hét melyik napjának felel meg (hétfő, kedd, ...). Ha nem megfelelő számot adott meg a felhasználó, írd ki, hogy ejnye bejnye.

5. Kérjen be egy karaktert és döntse el, hogy az kisbetű, nagybetű vagy szám (ne használj beépített függvényeket, hanem a karakterek kódja alapján programozz).

6. Kérjen be egy karaktert és írja ki a nagybetűsített változatát (ne használj beépített függvényeket, hanem a karakterek kódja alapján programozz).

7. Kéjen be három számot, ami évszámot, hónap számot és nap számot jelent. Döntse el, hogy a megadott adatok a 20. század valamelyik napját jelentik-e vagy sem.

8. Kérjen be egy számot 1 és 10 között és írja ki, hogy helló annyiszor. Nem megfelelő szám esetén írjuk ki, hogy ejnye bejnye.

4 Karakterláncok

1. Olvass be egy karakterláncot, majd írd ki
   a. a hosszát
   b. nagybetűsített alakájt (keress beépített függvényt erre)
   c. kisbetüsített alakját(keress beépített függvényt erre)

2. Írj programot mely beolvas két karakterláncot. Majd megnézi, hogy az egyik benne van-e a másikban és ennek megfelelően kírja, hogy az első része a másiknak, a második az elsőnek, egyenlőek, vagy egyik sem.

3. Írj programot ami beolvas két karakterláncot (vezetéknévnek és keresztnévnek), majd ebből kiír egy szépen formázott nevet. A szépen formázott azt jelenti, hogy a vezetéknév és a keresztnév első betűje nagy betű, a többi mind kicsi.

4. Vizsgáld meg, hogy mi történik, ha a 20-at és a 30-at minden lehetséges típuskombinációban összeadod (mindkettőt értelmezhetjük számként (int 20, double 20) vagy karakterláncként ("20")).
   Mely kombinációkat nem engedi a Java végrehajtani, mert nem tudja értelmezni? Amit enged, azoknak mi az eredménye?

5. Írj egy programot ami bekér egy karakterláncot, majd kiírja ezt úgy, hogy az első és utolsó karaktert felcseréli. Figyeljünk, hogy minden hosszúságú karakterláncra jól működjön a program.

6. Írj egy programot ami véletlenszerűen egyenletesen választ egy hossz értéket 5 és 10 között (5 és 10 is lehet), majd generál egy olyan véletlen karakterláncot, aminek hossza ez a szám. A karakterei pedig kizárólag kis betűk, nagy betűk, vagy számok. Pl: hossz 7, karakterlánc: aFst5Gw.

5 Tömbök

1. Írj egy programot, ami beolvas 3 keresztnevet (feltételezhető, hogy amit beír a felhasználó, az keresztnév), majd ezeket fordított sorrendben kiírja.

2. Írj egy programot, ami beolvas 3 keresztnevet (feltételezhető, hogy amit beír a felhasználó, az keresztnév), majd ezeket ábécé sorrendben kiírja. Ne használj beépített rendező függvényt.

3. Írj programot, ami be kér egy számot 1 és 5 között, majd pontosan ennyi keresztnevet kér be és ezeket kiírja fordított sorrendben.

4. Írj egy programot ami bekeri négy autó adatait (márka, szín, rendszám). Feltételezhetted, hogy a megadott adatok mindig értelmesek, ezt nem kelle ellenőrizni. Tárold az adatokat tömbök tömbjeként, úgy, hogy az első dimenzióban az autókat indexeljük, a másodikban pedig az adataikat. Beolvasás után írd is ki őket rendszám szerinti ABC sorrendben.

5. Írj egy programot amely feltölt egy 3 elemű tömböt véletlen számokkal, kiírja a tömb elemeit, megcseréli az első és az utolsó számot, majd újra kiírja a tömböt.

6 Ciklusok (for, while, do ... while,)

1. Kérjen be egy számot és írja ki, hogy helló annyiszor. (Emlékezzünk a régi szenvedős megoldásra és értékeljük a haladást.)

