1. Vágjon föl egy hosszú stringet 5 karakter hosszú darabokra. Rakja össze a darabokat fordított sorrendben.
2. Tróbálja megírni a megtalal() függvényt, ami az ellenkezőjét csinálja, mint amit az indexoperátor (vagyis a []) tesz. Ahelyett, hogy egy megadott indexhez megtalálja az annak megfelelő karaktert, ennek a függvénynek egy adott karakterhez tartozó indexet kell megtalálni.
   Máshogyan fogalmazva, egy olyan függvényt kell írni, ami két argumentumot vár : a kezelendő karakterlánc nevét és a keresendő karaktert. A függvénynek visszatérési értékként az első ilyen karakter stringbeli indexét kell megadni a karakterláncban. Így például : print megtalal("Juliette &
   Roméo", "&") eredménye : 9
   Figyelem : Gondolni kell minden lehetséges esetre. Arra is számítanunk kell, hogy a függvény visszatérési értékként egy speciális értéket (például -1et) ad, ha a keresett karakter nincs a karakterláncban.
3. Tökéletesítse az előző gyakorlat függvényét úgy, hogy egy harmadik paramétert ad hozzá : azt az indexet, amelyiktől kezdve keresni kell a karakterláncban. Így például a következő utasításnak : 15 öt kell kiírni (és nem 4 et!)
   print megtalal("César & Cléopâtre", "r", 5)
4. Írjon egy karakterszam() függvényt, ami megszámolja, hogy egy karakter hányszor fordul elő egy stringben. Így a következő utasításnak 4 et kell kiírni :
   print karakterszam("ananas au jus","a")
5. Egy amerikai mesében nyolc kiskacsát rendre : Jack, Kack, Lack, Mack, Nack, Oack, Pack és Qacknak hívnak. Írjon egy scriptet, ami ezeket a neveket a következö két stringből állítja elő : prefixes = 'JKLMNOPQ' és suffixe = 'ack'
   Ha egy for ... in ... utasítást alkalmaz, akkor a scriptjének csak két sort kell tartalmazni.
6. Határozza meg egy adott mondatban a szavak számát.
7. Írjon egy nagybetu() függvényt, aminek a visszatérési értéke akkor « igaz », ha az argumentuma nagybetű.
8. Írjon egy nagybetu() függvényt, aminek akkor «igaz» a visszaérési értéke, ha az argumentuma nagybetű (használjon az előzőekőll eltérő módszert).
9. Írjon egy függvényt, aminek a visszatérési értéke akkor « igaz », ha az argumentuma szám.
10. Írjon egy függvényt, ami egy mondatot szavakból álló listává alakít át.
11. Használja fel az előző gyakorlatokban definiált függvényeket egy olyan script írására, ami ki tudja szedni egy szövegből az összes nagybetűvel kezdődő szót.
12. Írjon egy függvényt, aminek akkor « igaz » a visszaérési értéke, ha az argumentuma egy alfabetikus karakter (nagy vagy kisbetű). Alkalmazza ebben az új függvényben az előzőekben definiált kisbetu() és nagybetu() függvényeket.
13. Írjon egy függvényt, aminek akkor « igaz » a visszaérési értéke, ha az argumentuma egy szám.
14. Írjon egy függvényt, ami az argumentumaként megadott mondatban lévő nagybetűk számát adja meg visszaérési értékként.

Az ord(ch) függvény bármilyen karaktert elfogad argumentumként. Visszatérési értékként a karakternek megfelelő ASCII kódot adja meg. Tehát ord('A') visszatérési értéke 65.

A chr(num) függvény ennek pontosan az ellenkezőjét teszi. Az argumentumának 0 és 255 közé eső számnak kell lenni, a 0t és 255öt is beleértve. Visszatérési értékként a megfelelő ASCII karaktert kapjuk meg : tehát chr(65) az A karaktert adja vissza.

