1. Kupac (PL)

Kupacnak nevezünk egy bináris fát, ha teljesíti a kupacfeltételt, azaz minden csomópontban az ott tárolt szám kisebb, mint a két fiában tároltak.

Írjon az alábbi fejkommentnek megfelelõ programot!

```
:- type kupac ---> elem(int, kupac, kupac) | null.
helyeskupac(+K): K kupac kielégíti a kupacfeltételt.
```

2. Lista megfordítása (PL)

Írjon az alábbi fejkommentnek megfelelõ programot!

```
reverse(+L1,?L2) az L2 lista az L1 lista megfordítottja.
```

3. Palindrómák (PL)

Írjon az alábbi fejkommentnek megfelelõ programot!

```
palindroma(+L) az L lista palindróma
```

4. Anagrammák (PL)

Írjon az alábbi fejkommentnek megfelelõ programot!

```
anagramma(+L1,?L2): L1 és L2 listák anagrammák, azaz @= szerint ugyanazon elemeket
tartalmazzák más-más permutációban
```

5. Átfogó (SML)

Írjon az alábbi fejkommentnek megfelelõ programot!

```
val atfogo : real * real -> real
atfogo (a,b) = az a és b befogójú derékszögû háromszög átfogója
```

PRE:

```
a >= 0, b >= 0
```

6. Karakter elkódolás (SML)

Írjon az alábbi fejkommentnek megfelelõ programot!

```
encode : Char -> Char
encode c = c karakter eggyel elrotálva az abc-ben
```

PRE:

```
c az angol abc nagybetûje
```

7. Lista megfordítása (SML)

Írjon az alábbi fejkommentnek megfelelõ programot!

```
rev : 'a list -> 'a list
rev l = az l lista megfordítottja
```

8. Palindrómák (SML)

Írjon az alábbi fejkommentnek megfelelõ programot!

```
palindroma : string -> bool
palindroma(s) igaz akkor és csak akkor, ha s palindróma
```

Pl:

```
palindroma "indulagörögaludni" = true
```

9. Lista darabolás 1 (SML)

Írjon az alábbi fejkommentnek megfelelõ programot!

```
val ndropwhile : ('a -> bool) -> 'a list -> 'a list * int
ndrophwile p l = egy pár, melynek elsõ tagja az l lista maradéka, ha elhagyjuk
        az elejérõl azokat az elemeket, melyekre a p predikátum igaz értéket ad,
        második tagja pedig az elhagyott elemek száma.
```

Pl:

```
ndropwhile (fn x => x > 3) [4, 5, 1, 6, 2] = ([1, 6, 2], 2)
```

10. Lista darabolás 2 (SML)

Írjon az alábbi fejkommentnek megfelelõ programot!

```
val ntakewhile : ('a -> bool) -> 'a list -> 'a list * int
ntakewhile p l = egy pár, melynek elsõ tagja az l lista elején álló olyan elemek
```

```
        listája, melyekre a p predikátum igaz értéket ad, második tagja pedig ezen
        elemek száma.
```

Pl:

```
ntakewhile (fn x => x > 3) [4, 5, 1, 6, 2] = ([4, 5], 2)
```

11. Lista darabolás 3 (SML)

Írjon az alábbi fejkommentnek megfelelő programot!

```
val takeanddrop : int -> 'a list -> 'a list * 'a list
takeanddrop n l = egy pár, melynek elsõ tagja az l lista elsõ n eleméből képzett
        lista, második tagja pedig a maradék elemek listája.
```

Pl:

```
takeanddrop 3 [4, 5, 1, 6, 2] = ([4, 5, 1], [6, 2])
```

12. Számrendszerváltás (SML)

Írjon az alábbi fejkommentnek megfelelő programot!

```
val szamrendszer : int -> int -> int list
szamrendszer n x = az x szám n-es számrendszerben vett jegyeinek listája.
```

PRE:

```
n > 1
```

Pl:

```
szamrendszer 2 11 = [1, 0, 1, 1]
```

Pl:

```
szamrendszer 15 201 = [13, 6]
```

13. Csalós számrendszerváltás (SML)

Írjon az alábbi fejkommentnek megfelelő programot!

```
dec2n : int -> int -> int
dec2n n num = az a szám, aminek tizes számrendszerben felírt alakja meg-
              egyezik num n-es számrendszerben felírt alakjával
```

PRE:

```
2 <= n <= 10, num >= 0
```

Pl:

```
dec2n 2 11 = 1011
```

14. Anagrammák (SML)

Írjon az alábbi fejkommentnek megfelelő programot!

```
anagramma : string * string -> bool
anagramma(s1,s2) igaz akor és csak akkor, ha s1 és s2 füzérek anagrammák
```

Pl:

```
anagramma ("matek tanar", "mertan atka") = true
```