2. Kérjen be egy karakterláncot. Majd ahány karakter van ebben (pl. alma esetén 4, kiskutya esetén 8), annyiszor írja ki az első karaktert (pl. alma esetén aaaa, kiskutya esetén kkkkkkkk).

3. Egyszerű palindrom teszt: Írjon programot amely beolvas egy karakterláncot és arról eldönti, hogy az visszafelé olvasva tökéletesen megegyezik-e önmagával.

4. Rendes palindrom teszt: Írjon programot amely beolvas egy karakterláncot és arról eldönti, hogy palindróm-e (egy karakterlánc akkor palindróm, ha visszafelé olvasva is ugyan az mint előre, pl: Rád rohan a hordár). A kis- és nagybetűk ne jelentsenek különbséget, valamint a szóközök se számítsanak.
   Olyan megoldást adjon, amiben nem használ csak egy karakterláncot, az eredeti bemenetet (újabb karakterláncokba való átalakítás nélkül oldja meg a feladatot). Tesztadatok: https://hu.wikiquote.org/wiki/Magyar_nyelvű_palindromok_listája

5. Írjon programot ami addig olvas be karakterláncokat ameddig azt nem adjuk meg neki, hogy Exit. Minden beolvasott karakterláncot írjon is ki azonnal fordítva, de az Exit-et már ne.

6. Írj egy programot ami bekér két karakterláncot, majd kiírja, hogy a második karakterlánc hánszor szerepel az elsőben. Ezt kétféleképpen is számolja meg:

   i. Átfedésekkel: pl: a „abrabrabra”-ban 3-szor szerepel az „abra”, ha átfedéseket megengedünk.
   ii. Átfedések nélkül: pl: a „abrabrabra”-ban 2-szor szerepel az „abra”, ha átfedéseket nem engedünk meg.

7. Írj egy programot ami bekér egy karakterláncot és az első betű összes előfordulását kicseréli !-re, kétféle képpen. Egyik esetben karakterláncfeldolgozó műveletekkel, a másik esetben egy Java beépített metódus használatával (google a barátod). Ha jól dolgozol, a két megoldás azonos kell legyen.

8. Írj egy programot ami karakterláncokat kér be addig amígy *-ot nem kap, majd kiírja a leghoszabb megadott karakterlánc hosszát.

9. Írj egy programot ami bekér egy karakterláncot, majd törli minden páros helyen álló karakterét és kiírja a képernyőre.

10. Írj egy programot ami bekér egy szöveget. Ezt a szöveget vesszők mentén feldarabolja, majd kiírja az összes előforduló szót, de csak egyszer. Pl.: „alma, körte, alma, kukorica” bemenetre „alma, körte, kukorica” íródik a képernyőre.

11. Írj egy programot, amely egy karakterláncot kér be. Majd minden olyan betűről, ami legalább kétszer szerepel, kiírja, hogy az hányszor szerepelt. Pl: „thequickbrownfoxjumpsoverthelazydog” esetén o 4, e 3, u 2, h 2, r 2, t 2. Bónuszpontért lehet előfordulási gyakoriság szerint csökkenő sorrendben kiírni.

12. Írj egy programot amely feltölt egy 10 elemű tömböt véletlen tört számokkal, kiírja a tömb elemeit, megcseréli a legnagyobb és a legkisebb számot, majd újra kiírja a tömböt.

13. Írj egy programot amely felölt egy 10 elemű egész szám tömböt 2 és 15000 közötti véletlen egész számokkal, kiírja a tömb elemeit sorrendben vesszővel elválasztva, megcseréli a legkisebb és a legnagyobb element, majd újra kiírja a tömb elemeit.
    Mivel a számok hossza eltérő, ezért a kiírás láthatóan igénytelennek hat. Gondoskodj róla, hogy a két kiírás egymás alatti sorokban legyen, és a számokat elválasztó vesszők egymás fölé legyenek igazítva. (Használd fel az előző feladat kódját.)