15. Írjon egy egy ASCII kódtáblát kiíró scriptet. A programnak minden karaktert ki kell írni. A táblázat alapján állapítsa meg a nagybetűs és kisbetűs karaktereket összekapcsoló relációt minden egyes karakterre.
16. Az előző gyakorlatban megtalált reláció alapján írjon egy függvényt, ami egy mondat valamennyi karakterét kisbetűre írja át.
17. Ugyanennek a relációnak az alapján írjon egy függvényt, ami valamennyi kisbetűt nagybetűvé alakít át és viszont (az argumentumként megadott mondatban).
18. Írjon egy függvényt, ami megszámolja, hogy az argumentumként megadott karakter hányszor fordul elő egy adott mondatban.
19. Írjon egy függvényt, ami visszatérési értékként megadja egy adott mondatban a magánhangzók számát.
20. Írjon egy scriptet, ami a 20 tól 40 ig terjedő számok négyzeteit és köbeit állítja elő.
21. Írjon egy scriptet, ami 5°onként automatikusan előállítja a 0° és 90° közé eső szögek sinusait. Figyelem : a math modul sin() függvénye úgy tekinti, hogy a szögek radiánban vannak megadva
    (360° = 2 p radián)
22. Írjon egy scriptet, ami a 2, 3, 5, 7, 11, 13, 17, 19 es szorzótáblák első 15 tagját állítják elő a képernyőn a következő táblázathoz hasonlóan (ezeket a számokat egy listába fogjuk elhelyezni):
    2 4 6 8 10 12 14 16 18 20 22 24 26 28 30
    3 6 9 12 15 18 21 24 27 30 33 36 39 42 45
    5 10 15 20 25 30 35 40 45 50 55 60 65 70 75
    stb.
23. Legyen adott a következő lista :
    ['JeanMichel', 'Marc', 'Vanessa', 'Anne', 'Maximilien', 'AlexandreBenoît', 'Louise']
    Írjon egy scriptet, ami kiírja a neveket és a nevekben lévő karakterek számát.
24. Van egy egész számokat tartalmazó lista, amiben egyes számok többször is előfordulnak. Írjon egy scriptet, ami a listát úgy másolja át egy másik listába, hogy figyelmen kívül hagyja a többszöri előfordulásokat. A végső listának rendezettnek kell lenni.
25. Írjon egy scriptet, ami megkeresi egy adott mondatban a leghosszabb szót (a program felhasználójának be kell tudnia írni egy általa választott mondatot).
26. Írjon egy scriptet, ami kiíratja egy csütörtöki nappal kezdődő képzeletbeli év napjainak a listáját. A scriptben három lista lesz : az egyik a hét napjainak a neveit fogja tartalmazni, a másik a hónapok neveit, a harmadik pedig hogy hány naposak a hónapok (a szökőéveket nem vesszük figyelembe).
    Példa :
    Január 1 csütörtök Január 2 péntek Január 3 szombat Január 4 vasárnap
    ... és így tovább December 31 csütörtök ig.
27. Legyenek adottak a következő listák :
    t1 = [31,28,31,30,31,30,31,31,30,31,30,31]
    t2 = ['Január', 'Február', 'Március', 'Április', 'Május', 'Június', Július', 'Augusztus', 'Szeptember', 'Október', 'November', 'December']
    Írjunk egy kis programot, ami a második listába úgy szúrja be az első lista összes elemét, hogy minegyik hónap nevét az illető hónap napjainak száma követi :
    ['Január',31,'Február',28,'Március',31, stb...].
28. Hozzunk létre egy A listát, ami tartalmaz néhány elemet. Hozzuk létre ennek egy valódi másolatát az új B változóban. Ötlet : először hozzunk létre egy csupa nullákat tartalmazó B

listát, aminek a mérete megegyezik az A lista méretével. Ezután helyettesítsük a nullákat az A elemeivel.

29. Ugyanaz a probléma, de más az ötlet : először hozzunk létre egy üres B listát. Ezután töltsük azt fel az A elemeinek segítségével.
30. Ugyanaz a probléma, de megint más az ötlet : a B lista létrehozásához az A listában vágjunk egy olyan szeletet, ami az összes elemet tartalmazza (a [:] operátor segítségével.)
31. Egy prímszám olyan szám, aminek pontosan két osztója van, egy és önmaga. Írjon programot, ami az eratosztenészi szita módszerének alkalmazásával az összes 1 és 1000 közötti prímszámot előállítja :
    Hozzon létre egy 1000 elemből álló listát, minden listaelem kezdőértéke 1 legyen.
    Járja be ezt a listát a 2 indexű elemtől kezdve :
    ha a vizsgált elem értéke 1, akkor tegye nullává a lista azon elemeit, melyeknek indexe egész számú többszöröse az aktuális indexnek amikor így bejárta az egész listát, akkor azoknak az elemeknek az indexei lesznek a keresett prímszámok, mely elemek értéke 1 maradt.
32. Írjon egy scriptet, ami véletlen szerűen húz kártyalapokat. A kihúzott kártya nevét korrekten kell, hogy megadja. A program például kiírja :
    Nyomjon <Enter>t egy lap húzásához :
    Treff 10