14. Írj egy programot ami először feltölt egy 5 elemű tömböt véletlen számokkal, majd rendezi és kiírja. Ezek után addig kér be adatot a felhasználótól, amíg az *-ot nem ad meg. Ha nem számot adott meg, akkor kiírja, hogy csak számokat vagy *-ot fogadunk el. Ha számot adott meg, akkor a tömb azon elemét amelynek értéke a legközelebb esik a megadott új számhoz kicseréli az új számra, rendezi a tömböt, majd kiírja.

7 Logikai függvények

1. Fogalmazd át a következő igaz-hamis feltételt a lehető legrövidebb alakba:
   if ((!a && !b) || (b && a) || (!b && a)){...}

2. Fogalmazd át az:
   if (!b && a) {…}
   feltételt óly módon, hogy ne használjon && kapcsolatot. (Tipp: !!x = x.)

3. A HBC parkolója akkor nyit(na) sorompót, ha ismeri a telefonszámunkat, vagy felismeri az autó rendszámát. Felhasználva a
   boolean telefonszamIsmert = ...;
   boolean rendszamIsmert = …;
   változókat, fogalmazd meg a sorompónyitási feltételt két féle képpen.

4. Fogalmazd át a következő igaz-hamis feltételt a lehető legrövidebb alakba:
   if ((!a && !b && !c) || (a && b && !c) || (a && b && c) || (!a && b && !c) || (a && !b && !c) || (!a && b && c)){...}

5. Fogalmazd át a következő igaz-hamis feltételt a lehető legrövidebb alakba:
   if ((!a && b) || (!b && c) || (a && c) || (!a && !c) || (b && c)){...}

6. Fogalmazd át a következő igaz-hamis feltételt a lehető legrövidebb alakba:
   if ((!c && (b || !a)) || (a && (!b || c)) || (b && a)){...}

8 Függvények

1. (padding) Írj egy függvényt ami egy karakterláncot kiegészít adott hosszúságúra. A függvény paraméterként várja a kiegészítendő karakterláncot, a kiegészítéshez használatos karaktert, az elérni kívánt hosszt valamint azt, hogy a karakterlánc elejére vagy végére kerüljenek a kitöltő karakterek.

2. Oldd meg a 6.13 feladatot majd refaktoráld a megoldásodat, hogy függvények felhasználásával oldd meg, jelentősen csökkentve a kód mennyiségét és javítva az olvashatóságot.
   Használd az IDE beépített refaktorálási funkcióit (jobb klick, refactor, extract method).

3. Oldd meg a 6.14 feladatot majd refaktoráld a megoldásodat, hogy függvények felhasználásával oldd meg, jelentősen csökkentve a kód mennyiségét és javítva az olvashatóságot. Figyelj külön a változók logikus kategorizálására (globális, lokális, paraméter,...).
   Használd az IDE beépített refaktorálási funkcióit (jobb klick, refactor, extract method).

4. Írj egy programot ami *-ig kér be neveket. Utána újra *-ig kér be karaktereket. Minden karakter bekérése után kiírja azokat a neveket, amelyek tartalmazzák a megadott karaktert. Ez után felajánlja az új karakter megadási lehetőségét.

5. Írjon egy kávé autómata tesztelését segítő alkalmazást. A program úgy működik, hogy a képernyőre kiír egy 5 menüpontból álló programot:
   a. Espresso, rövid, cukorral – 1
   b. Espresso, rövid, tejjel és cukorral – 1
   c. Espresso, hosszu, cukorral – 1
   d. Espresso, hosszu, tejjel és cukorral – 1
   e. Cappuccino-1
   f. Olasz Cappuccino-1
   g. Moccaccino-2
   h. Forró csoki-2
   i. Kilépés

   A menü kiírása után bekéri az opció menükódját, majd kiírja, hogy „Ön egy XXXX-t vett ZZZZ lejért. Várja meg míg elkészül és fogyassza egészséggel.”

6. Egészítse ki a 8.4 feladatot egy olyan adminisztrátor menüvel, amiben meg lehet változtatni az italok árát. Ha ezt a menüt kéri a felhasználó, akkor újra írjuk ki a lehetséges italokat, amiből ki lehet választani, hogy melyik ára változzon meg. Ez után meg lehet adni az új árat, majd visszalépünk a főmenübe. Innen a változott árakkal íródik ki a jókívánság.
   Gondolja át, hogy mely változókat érdemes átalakítani globálissá, illetve a függvényeket is szükséges lehet átszervezni az eredeti megoldásának függvényében.

7. Írjon egy programot ami bekér egy számot, majd létrehoz egy ekkora tömböt amit megtölt véletlen számokkal. Ezek után egy másik tömbbe számoljuk meg, hogy az adott pozícióban lévő elemhez képest az eredeti tömbben hány nagyobb elem volt.
   A tömbhossz megadása utáni futási időt mérd meg és az algoritmus futása után írd ki mp-ben.

Példa: első tömb: {0.7, 0.6, 0.3, 0.9, 0.2, 0.8}
második tömb: { 2, 3, 4, 0, 5, 1}

9 Komplexitás

1. Számold ki a 8.7 feladat közelítő komplexitását. Ezek után futtasd a programot különböző bemenetekre és vizsgáld meg, hogy mit tapasztalsz. Mennyire van ez összhangban az elméleti számolásoddal?

2. A 6. alfejezet összes feladatának számold végig a komplexitását. Válaszd ki a 2-3 bonyolultabbat és azokra végezd el az előző feladatban megnézett mérést, hogy ellenőrizd jól gondolkodtál-e.

3. Az előző feladat folytatása. Szervezze át az előző feladatban kiválasztott feladatot, úgy hogy egy függvény legyen. Ezek után írjon programot, ami különböző bemenetekre leméri az így kapott függvény futási idejét. Minden adott bemeneti méretre végezzen 100 mérést, majd írja ki a képernyőre a minden bemeneti méret esetén a 100 odavágó mérés átlagát és szórását.

10 Véletlen számok

1. Írj egy programot, ami egy számolást hajt végre 100000000 alkalommal, majd a részeredmények átlagát kiírja. A részeredmények egész számok, amelyeket a következőképpen kapunk meg. 0 és 1 közötti véletlen számokat kérünk, és ezeket addig adjuk össze, amíg az összeg nagyobb lesz mint 1. Az így összeadott véletlen számok száma a vizsgált mennyiség.
   Pl.: Ha 100000000 helyett csak három részeredmény átlagát számoljuk, és a véletlen számok a következő sorrendben érkeznek

0.164074	0.713649	0.823336	0.011426	0.993275	0.991575	0.386579

   , akkor részeredmények a következőképpen alakulnak:

3 véletlen szám	0.164074 + 0.713649 + 0.823336 = 1.701060
2 véletlen szám	0.011426 + 0.993275 = 1.004702
2 véletlen szám	0.991575 + 0.386579 = 1.378154

   Így a fenti 3 alkalommal megismételt számolások átlaga (3+2+2)/3 = 2.(3).

11 Átfogó

1. Írjon egy olyan programot, amivel sakkozni lehet. A program a képernyőre karakterekkel kirajzolja az induló állást, majd egyszer a fekete, egyszer a fehér játékostól kéri be a következő lépését (melyik mezőről melyikre szeretne lépni). Ha *-ot ad meg a játékos, akor feladta a mérkőzést.
   A program ellenőrizze minden lépésnél, hogy azt a bábú megteheti-e (a típusa engedi, király nem kerül sakkba,...).
   A kiírásban jelülje _-al a fekete üres cellák helyét és szóközzel a fehéreket. Minden bábúnak válasszon egy betűt (F és f a futónak, és így tovább). A kis betűs lehet a fehér a nagy a fekete jelölője.